omarsebres

Mainly Programming blog for experience sharing.

Posts Tagged '2D Segment Tree'

11297 – Census

Posted by OmarSaber on December 6, 2012

Problem Statement (http://uva.onlinejudge.org/external/112/11297.html)

Use a segment tree for each row.

```
1  using namespace std;
2
3  #define L(x) (1 + ((x) << 1))
4  #define R(x) (2 + ((x) << 1))
5  typedef pair<int, int> state;
6  int const MAXN = 505;
7
8  struct io {
9      template<class T> inline static T next() {
10         T n; std::cin >> n;
11         return n;
12     }
13 };
14
15 struct node {
16     node() : max(INT_MIN), min(INT_MAX) {}
17     int begin, end, mid, min, max;
18 };
19
20 template<int size> struct segment_tree {
21     node tree[size << 2];
```

```
22
23      void init_tree(int index, int left, int right, int values[]) {
24          node& now = tree[index];
25          now.begin = left, now.end = right;
26          now.mid = (now.begin + now.end) >> 1;
27          if (now.begin != now.end) {
28              init_tree(L(index), now.begin, now.mid, values);
29              init_tree(R(index), now.mid + 1, now.end, values);
30              now.max = max(now.max, max(tree[L(index)].max, tree[R(ind
31              now.min = min(now.min, min(tree[L(index)].min, tree[R(ind
32          } else
33              now.min = now.max = values[now.begin];
34      }
35
36      state update(int index, int at, int value, int values[]) {
37          node& now = tree[index];
38          if (now.begin == at && now.end == at) {
39              now.max = now.min = value;
40              values[at] = value;
41              return state(value, value);
42          }
43          if (now.begin > at || now.end < at)
44              return state(now.max, now.min);
45          state s1 = update(L(index), at, value, values);
46          state s2 = update(R(index), at, value, values);
47          now.max = max(s1.first, s2.first);
48          now.min = min(s1.second, s2.second);
49          return state(now.max, now.min);
50      }
51
52      state query(int index, int left, int right) {
53          node& now = tree[index];
54          if (now.begin >= left && now.end <= right)
55              return state(now.max, now.min);
56          if (now.begin > right || now.end < left)
57              return state(INT_MIN, INT_MAX);
58          state result(INT_MIN, INT_MAX);
59          state s1 = query(L(index), left, right);
60          state s2 = query(R(index), left, right);
61          result.first = max(result.first, max(s1.first, s2.first));
62          result.second = min(result.second, min(s1.second, s2.second))
63          return result;
64      }
65  };
66
67  segment_tree<MAXN> tree[MAXN];
68  int values[MAXN][MAXN];
69
70  int main(int argc, char **args) {
71      for (int n; scanf("%d%d", &n, &n) != EOF;) {
72          for (int i = 0; i < n; ++i) {
73              for (int j = 0; j < n; ++j)
74                  values[i][j] = io::next<int>();
75              tree[i].init_tree(0, 0, n - 1, values[i]);
```

```
76          }
77          int query = io::next<int>();
78          while (query-- > 0) {
79              char cmd = io::next<char>();
80              int x1 = io::next<int>() - 1;
81              int y1 = io::next<int>() - 1;
82              if (cmd == 'q') {
83                  int x2 = io::next<int>() - 1;
84                  int y2 = io::next<int>() - 1;
85                  if (x1 > x2) swap(x1, x2);
86                  if (y1 > y2) swap(y1, y2);
87                  state result(INT_MIN, INT_MAX);
88                  for (int i = x1; i <= x2; ++i) {
89                      state now = tree[i].query(0, y1, y2);
90                      result.first = max(result.first, now.first);
91                      result.second = min(result.second, now.second);
92                  }
93                  printf("%d %d\n", result.first, result.second);
94              } else
95                  tree[x1].update(0, y1, io::next<int>(), values[x1]);
96          }
97      }
98      return 0;
99  }
```

Posted in Chapter 2 | Tagged: 2D Segment Tree | Leave a Comment »