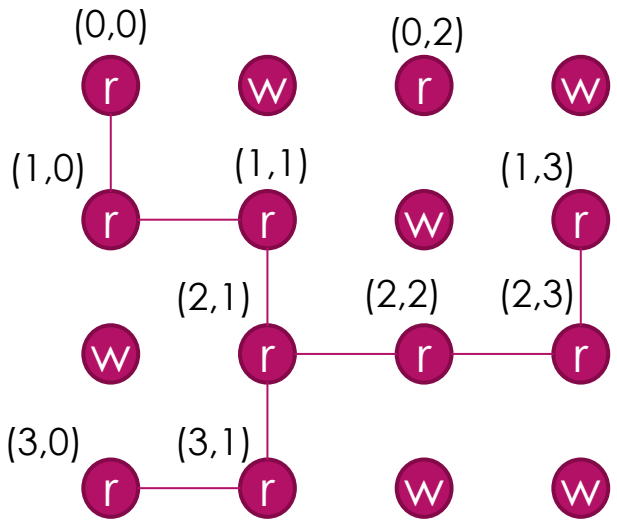
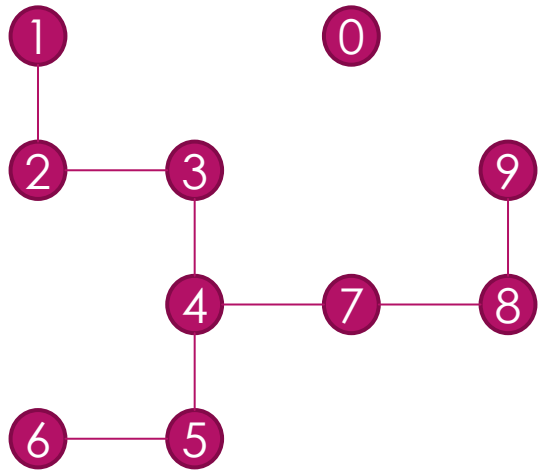
Lab2问题分析

赵耀

图可以表示为矩阵，矩阵表示图？





当用矩阵表示图

- 每个节点标识为矩阵的坐标(x,y)
- 节点与节点是否有边，取决于问题的描述，比如本题，如果两个点的坐标满足|x1-x2|=1且|y1-y2|=0或者|x1-x2|=0且|y1-y2|=1时，点(x1,y1)与(x2,y2)是连通的。
- r或w可以认为是节点的属性，节点属性如果为w，可认为任何点到w的距离都为 ∞ ，w到任何其他点的距离也为∞。
- 属性为w也可不纳入图的结点范围，r的结点如果周围有w，可认为此方向没有边，BFS不到达。

BFS求源到目标的算法流程

```
int BFS(Node T, Node goal){
    queue.init();
    queue.add(T);
    T.setDist(0);
    while(!queue.empty()){
        e = queue.remove();
        setVisited(e);
        while(e.hasNextChild){
            child = e.getNextChild();
            if (child.equals(goal)){
                return (e.getDist()+1);
            }
            if (!child.isVisited()&&!queue.exist(child)){
                    child.setDist(e.getDist()+1);
                    queue.add(child);
            }
        }//end while
    }//end while
    return -1;
} //end BFS
```

- 矩阵图中节点的表示：

Node: x , y , dist

- 在矩阵图中搜索孩子节点：

1、|u.x-v.x|=1且|u.y-v.y|=0或者|u.x-v.x|=0且|u.y-v.y|=1

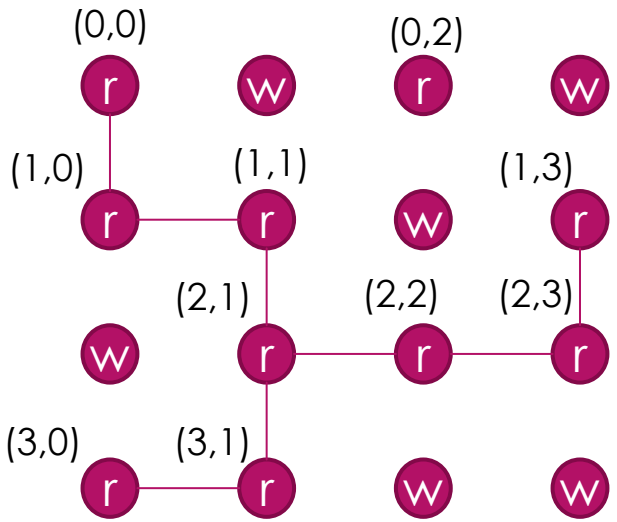
2、x,y不得超过矩阵范围n*m

3、x,y中字符不为'W'

