二分答案、树状数组、区间最值查询

陈劭源

Feb 12, 2019

二分答案

有的时候，我们会遇到这样一类问题：

> 求最小（或最大）的 x，使得 x 满足某一性质 P（记为 $P(x)$）
> 其中，性质 P 满足单调性，即如果 $P(x)$，那么对于任意 $y > x$（或 $y < x$），有 $P(y)$。

二分答案

有的时候，我们会遇到这样一类问题：

> 求最小（或最大）的 x，使得 x 满足某一性质 P（记为 $P(x)$）
> 其中，性质 P 满足单调性，即如果 $P(x)$，那么对于任意 $y > x$（或 $y < x$），有 $P(y)$。

这类问题可以用二分答案的方法解决。若答案所在的范围为 $[l, r]$，我们可以检查 $m = (l + r)/2$ 是否满足答案，然后根据单调性将答案所在范围减半。这样，就可以把最优化问题转化为 $\log N$ 次判定问题，如果求解判定问题的时间复杂度为 $O(N)$，那么总的时间复杂度为 $O(N \log N)$。

二分答案

以下是二分答案的代码示例，用于求最小的 x，使得 x 满足某一单调性质 P。

- check(x)用于检查 x 是否满足性质 P；
- $[l, r]$ 为答案所在区间；
- 当(l+r)/2可能发生溢出时，可以改写成l+(r−l)/2。

```
int solve(int l, int r) {
    while (l < r) {
        int mid = (l + r) / 2;
        if (check(mid)) r = mid;
        else l = mid + 1;
    }
    return r;
}
```

