带修改莫队

请确保您已经会普通莫队算法了。如果您还不会，请先阅读前面的“普通莫队算法”。

特点

普通莫队是不能带修改的。

我们可以强行让它可以修改，就像 DP 一样，可以强行加上一维 **时间维**，表示这次操作的时间。

时间维表示经历的修改次数。

即把询问 $[l,r]$ 变成 $[l,r,time]$ 。

那么我们的坐标也可以在时间维上移动，即 $[l,r,time]$ 多了一维可以移动的方向，可以变成：

- $[l-1,r,time]$
- $[l+1,r,time]$
- $[l,r-1,time]$
- $[l,r+1,time]$
- $[l,r,time-1]$
- $[l,r,time+1]$

这样的转移也是 $O(1)$ 的，但是我们排序又多了一个关键字，再搞搞就行了。

可以用和普通莫队类似的方法排序转移，做到 $O(n^{\frac{5}{3}})$ 。

这一次我们排序的方式是以 $n^{\frac{2}{3}}$ 为一块，分成了 $n^{\frac{1}{3}}$ 块，第一关键字是左端点所在块，第二关键字是右端点所在块，第三关键字是时间。

还是来证明一下时间复杂度（默认块大小为 $\sqrt{n}$ ）：

- 左右端点所在块不变，时间在排序后单调向右移，这样的复杂度是 $O(n)$ ；
- 若左右端点所在块改变，时间一次最多会移动 n 个格子，时间复杂度 $O(n)$ ；
- 左端点所在块一共有 $n^{\frac{1}{3}}$ 中，右端点也是 $n^{\frac{1}{3}}$ 种，一共 $n^{\frac{1}{3}} \times n^{\frac{1}{3}} = n^{\frac{2}{3}}$ 种，每种乘上移动的复杂度 $O(n)$ ，总复杂度 $O(n^{\frac{5}{3}})$ 。

例题

例题「国家集训队」数颜色 / 维护队列 [https://www.luogu.com.cn/problem/P1903]

题目大意：给你一个序列，M 个操作，有两种操作：

1. 修改序列上某一位的数字
2. 询问区间 $[l,r]$ 中数字的种类数（多个相同的数字只算一个）

我们不难发现，如果不带操作 1（修改）的话，我们就能轻松用普通莫队解决。

但是题目还带单点修改，所以用 **带修改的莫队** 。

先考虑普通莫队的做法：

- 每次扩大区间时，每加入一个数字，则统计它已经出现的次数，如果加入前这种数字出现次数为 0 ，则说明这是一种新的数字，答案 $+1$ 。然后这种数字的出现次数 $+1$ 。
- 每次减小区间时，每删除一个数字，则统计它删除后的出现次数，如果删除后这种数字出现次数为 0 ，则说明这种数字已经从当前的区间内删光了，也就是当前区间减少了一种颜色，答案 -1 。然后这种数字的出现次数 -1 。

现在再来考虑修改：

- 单点修改，把某一位的数字修改掉。假如我们是从一个经历修改次数为 i 的询问转移到一个经历修改次数为 j 的询问上，且 $i<j$ 的话，我们就需要把第 $i+1$ 个到第 j 个修改强行加上。
- 假如 $j<i$ 的话，则需要把第 i 个到第 $j+1$ 个修改强行还原。

怎么强行加上一个修改呢？假设一个修改是修改第 pos 个位置上的颜色，原本 pos 上的颜色为 a ，修改后颜色为 b ，还假设当前莫队的区间扩展到了 $[l, r]$ 。

- 加上这个修改：我们首先判断 pos 是否在区间 $[l, r]$ 内。如果是的话，我们等于是从区间中删掉颜色 a ，加上颜色 b ，并且当前颜色序列的第 pos 项的颜色改成 b 。如果不在区间 $[l, r]$ 内的话，我们就直接修改当前颜色序列的第 pos 项为 b 。
- 还原这个修改：等于加上一个修改第 pos 项、把颜色 b 改成颜色 a 的修改。

因此这道题就这样用带修改莫队轻松解决啦！

参考代码

```cpp
#include <bits/stdc++.h>
#define SZ (10005)
using namespace std;
template <typename _Tp>
inline void IN(_Tp& dig) {
  char c;
  dig = 0;
  while (c = getchar(), !isdigit(c))
    ;
  while (isdigit(c)) dig = dig * 10 + c - '0', c =
getchar();
}
int n, m, sqn, c[SZ], ct[SZ], c1, c2, mem[SZ][3], ans,
tot[1000005], nal[SZ];
struct query {
  int l, r, i, c;
  bool operator<(const query another) const {
    if (l / sqn == another.l / sqn) {
      if (r / sqn == another.r / sqn) return i <
another.i;
      return r < another.r;
    }
    return l < another.l;
  }
} Q[SZ];
void add(int a) {
  if (!tot[a]) ans++;
  tot[a]++;
}
void del(int a) {
  tot[a]--;
  if (!tot[a]) ans--;
```

```
}
char opt[10];
int main() {
  IN(n), IN(m), sqn = pow(n, (double)2 / (double)3);
  for (int i = 1; i <= n; i++) IN(c[i]), ct[i] = c[i];
  for (int i = 1, a, b; i <= m; i++)
    if (scanf("%s", opt), IN(a), IN(b), opt[0] == 'Q')
      Q[c1].l = a, Q[c1].r = b, Q[c1].i = c1, Q[c1].c =
c2, c1++;
    else
      mem[c2][0] = a, mem[c2][1] = ct[a], mem[c2][2] =
ct[a] = b, c2++;
  sort(Q, Q + c1), add(c[1]);
  int l = 1, r = 1, lst = 0;
  for (int i = 0; i < c1; i++) {
    for (; lst < Q[i].c; lst++) {
      if (l <= mem[lst][0] && mem[lst][0] <= r)
        del(mem[lst][1]), add(mem[lst][2]);
      c[mem[lst][0]] = mem[lst][2];
    }
    for (; lst > Q[i].c; lst--) {
      if (l <= mem[lst - 1][0] && mem[lst - 1][0] <= r)
        del(mem[lst - 1][2]), add(mem[lst - 1][1]);
      c[mem[lst - 1][0]] = mem[lst - 1][1];
    }
    for (++r; r <= Q[i].r; r++) add(c[r]);
    for (--r; r > Q[i].r; r--) del(c[r]);
    for (--l; l >= Q[i].l; l--) add(c[l]);
    for (++l; l < Q[i].l; l++) del(c[l]);
    nal[Q[i].i] = ans;
  }
  for (int i = 0; i < c1; i++) printf("%d\n", nal[i]);
  return 0;
}
```

本页面最近更新：2020/7/17 19:50:44，更新历史 [https://github.com/OI-wiki/OI-wiki/commits/master/docs/misc/modifiable-mo-algo.md]
发现错误？想一起完善？ 在 GitHub 上编辑此页！ [https://oi-wiki.org/edit-landing/?ref=/misc/modifiable-mo-algo.md]
本页面贡献者：countercurrent-time [https://github.com/countercurrent-time], Ir1d [https://github.com/Ir1d], StudyingFather [https://github.com/StudyingFather], Backl1ght [https://github.com/Backl1ght], greyqz [https://github.com/greyqz], MicDZ