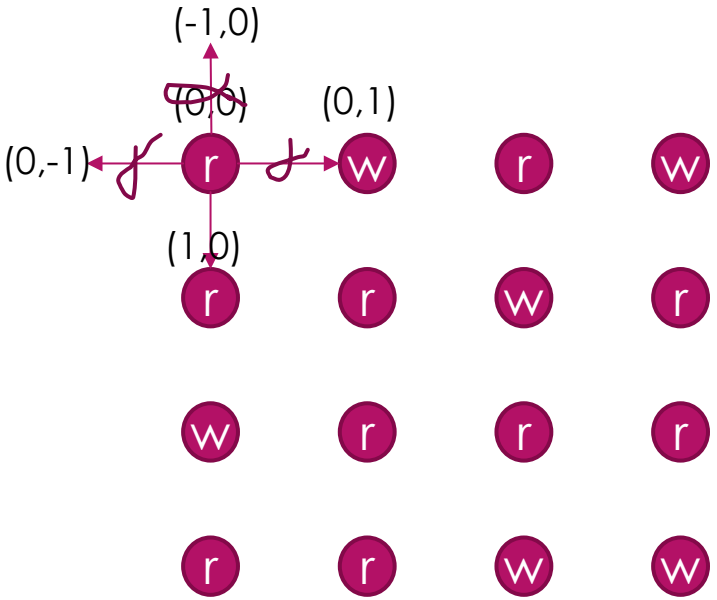
- 设置访问标志时，由于节点为矩阵的元素，用二维数组表示访问状态比较高效

输入

```
4 4
rwrw
rrwr
wrrr
rrww
2
3 0
2 2
```



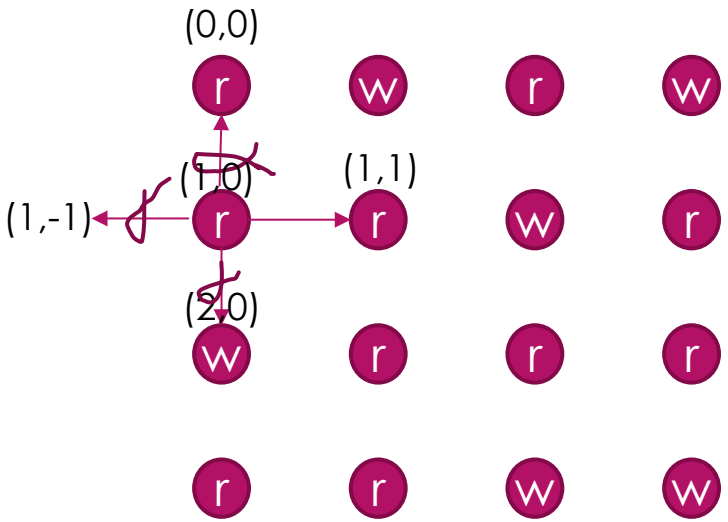
从(0,0)出发，step1



Reject:
(-1,0) (0,-1)越界
(0,1)为'w'