练习

洛谷 P2678：二分答案练习题

练习

洛谷 P2678：二分答案练习题

有 N 块岩石，移走不超过 M 块岩石，使得相邻两块岩石之间的距离不能小于 x。求最大的 x。

练习

洛谷 P2678：二分答案练习题

有 N 块岩石，移走不超过 M 块岩石，使得相邻两块岩石之间的距离不能小于 x。求最大的 x。

显然满足二分性质。问题变成，给定 x，要移除不超过 M 块岩石，使得相邻两块岩石之间的距离不能小于 x。

练习

洛谷 P2678：二分答案练习题

有 N 块岩石，移走不超过 M 块岩石，使得相邻两块岩石之间的距离不能小于 x。求最大的 x。

显然满足二分性质。问题变成，给定 x，要移除不超过 M 块岩石，使得相邻两块岩石之间的距离不能小于 x。

这个问题怎么做？直接贪心就可以了！

树状数组

树状数组是一种数据结构，用于维护一个数组 A，并且支持在 $O(\log n)$ 的时间复杂度内完成以下操作

树状数组

树状数组是一种数据结构，用于维护一个数组 A，并且支持在 $O(\log n)$ 的时间复杂度内完成以下操作

1. （**查询前缀和**） 求出 A 的前 n 项和；
2. （**单点修改**） 将 A 的第 n 项的值增加 x。

树状数组

如何利用树状数组实现以下操作？

1. 求 $A[l], A[l+1], \cdots, A[r]$ 的和；
2. 将第 i 项的值修改为 x。

树状数组

如何利用树状数组实现以下操作？

1. 求 $A[l], A[l+1], \cdots, A[r]$ 的和；
2. 将第 i 项的值修改为 x。

解答：

1. 求出前 r 项的和，然后减去前 $l-1$ 项的和即可；

树状数组

如何利用树状数组实现以下操作？

1. 求 $A[l], A[l+1], \cdots, A[r]$ 的和；
2. 将第 i 项的值修改为 x。

解答：

1. 求出前 r 项的和，然后减去前 $l-1$ 项的和即可；
2. 先查出第 i 项的和（前 i 项的和减去前 $i-1$ 项的和），然后加上它们的差值。

树状数组

lowbit：对于正整数 n，将 n 的二进制表示中，除了最低的那个 1 以外，其他的 1 全部置 0 后的值记为 `lowbit(n)`。

树状数组

lowbit：对于正整数 n，将 n 的二进制表示中，除了最低的那个 1 以外，其他的 1 全部置 0 后的值记为 `lowbit(n)`。

