

# 带修莫队

- Edit by **Withinlover**

本文建立在已经掌握并熟知普通莫队的基础上，如果你对普通莫队的使用仍有疑惑，请前往阅读普通莫队。

## 算法介绍

普通莫队的一大局限在于不支持修改操作。

我们可以通过增加时间轴的方式让莫队强行支持修改。

具体的，就是由  $(l, r)$  变为  $(l, r, t)$

若已知  $(l, r, t)$  我们考虑如下 6 种情况

- $(l - 1, r, t)$
- $(l + 1, r, t)$
- $(l, r - 1, t)$
- $(l, r + 1, t)$
- $(l, r, t - 1)$
- $(l, r, t + 1)$

若这些状态均可以由  $(l, r, t)$  在  $O(1)$  或其他极短的时间内完成。那么便可以考虑带修莫队。假设  $n$  和  $m$  同阶，其复杂度为  $O(\log(n^{\frac{5}{3}}))$

## 写法对比

### 普通莫队

```
for(int i = 1; i <= m; ++i) {
    while(r < Q[i].r) ans = ans + Calc(++r, 1);
    while(r > Q[i].r) ans = ans + Calc(r--, -1);
    while(l < Q[i].l) ans = ans + Calc(l++, -1);
    while(l > Q[i].l) ans = ans + Calc(--l, 1);
    Ans[Q[i].pos] = ans;
}
```

### 带修莫队

```
for(int i = 1; i <= m; ++i) {
    while(r < Q[i].r) Add(a[++r]);
    while(r > Q[i].r) Del(a[r--]);
    while(l < Q[i].l) Del(a[l++]);
    while(l > Q[i].l) Add(a[--l]);
    while(t < Q[i].t) Make(++t);
    while(t > Q[i].t) Make(t--);
    Ans[Q[i].pos] = ans;
}
```

基本上没什么差距，会普通的约等于会带修改的。

## 复杂度证明

假设共有  $N$  个点  $M$  次操作  $A$  次查询  $B$  次修改，块大小为  $S$

对于  $l$  指针：

- 块内移动，每次查询的最大代价为  $S$  共有  $A$  次，复杂度  $SA$

对于  $r$  指针

- 随  $l$  指针移动  $l$  固定在一块中时  $r$  指针单调递增，最大代价为  $N$  共有  $\frac{N}{S}$  次，复杂度  $\frac{N^2}{S}$

对于  $t$  指针

- 当  $l, r$  固定时  $t$  单调递增，最大代价为  $B$  次，共有  $\frac{N^2}{S^2}$  次，复杂度  $\frac{BN^2}{S^2}$

推一推，由于题目中不会告诉你查询和修改的次数，所以  $A$  和  $B$  均视为  $M$   
 $O\left(SM + \frac{N^2}{S} + \frac{MN^2}{S^2}\right)$

这玩意的极值不好确定，取得精确结果比较困难。一般而言，在题目中  $N$  和  $M$  是同阶的，设  $S = N^x$  则可得复杂度为  $O(N^{x+1} + N^{2-x} + N^{3-2x})$  我们希望其指数部分尽可能的小。即  $\max\{x+1, 2-x, 3-2x\}$  最小。解得  $x = \frac{2}{3}$  即取  $S = N^{\frac{2}{3}}$  此时的复杂度为  $O(N^{\frac{5}{3}})$

## 例题

咕咕咕

From:  
<https://wiki.cvbbacm.com/> - CVBB ACM Team

Permanent link:  
[https://wiki.cvbbacm.com/doku.php?id=2020-2021:teams:mian:withinlover:mos\\_algorithm&rev=1589261582](https://wiki.cvbbacm.com/doku.php?id=2020-2021:teams:mian:withinlover:mos_algorithm&rev=1589261582)

Last update: 2020/05/12 13:33

