实验项目三 基于A*搜索算法迷宫游戏开发

原创

qq_58364169 于 2021-12-14 13:23:08 发布 2679 收藏
文章标签： 机器学习 python 数据挖掘

实验系列目录

计算器；贪吃蛇；基于A*算法的迷宫开发

文章目录

前言

新一周，新的实验任务。迷宫游戏是非常经典的游戏，在该题中要求随机生成一个迷宫，并求解迷宫。

一、需求分析

迷宫游戏是非常经典的游戏，在该题中要求随机生成一个迷宫，并求解迷宫。

要求游戏支持玩家走迷宫，和系统走迷宫路径两种模式。玩家走迷宫，通过键盘方向键控制，并在行走路径上留下痕迹；系统走迷宫路径要求基于A*算法实现，输出走迷宫的最优路径并显示。

设计交互友好的游戏图形界面。

要求：

1、迷宫随机生成

2、玩家走迷宫，留下足迹；

3、系统用A*算法寻路，输出路径

解决问题：

1、如何显示迷宫的图形界面；

2、如何生成随机的迷宫；

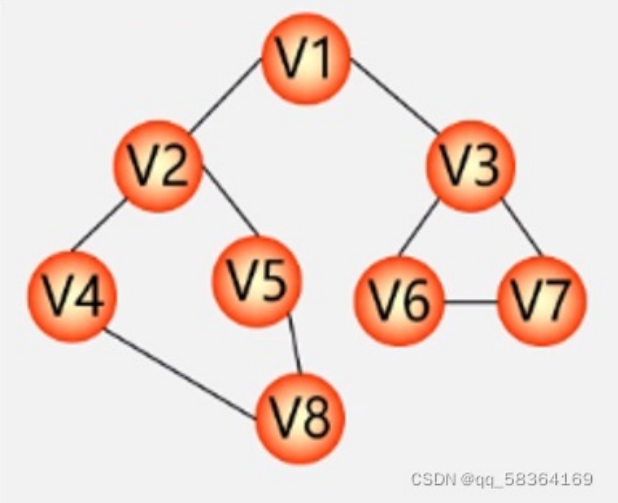
3、怎样移动游戏中走迷宫的“玩家”；

4、用A*算法求解迷宫；

二、迷宫地图分类

1.自然分叉型

第一种方法就是比较好理解的深度优先遍历算法（DFS），可以按照下图这样理解：



可以这样描述深度遍历：

（1）访问顶点v；

（2）从v的未被访问的邻接点中选取一个顶点w，重复第一步，如果v没有未访问的邻接点，回溯至上一顶点；

（3）重复上述两步，直至图中所有和v有路径相通的顶点都被访问到。

所以，访问次序就是下图这样：

因此，迷宫生成策略：利用深度遍历的思想。访问到一个节点时，搜索这个节点没有被访问过的相邻节点，选择一个继续做同样的操作，直到没有邻节点为止再回溯到上一个访问的节点，并选择另外的邻节点。

具体参考如下：

```
#include<stdio.h>
#include <conio.h>
#include<time.h>
#include<Windows.h>
#define LEVEL 2//迷宫生成方式,
#define M 50 //地图大小
#define Wall 1//障碍物
#define Road 0//通路
#define Start 2 //开始
#define End 3 //结束
#define Tmp 6 //临时值
#define Empty -1//空白
#define UP 'w'
#define Down 's'
#define Left 'a'
#define Right 'd'
#define Red 9
#define Blue 8
```

