УДК 519.67

Задача об охране картинной галереи на клетчатой плоскости

Вылегжанин Д.В., Гринкевич А.В., Оскорбин Д.Н.
Белорусский государственный университет, г. Минск (Белоруссия)
Алтайский государственный университет, г. Барнаул
vylegzhanin@bsu.by, alexander.grin97@gmail.com, oskorbin@yandex.ru

Аннотация

В данной работе рассматривается задача об охране картинной галереи в случае, когда план галереи представляет собой ортогональный многоугольник с вершинами в узлах целочисленной решетки. Проводится точная оценка на число охранников, а также разрабатывается жадный алгоритм расстановки охранников. Для реализации алгоритма выбран язык программирования Python.

Ключевые слова: вычислительная геометрия, ортогональный многоугольник, задача об охране картинной галереи.

На сегодняшний день задача об охране картинной галереи является одной из хорошо изученных задач в области вычислительной геометрии [1], она возникает в реальном мире как задача охраны художественной галереи минимальным числом охранников, которые могут наблюдать за всей галереей. В вычислительной геометрии галерея представлена в виде простого многоугольника, а охранник — точкой внутри него.

Существует теорема о картинной галерее, которая принадлежит Вацлаву Хваталу, которая утверждает, что для охраны многоугольника из n вершин всегда достаточно $\left[\frac{n}{3}\right]$ охранников [2]. Позднее той же оценки добился Стив Фиск, значительно упростив доказательство Хватала. Для этого он использовал раскраску вершин многоугольника в три цвета [3].

Теорема 1. *Для охраны многоугольника из n вершин всегда достаточно, а иногда и необходимо $\left[\frac{n}{3}\right]$ охранников.*

Схема доказательства. Триангулируем многоугольник S без добавления вершин. У каждой такой триангуляции вершины могут быть раскрашены в три цвета a, b, c. Пусть T_k набор вершин, окрашенных в цвет a и предположим, что $|T_a| \leq |T_b| \leq |T_c|$. Выбирая $T = T_a$, подразумеваем, что $|T| \leq \left[\frac{n}{3}\right]$. Таким образом, каждая точка q из S лежит в некотором треугольнике из S, и каждый треугольник из S имеет точку $p \in T$. Поскольку треугольники выпуклые, мы имеем $pq \in S$.

Оценка $\left[\frac{n}{3}\right]$ является точной, т.е. для охраны многоугольника из n вершин иногда необходимо ровно столько охранников (рисунок 1).

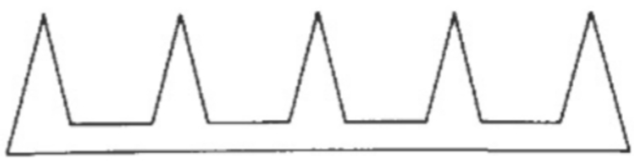
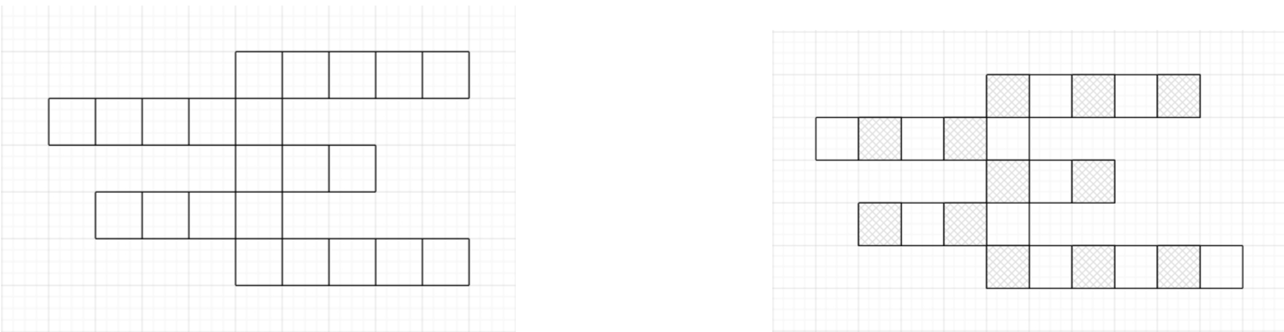


Рисунок 1. Многоугольники, для охраны которых требуется ровно $\left[\frac{n}{3}\right]$ охранников

В данной работе изучается версия задачи об охране картинной галереи, когда она представлена многоугольником с вершинами в узлах целочисленной решетки, т.е. соседние стороны пересекаются под прямыми углами. Ограничимся случаем, когда вершины многоугольника находятся в узлах целочисленной решетки на плоскости, а стороны параллельны линиям сетки, т.е. план картинной галереи — клетчатая фигура, где каждая клетка — это зал, и из любой клетки можно дойти до любой другой, переходя в соседние по сторонам клетки. Смотритель, находясь в одном из залов, следит за всеми залами, в которые можно попасть из этой клетки одним ходом ладьи (не выходя за пределы галереи). Какое наименьшее число смотрителей потребуется, чтобы в любой галерее из n залов $(n > 1)$ все залы оказались под присмотром?

