区间覆盖最值问题

题目

给定数列范围[1,n],权值为 a_i .有m次操作,每次操作能将给定区间 $[l_i,r_i]$ 的权值变成 v_i . 求m次操作后数列总权值的最大(最小)值

做法

法一:

显然是个区间覆盖问题. 所以我们可以通过差分 + 最大(最小)堆,然后统计答案的时候从左到右扫一遍. 法二:

由于要求极值,那不难知道,对于权值 v_i 较大的区间,较小的 v_i 是不可能覆盖掉它的,否则答案不是最值. 或者我们将操作的权值从大到小(从小到大)排序一遍,由于每次操作的区间权值 v_i 都是当前最大值, 所以后面的操作无法覆盖它,所以我们可以缩点一般,把这个区间合成一个点

因此我们可以通过并查集来维护这个缩点.

每次操作都暴力遍历操作区间 $[l_i, r_i]$,将当前区间合并.

Code

法一:

```
1 #include <bits/stdc++.h>
 2 #define int long long
 3 #define endl '\n'
 4 #define LL int128
    using namespace std;
   int qpow(int a, int b, int p) {int ret = 1; for(a \%= p; b; b >>= 1, a = a *
    a % p) if(b & 1) ret = ret * a % p; return ret; }
    int qpow(int a, int b) {int ret = 1; for(; b; b >>= 1, a *= a) if(b & 1) ret
    *= a; return ret; }
    int gcd(int x,int y) {return y ? gcd(y, x % y) : x; }
    pair<int,int> exgcd(int a,int b) { if(!b) return {1, 0}; pair<int,int> ret =
    exgcd(b, a % b); return {ret.second, ret.first - a / b * ret.second }; }
   int lcm(int x,int y){ return x / gcd(x, y) * y; }
10
11 | const int N = 5 + 1e5;
12
    map<char, int> add[N], del[N];
13
    signed main() {
14
15
        int n, m; cin >> n >> m;
        string s; cin \gg s; s = ' ' + s;
16
17
        while (m--) {
            int 1, r; char c; cin >> 1 >> r >> c;
18
19
            add[1][c]++; del[r + 1][c]++;
20
        }
21
        map<char, int> mp;
22
        for (int i = 1; i <= n; i++) {
            for (auto &v: add[i]) {
23
                mp[v.first] += v.second;
24
```

```
25
26
             for (auto &v: del[i]) {
27
                 mp[v.first] -= v.second;
28
                 if (!mp[v.first]) mp.erase(v.first);
29
30
             if (mp.size()) {
31
                 auto u = mp.end(); u--;
32
                 if ((*u).second) s[i] = max(s[i], (*u).first);
33
             }
34
        }
35
        int ans = 0;
36
        for (int i = 1; i \le n; i++) ans += s[i];
37
        cout << ans << endl;</pre>
38
   }
```

法二:

```
1 #include <bits/stdc++.h>
 2
    #define int long long
   #define endl '\n'
 3
    #define LL __int128
    using namespace std;
 5
    int qpow(int a, int b, int p) {int ret = 1; for(a \%= p; b; b >>= 1, a = a *
    a % p) if(b & 1) ret = ret * a % p; return ret; }
 7
    int qpow(int a, int b) {int ret = 1; for(; b; b >>= 1, a *= a) if(b & 1) ret
    *= a; return ret; }
    int gcd(int x,int y) {return y ? gcd(y, x % y) : x; }
    pair<int,int> exgcd(int a,int b) { if(!b) return {1, 0}; pair<int,int> ret =
    exgcd(b, a % b); return {ret.second, ret.first - a / b * ret.second }; }
10
    int lcm(int x,int y){ return x / gcd(x, y) * y; }
11
    struct T {
12
        int 1, r; char c;
13
        friend bool operator < (T x, T y) {
14
            return x.c < y.c;
15
        }
16
    };
    const int N = 5 + 1e7;
17
18
    int fa[N];
19
    int find(int x) { return x == fa[x] ? x : fa[x] = find(fa[x]); }
20
    signed main() {
21
        ios :: sync_with_stdio(false), cin.tie(0), cout.tie(0);
22
        int n, m; cin >> n >> m;
        string s; cin \gg s; s = ' ' + s;
23
24
        priority_queue<T> q;
25
        for (int i = 0; i < m; i++) {
26
            int 1, r; char c; cin >> 1 >> r >> c;
27
            q.push({1, r, c});
28
29
        for (int i = 1; i \le n + 1; i++) fa[i] = i;
30
        int ans = 0;
31
        while (q.size()) {
32
            auto u = q.top(); q.pop();
33
            for (int i = u.1; i \le u.r; i = fa[i + 1]) {
34
                fa[i] = find(i + 1);
35
                if (s[i] < u.c) s[i] = u.c;
36
            }
37
        }
```

```
for (int i = 1; i <= n; i++) ans += s[i];
cout << ans << endl;
}</pre>
```

```
1
    #include <bits/stdc++.h>
    #define int long long
 2
 3
    #define endl '\n'
    #define LL __int128
 4
 5
    using namespace std;
    int qpow(int a, int b, int p) {int ret = 1; for(a \%= p; b; b >>= 1, a = a *
 6
    a % p) if(b & 1) ret = ret * a % p; return ret; }
    int qpow(int a, int b) {int ret = 1; for(; b; b >>= 1, a *= a) if(b & 1) ret
 7
    *= a; return ret; }
 8
    int gcd(int x,int y) {return y ? gcd(y, x % y) : x; }
 9
    pair<int,int> exgcd(int a,int b) { if(!b) return {1, 0}; pair<int,int> ret =
    exgcd(b, a % b); return {ret.second, ret.first - a / b * ret.second }; }
    int lcm(int x,int y){ return x / gcd(x, y) * y; }
10
11
    struct T {
12
        int 1, r; char c;
13
        friend bool operator < (T x, T y) {
14
            return x.c < y.c;
        }
15
16
    };
17
    const int N = 5 + 1e7;
    int fa[N];
18
19
    int find(int x) { return x == fa[x] ? x : fa[x] = find(fa[x]); }
20
    signed main() {
21
        ios :: sync_with_stdio(false), cin.tie(0), cout.tie(0);
22
        int n, m; cin >> n >> m;
23
        string s; cin \gg s; s = ' ' + s;
        priority_queue<T> q;
24
25
        for (int i = 0; i < m; i++) {
26
            int 1, r; char c; cin >> 1 >> r >> c;
27
            q.push({1, r, c});
28
        }
29
        for (int i = 1; i \le n; i++) fa[i] = i;
30
        int ans = 0;
        int pre = 0;
31
32
        while (q.size()) {
33
            auto u = q.top(); q.pop();
            //cout << u.l << "," << u.r << "," << u.c << endl;
34
35
            stack<pair<int, char>> st;
36
            for (int i = u.1, pre = 0; i \le u.r; pre = i, i++) {
37
                if (find(i) == i) {
38
                     st.push({i, u.c});
39
                } else {
                     while (st.size()) {
40
41
                         if (!pre) pre = st.top().first;
42
                         fa[st.top().first] = find(pre);
43
                         s[st.top().first] = max(s[st.top().first],
    st.top().second);
44
                         st.pop();
45
                     }
46
                     i = fa[i];
                }
47
48
            }
            while (st.size()) {
49
```

```
fa[st.top().first] = find(u.r);
s[st.top().first] = max(s[st.top().first], st.top().second);
st.pop();
st.pop();
}
for (int i = 1; i <= n; i++) ans += s[i];
cout << ans << end];
}</pre>
```