Visited:
(0,0){0}
Queue：
(1,0) {1}

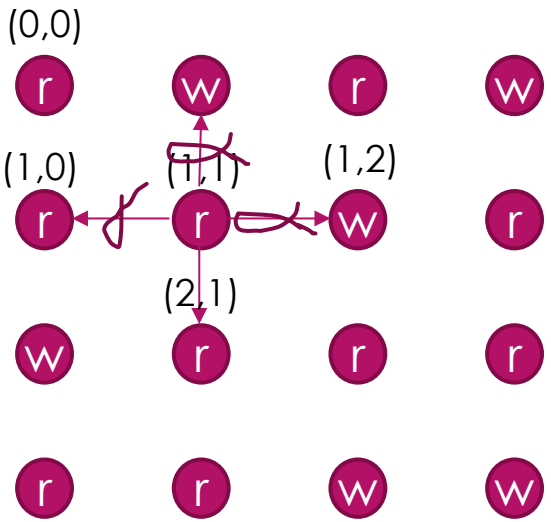
step2



Reject:
(1,-1)越界
(2,0)为'w'
(0,0)访问过

Visited:
(0,0) {0} (1,0) {1}
Queue：
(1,1) {2}

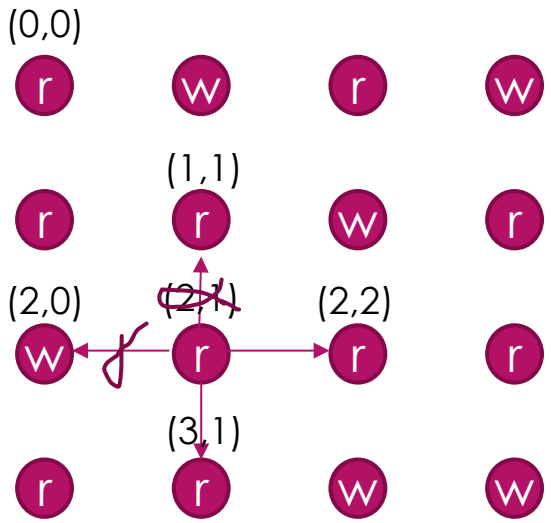
step3



Reject:
(0,1)(1,2)为'w'
(1,0)访问过

Visited:
(0,0) {0} (1,0) {1}(1,1) {2}
Queue：
(2,1) {3}

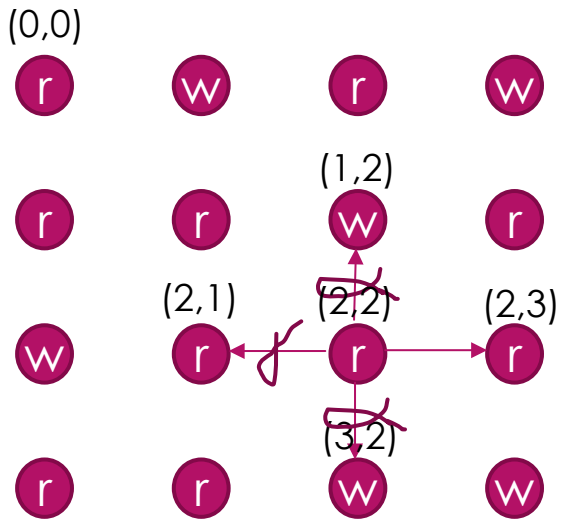
step4



Reject:
(2,0)为'w'
(1,1)访问过

Visited:
(0,0) {0} (1,0) {1}(1,1) {2}(2,1) {3}
Queue：
(2,2) {4}(3,1) {4}

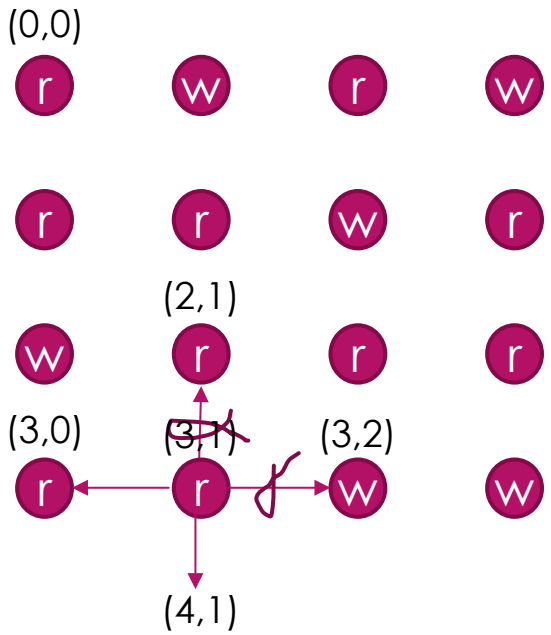
step5



Reject:
(2,2)(3,2)为'w'
(2,1)访问过

Visited:
(0,0) {0} (1,0) {1}(1,1) {2}(2,1) {3} (2,2) {4}
Queue：
(3,1) {4}(2,3) {5}

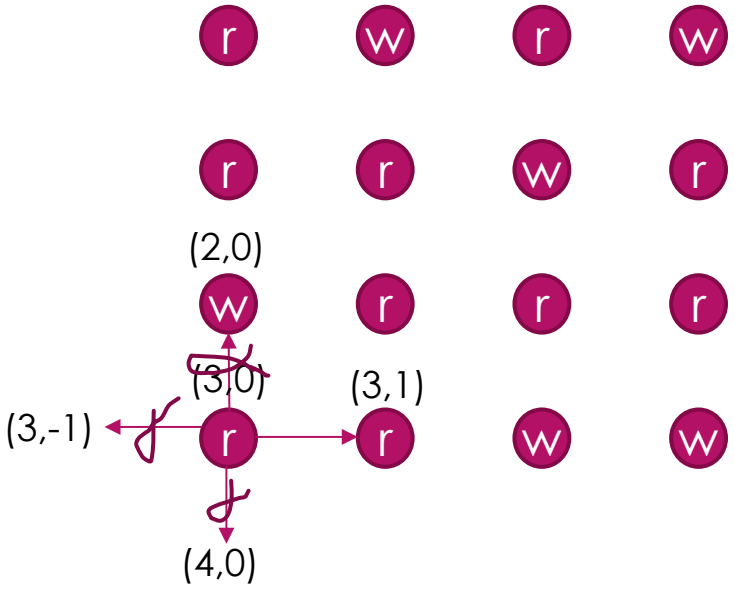
找到第一个目标点



Reject:
(3,2)为'w'
(2,1)访问过
(4,1)越界

Visited:
(0,0) {0} (1,0) {1}(1,1) {2}(2,1) {3} (2,2) {4}(3,1) {4}
Queue：
(2,3) {5}
Find Goal:
(3,0) {5}

一下轮查找，清除状态



Reject:
(4,0) (3,-1)越界
(2,1)为'w'

Visited:
(3,0){0}
Queue：
(3,1) {1}

代码容易出现的问题

- BFS搜索时忘记做边界检查或边界检查不正确。
- 由于有K个目标点，所以找完最初的最短距离之后，如果复用了中间变量，比如访问状态，距离累加变量忘记重新初始化。
- 第一个点为起始点，第二点为目标点，但是当第一个最短距离求出后，忘记将第二个点设置为第三个点的起始点，依次类推。
- 如果中间有一个点到不了下一个目标点，此时应该放弃后续的查找，返回-1。
- 有的同学用的自定义queue或heap，不需要的，直接用库中的Queue和Priority。已经不是数据结构的课程，不需要大家去练习自定义数据结构了。