例如：6 的二进制表示为 110_2，其 lowbit 值为 010_2，即 `lowbit(6)=2`。

树状数组

lowbit：对于正整数 n，将 n 的二进制表示中，除了最低的那个 1 以外，其他的 1 全部置 0 后的值记为 `lowbit(n)`。

例如：6 的二进制表示为 110_2，其 lowbit 值为 010_2，即 `lowbit(6)=2`。

下面是前 10 个正整数的 `lowbit`值：

n	1	2	3	4	5	6	7	8	9	10
`lowbit(n)`	1	2	1	4	1	2	1	8	1	2

树状数组

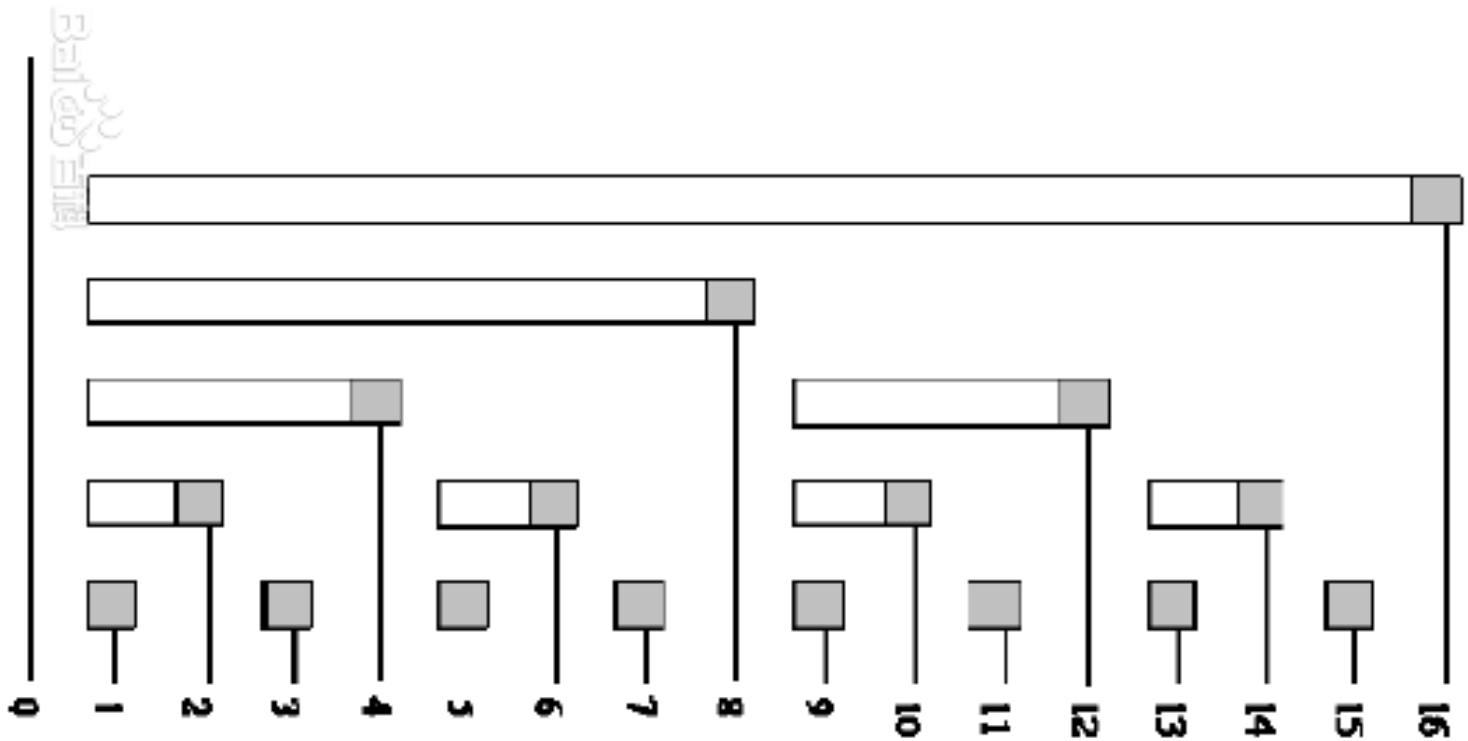
树状数组：数组 A 的树状数组 C 是和 A 长度相同的数组，其定义如下

$$C[i] = \sum_{i-\text{lowbit(i)}+1}^{i} A[i]$$

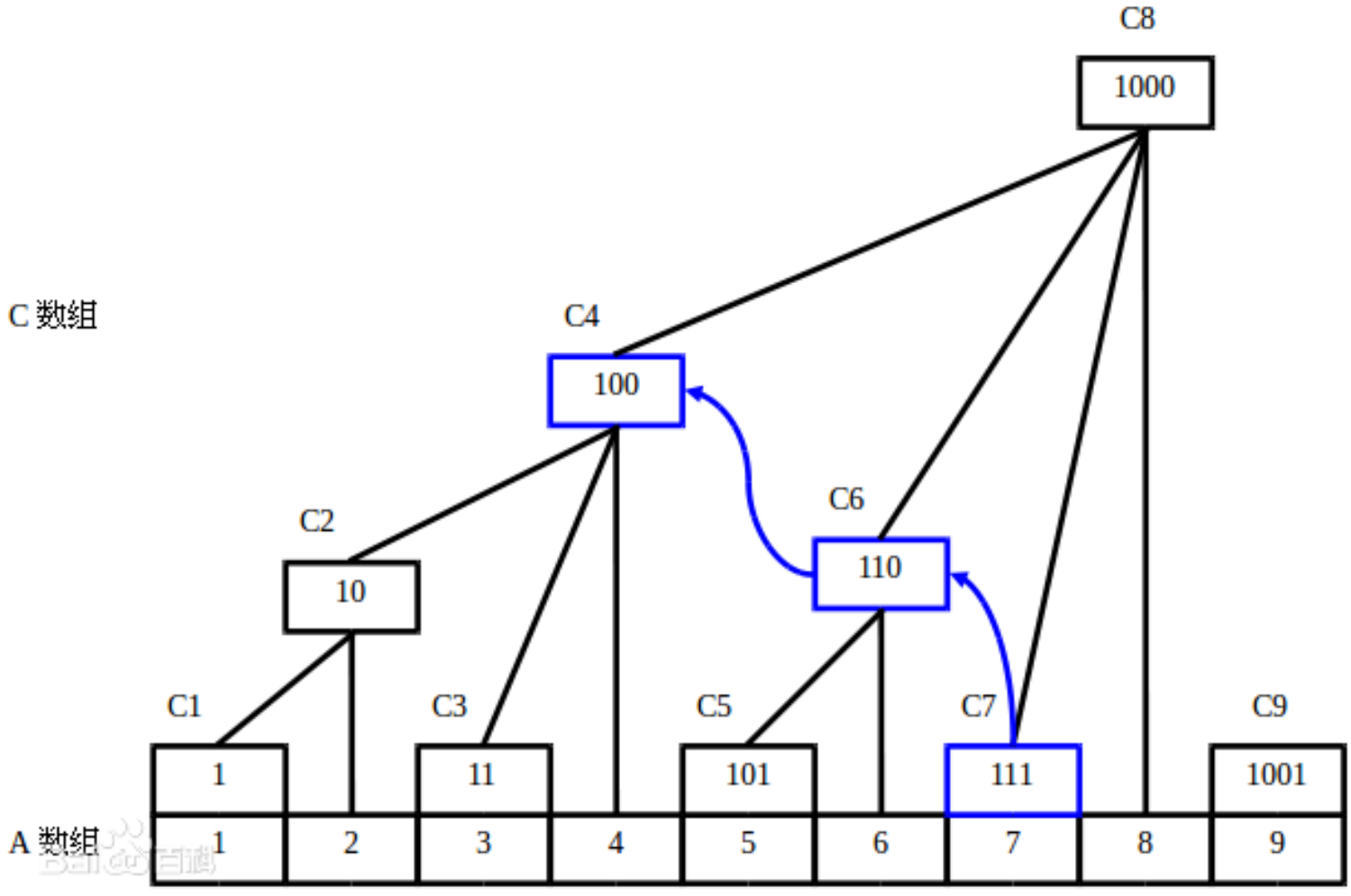
树状数组

树状数组：数组 A 的树状数组 C 是和 A 长度相同的数组，其定义如下

$$C[i] = \sum_{i-\text{lowbit(i)}+1}^{i} A[i]$$



树状数组


C8
1000
C 数组
C4
100
C6
110
C2
10
C1
C3
C5
C7
C9
1
11
101
111
1001
A 数组
1
2
3
4
5
6
7
8
9

树状数组

树状数组操作的代码实现如下：

```
int lowbit(int x) { return x & -x; }
const int MAXN = 1000005;
int bit[MAXN];
int sum(int n) {
    int ans = 0;
    while (n) {
        ans += bit[n];
        n -= lowbit(n);
    }
    return ans;
}
void add(int n, int x) {
    while (n < MAXN) {
        bit[n] += x;
        n += lowbit(n);
    }
}
```

练习

洛谷 P3374：树状数组模板题

练习

洛谷 P3374：树状数组模板题

