Куприянов Михаил, ПЗПИ-19-2

Ревью 3

B. Ваня и матрица

Алгоритм решения

Создадим двумерное дерево отрезков с сохранением максимумов – дерево отрезков, в каждой вершине которого хранится по одномерному дереву отрезков для соответствующей группы строк (если i-я вершина двумерного дерева отрезков отвечает за индексы с A по B – в вершине j, отвечающей за элементы с C по D одномерного дерева отрезков хранящегося в вершине i, хранится максимум в прямоугольнике с A по B по строкам и с C по D по столбцам).

Построение двумерного дерева отрезков: построим по одномерному дереву отрезков максимумов для каждой строки, затем для отца 2*i и 2*i+1 вершин каждый элемент одномерного дерева a[i][j]=max(a[2*i][j], a[2*i+1][j]) (действительно, максимум в прямоугольнике равен максимуму в двух составляющих его меньших прямоугольниках).

Запрос модификации: изменяем элемент, соответствующий индексу (x,y), затем обновляем все деревья отрезков накрывающие x, а в каждом из них все элементы накрывающие y (таким образом все элементы двумерного дерева, имеющие отношение к (x,y) будут содержать актуальную информацию). Заметим, кроме того, что количество деревьев отрезков, накрывающих x – O(log n) (хотя бы потому что глубина дерева отрезков на n вершин – O(log n)), а количество вершин в каждой из них, накрывающих y – O(log(m)) (по аналогичным причинам).

Запрос максимума для (x1, x2, y1,y2): обрабатываем запрос максимума на (x1,x2) как для одномерного дерева отрезков, однако из каждой вершины вызываем максимум на отрезке (y1,y2). Действительно, с помощью первой части мы разбиваем отрезок (x1,x2) на O(log n) отрезков строк, для которых у нас уже построены одномерные деревья отрезков, а с помощью второй в каждом из них ищем максимум на (y1,y2).

Временная сложность

Мы считываем все числа в матрицу за O(n*m), затем строим по ней дерево отрезков за O(n*m), затем выполняем q запросов. Запрос модификации, как показано выше, требует O(log n * log m) времени. Запрос максимума требует от нас O(log n) операций размера O(log m) (по тем же причинам, по которым для обычного дерева отрезков на n вершин запрос максимума требует O(log n) времени – на каждом уровне дерева отрезков мы обрабатываем не более 4 отрезков). Итого: O(n*m)+a*O(log n * log m) + (q-a)*O(log n)*O(log m)=**O(n*m+q*log n * log m)**

Затраты памяти

В нашей реализации дерева отрезков используется ровно 2*n ячеек памяти для дерева отрезков на n элементов. Таким образом, у нас есть 2*n деревьев отрезков в каждом из которых по 2*m элементов, итого 4*n*m. Так как информация о прошлых запросах нигде не хранится, итоговая сложность – **O(n*m)**