Теорема 2. *Для охраны описанного выше многоугольника из n залов* $(n > 1)$ *всегда достаточно, а иногда необходимо* $\left[\frac{n}{2}\right]$ *охранников.*

Доказательство. 1. Достаточность. Представим план галереи следующим образом. Поместим данную фигуру на плоскость, раскрашенную в шахматном порядке (рисунок 2).



a) b)

Рисунок 2. a) План галереи; b) план галереи на шахматной доске

Пусть n – число залов. Найдем количество белых и черных клеток, попадающих в план галереи. Число охранников для данной галереи определяется как $\min\{m, k\}$, где m – черные клетки, k – белые. Т.е. $m = \left[\frac{n}{2}\right]$.

2. Необходимость. План галереи представлен в виде клетчатой фигуры (рисунок 3).

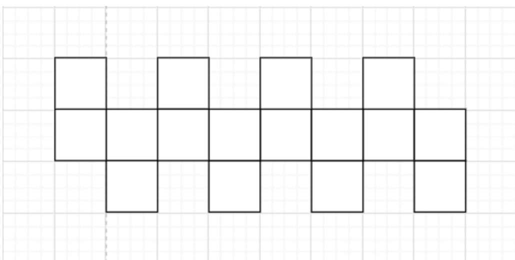


Рисунок 3. План галереи

Пусть n – число залов. По условию задач охранник может наблюдать только за теми залами, в которые может попасть одним ходом ладьи, при этом не выходя за план галереи. Таким образом, для наблюдения за галереей требуется $\left[\frac{n}{2}\right]$ охранников. □

Для расстановки охранников воспользуемся "жадным" алгоритмом.

На вход подается план галереи, который хранится в виде матрицы размера $m \times n$, состоящей из элементов 0 и 1, где 1 – зал, 0 – пусто (пример 1).

Пример 1.

$$\begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 1 & 0 \end{pmatrix}$$

На выходе имеем расстановку охранников и залы, которые находятся под их наблюдением, которая также хранится в виде матрицы $m \times n$. В матрице позиция охранника обозначена как *10× (номер строки) + (номер столбца)*, а залы, находящиеся под его присмотром – 5 (пример 2).

Пример 2.

$$\begin{pmatrix} 5 & 12 & 0 \\ 0 & 5 & 23 \\ 31 & 5 & 0 \end{pmatrix}$$

Далее приводим описание алгоритма вышеупомянутого алгоритма.

I. Необходимо найти максимально длинный коридор, т.е. цепочку из единиц максимальной длины по строке и для нее отыскать столбец с максимальной по длине цепочкой из единиц. За это отвечают функции *indexOfMaxRow(a)* и *indexOfMaxColumn(a, row)*.

indexOfMaxRow(a)

1. Создать переменные row = -1 и $length$ = -1, которые отвечают за номер строки с максимально длинной цепочкой из единиц и длину цепочки соответственно.
2. for каждого i to n
do
3. Создать переменную-счетчик $l = 0$
4. for каждого элемента строки
do
5. if элемент строки равен 1
6. then $l + 1$
7. else цепочка обрывается и $l = 0$
8. if $l > length$
9. then сохранить найденную, более длинную цепочку из единиц.

indexOfMaxColumn(a, row)

1. Создать переменные $column$ = -1 и $length$ = -1, которые хранят номер столбца и максимальную длину цепочки из единиц соответственно.
2. for каждого c to m
do
3. if столбец не входит в выбранную строку
4. then $l = 0$
5. for каждого r to n
do
6. if очередной элемент равен 1

7. then $l + 1$
8. else цепочка обрывается и $l = 0$
9. if длина цепочки больше чем начальная
10. then сохранить более длинную цепочку.

II. Найти элемент на пересечении найденных строки и столбца и заменить единицу на *10* $\times$ *(номер строки)* + *(номер столбца)* для выделения позиции охранника в галерее.

1. Создать переменную $replace = 5$, значение которой будет присваиваться элементам (залам), которые находятся в ходе ладьи от позиции охранника.