洛谷 P3368：要实现一个数据结构，支持区间加法和单点查询

练习

洛谷 P3374：树状数组模板题

洛谷 P3368：要实现一个数据结构，支持区间加法和单点查询

用树状数组维护原数组的差分数组，区间加法就可以用单点加法实现，单点查询则可以用区间查询实现。

树状数组的应用：求逆序对数

逆序对：对于数组 A，如果 $i < j$ 而 $A[i] > A[j]$，则称 (i,j) 是 A 的一个逆序对

树状数组的应用：求逆序对数

逆序对：对于数组 A，如果 $i < j$ 而 $A[i] > A[j]$，则称 (i,j) 是 A 的一个逆序对

逆序对的性质

1. 当数组升序排列时，无逆序对；当数组降序排列时，每个 (i,j) $(i < j)$ 都是逆序对；

树状数组的应用：求逆序对数

逆序对：对于数组 A，如果 $i<j$ 而 $A[i]>A[j]$，则称 (i,j) 是 A 的一个逆序对

逆序对的性质

1. 当数组升序排列时，无逆序对；当数组降序排列时，每个 (i,j) $(i<j)$ 都是逆序对；
2. 当数组不是升序排列时，总存在形如 $(i,i+1)$ 的逆序对；

树状数组的应用：求逆序对数

逆序对：对于数组 A，如果 $i<j$ 而 $A[i]>A[j]$，则称 (i,j) 是 A 的一个逆序对

逆序对的性质

1. 当数组升序排列时，无逆序对；当数组降序排列时，每个 (i,j) $(i<j)$ 都是逆序对；
2. 当数组不是升序排列时，总存在形如 $(i,i+1)$ 的逆序对；
3. 逆序对数等于将数组升序排序所需最少的交换相邻元素的次数。

树状数组的应用：求逆序对数

利用树状数组求数列 A 中的逆序对的方法：

1. 将数组 A 离散化（即用每个值的名次替换这个值）；

树状数组的应用：求逆序对数

利用树状数组求数列 A 中的逆序对的方法：

1. 将数组 A 离散化（即用每个值的名次替换这个值）；
2. 依次扫描 A 中的元素 $A[i]$，首先在树状数组中查询第 $A[i]$ 项之后所有项之和，然后对第 $A[i]$ 项加 1。

树状数组的应用：求逆序对数

利用树状数组求数列 A 中的逆序对的方法：

1. 将数组 A 离散化（即用每个值的名次替换这个值）；
2. 依次扫描 A 中的元素 $A[i]$，首先在树状数组中查询第 $A[i]$ 项之后所有项之和，然后对第 $A[i]$ 项加 1。

