**树状数组总结及模板（包括一维和二维）**

2009-08-13 13:54

|  |
| --- |
| 树状数组总结：假设c[]为树状数组，a[]为原数组，则两者之间存在这么一个关系，c[i]代表的意义是从a[i]开始往前2^k个元素的和 (k为i化成二进制后尾部包含的0的个数)，举例来说就是：(括号后面的数字代表几进制) c[1] = a[1]( (1)10 -> (1)2 -> k=0 ) c[2] = a[1] + a[2]( (2)10 -> (10)2 -> k=1 ) c[3] = a[3]( (3)10 -> (11)2 -> k=0 ) c[4] = a[1] + a[2] + a[3] + a[4]( (4)10 -> (100)2 -> k=2 ) ...... 有位运算的性质可以得到：对于i来说，i&(-i) = 2^k，那么树状数组的基本功能函数自己手算一下就能明白了，下面附上模板：  树状数组模板：元素下标均从1开始  (1).一维树状数组：一定要注意数组c的初始化，N是数组的大小，modify的功能是把第n个元素加上delta，而sum的功能则是求从第一个元素开始到第n个元素之和，当然也可以求一段区间上的和，比如我要求第x个元素到第y个元素的和，答案就是sum(y) - sum(x - 1)  **int** lowbit( **int** n ) { **return** n & (-n); }  **void** modify( **int** n, **int** delta ) { **while**( n <= N ) { c[n] += delta; n += lowbit(n); } }  **int** sum( **int** n ) { **int** ret = 0; **while**( n != 0 ) { ret += c[n]; n -= lowbit(n); } **return** ret; }  (2).二维树状数组：同样不要忘记c的初始化，modify 的功能是改变元素(x, y)，sum的功能则是求从元素(1, 1)开始到(x, y)的总和，同样，可以求出任意一个子矩阵内的所有元素之和，即sum(x2, y2) - sum(x1-1, y2) - sum(x2, y1-1) + sum(x1-1, y1-1)  **int** lowbit( **int** x ) { **return** x & (-x); }  **void** modify( **int** x, **int** y, **int** delta ) { **int** i, j; **for**(i=x; i<=N; i+=lowbit(i)) { **for**(j=y; j<=N; j+=lowbit(j)) { c[i][j] += delta; } } }  **int** sum( **int** x, **int** y ) { **int** res = 0, i, j; **for**(i=x; i>0; i-=lowbit(i)) { **for**(j=y; j>0; j-=lowbit(j)) { res += c[i][j]; } } **return** res; } |