   Далее опишем процесс замены единиц на значение переменной *replace*. Известно, что охранник находится в зале $[row, col]$. Осмотр охраняемых залов начинается с ближайшего, так $r = row - 1$. Взор охранника будет направлен в сторону уменьшения строк до тех пор, пока на пути не встретится стена, т.е., пока $a[r][col] = 1$. Т.к. по условию задачи обзор галереи охранником ограничен ходом ладьи, то для оставшихся трех направлений записываются аналогичные конструкции.
2. $r = row - 1$
3. while $(r >= 0)$ и элемент $(a[r][col] == 1)$
   do
4. Заменить $a[r][col]$ значением переменной *replace*.
5. $r - 1$
6. $r = row + 1$
7. while $(r < n)$ и элемент $(a[r][col] == 1)$
   do
8. Заменить $a[r][col]$ значением переменной *replace*
9. $r + 1$
10. $c = col - 1$
11. while $(c >= 0)$ и элемент $(a[row][c] == 1)$
    do
12. Заменить $a[row][c]$ значением переменной *replace*
13. $c - 1$
14. $c = col + 1$
15. while $(c < m)$ и элемент $(a[row][c] == 1)$
    do
16. Заменить $a[row][c]$ значением переменной *replace*
17. $c + 1$

III. Данный алгоритм выполняется до тех пор, пока в галерее есть залы, которые не находятся под присмотром охранника (пока в матрице плана галереи есть единицы).

Проанализируем время работы данного алгоритма. Каждый входной элемент подвергается обработке. При увеличении размерности матрицы увеличивается общее число элементов, что сказывается на времени выполнения программы. Обработка матрицы размерности $m \times n$ занимает линейное время $O(n \times m)$.

Пример работы алгоритма приведены на рисунке 4.

```
Введите количество строк: 5
Введите количество столбцов: 6
a[0,0]=0
a[0,1]=0
a[0,2]=0
a[0,3]=1
a[0,4]=1
a[0,5]=1
a[1,0]=0
a[1,1]=0
a[1,2]=1
a[1,3]=1
a[1,4]=0
a[1,5]=0
a[2,0]=0
a[2,1]=1
a[2,2]=1
a[2,3]=0
a[2,4]=0
a[2,5]=0
a[3,0]=1
a[3,1]=1
a[3,2]=0
a[3,3]=0
a[3,4]=0
a[3,5]=0
a[4,0]=1
a[4,1]=1
a[4,2]=0
a[4,3]=0
a[4,4]=0
a[4,5]=0
```

a)

```
Карта
  0  0  0  1  1  1
  0  0  1  1  0  0
  0  1  1  0  0  0
  1  1  0  0  0  0
  1  1  0  0  0  0
Количество залов: 11
строчка 1 (длина 3) столбец 4 (длина 2)
Карта
  0  0  0 14  5  5
  0  0  1  5  0  0
  0  1  1  0  0  0
  1  1  0  0  0  0
  1  1  0  0  0  0
строчка 3 (длина 2) столбец 2 (длина 3)
Карта
  0  0  0 14  5  5
  0  0  1  5  0  0
  0 32  5  0  0  0
  1  5  0  0  0  0
  1  5  0  0  0  0
строчка 2 (длина 1) столбец 3 (длина 1)
Карта
  0  0  0 14  5  5
  0  0 23  5  0  0
  0 32  5  0  0  0
  1  5  0  0  0  0
  1  5  0  0  0  0
строчка 4 (длина 1) столбец 1 (длина 2)
Карта
  0  0  0 14  5  5
  0  0 23  5  0  0
  0 32  5  0  0  0
 41  5  0  0  0  0
  5  5  0  0  0  0
Готово
Карта
  0  0  0 14  5  5
  0  0 23  5  0  0
  0 32  5  0  0  0
 41  5  0  0  0  0
  5  5  0  0  0  0
```

b)

Рисунок 4. a) Ввод данных; b) результат работы алгоритма

Список литературы

1. O'Rourke J. Art Gallery Theorems and Algorithms. — New York : Oxford University Press, 1987. — 282 p.

2. Chvatal V. A combinatorial theorem in plane geometry // Journal of Combinatorial Theory, Series B. — 1975. — Vol. 18. — P. 39–41.

3. Fisk S. A short proof of Chvatal's watchman theorem // Journal of Combinatorial Theory, Series B. — 1978. — Vol. 24, no. 3. — P. 374.

4. Берг М., Чеонг О., Кревельд М., Овермарс М. Вычислительная геометрия. Алгоритмы и приложения. — М. : ДМК Пресс, 2017. — 438 с.