树状数组的应用：求逆序对数

利用树状数组求数列 A 中的逆序对的方法：

1. 将数组 A 离散化（即用每个值的名次替换这个值）；
2. 依次扫描 A 中的元素 $A[i]$，首先在树状数组中查询第 $A[i]$ 项之后所有项之和，然后对第 $A[i]$ 项加 1。

练习：洛谷 P1908

树状数组的应用：求逆序对数

利用树状数组求数列 A 中的逆序对的方法：

1. 将数组 A 离散化（即用每个值的名次替换这个值）；

树状数组的应用：求逆序对数

利用树状数组求数列 A 中的逆序对的方法：

1. 将数组 A 离散化（即用每个值的名次替换这个值）；
2. 依次扫描 A 中的元素 $A[i]$，首先在树状数组中查询第 $A[i]$ 项之后所有项之和，然后对第 $A[i]$ 项加 1。

树状数组的应用：求逆序对数

利用树状数组求数列 A 中的逆序对的方法：

1. 将数组 A 离散化（即用每个值的名次替换这个值）；
2. 依次扫描 A 中的元素 $A[i]$，首先在树状数组中查询第 $A[i]$ 项之后所有项之和，然后对第 $A[i]$ 项加 1。

树状数组的应用：求逆序对数

利用树状数组求数列 A 中的逆序对的方法：

1. 将数组 A 离散化（即用每个值的名次替换这个值）；
2. 依次扫描 A 中的元素 $A[i]$，首先在树状数组中查询第 $A[i]$ 项之后所有项之和，然后对第 $A[i]$ 项加 1。

练习：洛谷 P1908

区间最值查询

区间最值查询（RMQ）问题：给定一个数组 A，每次询问一个区间 $[L, R]$，求 $\max\{A[L], A[L+1], \cdots, A[R]\}$。

区间最值查询

区间最值查询（RMQ）问题：给定一个数组 A，每次询问一个区间 $[L, R]$，求 $\max\{A[L], A[L+1], \cdots, A[R]\}$。

这里介绍一个叫**稀疏表（ST 表）**的数据结构，经过 $O(n \log n)$ 时间的预处理后，可以在 $O(1)$ 时间内回答每个询问。

区间最值查询

ST 表存放了每个长度为 2 的次幂的区间的最大值：

$$ST[i][k] = \max\{A[i], A[i+1], \cdots, A[i+2^k-1]\}$$

区间最值查询

ST 表存放了每个长度为 2 的次幂的区间的最大值：

$$ST[i][k] = \max\{A[i], A[i+1], \cdots, A[i+2^k-1]\}$$

对于询问 $[l, r]$，令 $k = \lfloor \log(r-l+1) \rfloor$，则这段区间恰好可以用两个长度为 2^k 的区间覆盖：

$$\max\{A[l], A[l+1], \cdots, A[r]\} = \max\{ST[l][k], ST[r+1-2^k][k]\}$$

区间最值查询

ST 表存放了每个长度为 2 的次幂的区间的最大值：

$$ST[i][k] = \max\{A[i], A[i+1], \cdots, A[i+2^k-1]\}$$

对于询问 $[l, r]$，令 $k = \lfloor \log(r-l+1) \rfloor$，则这段区间恰好可以用两个长度为 2^k 的区间覆盖：

$$\max\{A[l], A[l+1], \cdots, A[r]\} = \max\{ST[l][k], ST[r+1-2^k][k]\}$$

而 ST 表可以很容易用递推的方法求出

$$ST[i][k] = \max\{ST[i][k-1], ST[i+2^{k-1}][k-1]\}$$

区间最值查询

ST 表的代码实现

```
const int MAXN = 100005;
int a[MAXN];
int st[MAXN][30];

void build(int n) {
    int l = log2(n);
    for (int i = 0; i < n; i++) st[i][0] = a[i];
    for (int j = 0; j < l; j++)
        for (int i = 0; i <= n - (1 << j); i++)
            st[i][j+1] = max(st[i][j], st[i+(1<<j)][j]);
}

int query(int l, int r) {
    int k = log2(r - l + 1);
    return max(st[l][k], st[r-(1<<k)+1][k]);
}
```

区间最值查询

练习：洛谷 P3865