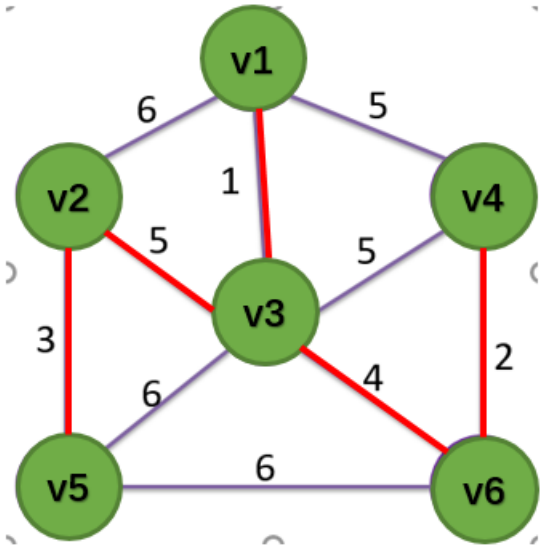
```
int level = 0;//当前等级难度
int count = 0;//设置地图大小
int map[M + 1][M + 1] = { Wall };//用二维数组存放地图中不同位置的标记
int my_x, my_y;//我（游戏小人）的位置

void SetMenuInfor();//菜单，设置等级和地图大小
void InitMap(int count);//初始化地图
void Creat_Map01(int, int);//深度优先
void Creat_Map02(int, int);//普利姆
int IsHaveNeighbor(int, int);//判断当前位置的四周是否还有有效通道Road
void printMap();//打印地图
//int move();
void find();//找到自己
int moveUp();//上移
int moveDown();//下移
int moveRight();//右移
int moveLeft();//左移
```


2.块状分割型

当然，上面的迷宫生成算法所生成的迷宫有点简单，主路比较明显。下面，我们将介绍第二种方法，随机性迷宫，也就是利用prime算法。

普利姆算法(prim)

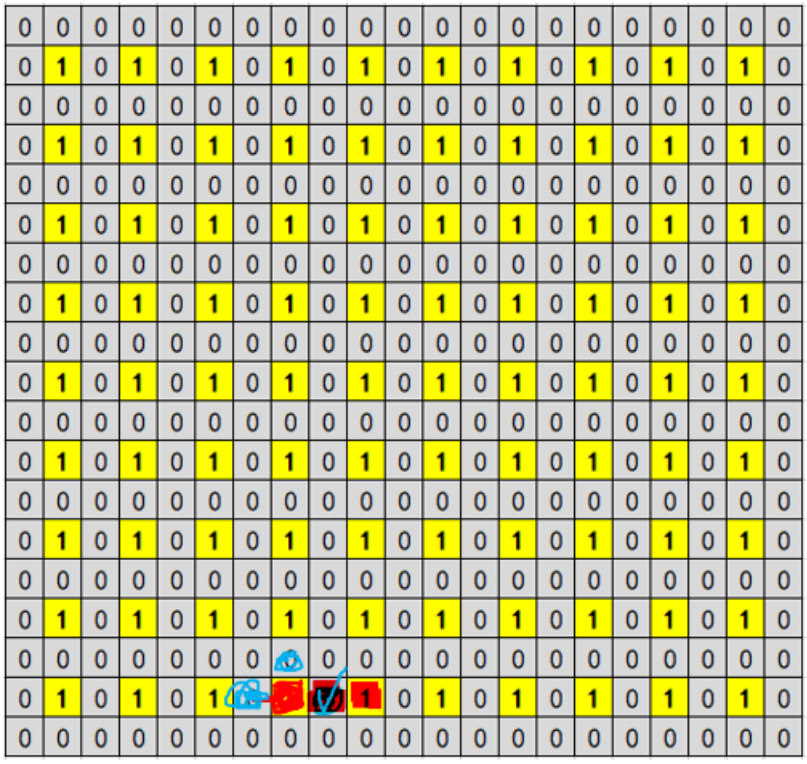


算法思想：

- 设N=（V,E）是连通网，*TE*是N上最小生成树中边的集合。
- 初始令U={U_0},(U_0∈V)，*TE*={ }。
- 在所有u∈U,v∈V-U的边（u,v）∈E中，找一条代价最小的边（u_0,v_0）.
- 将（u_0,v_0）并入集合TE,同时v_0并入U_0.
- 重复上述操作直至U=V为止，则T=(V.TE)为N的最小生成树。

算法理解分解如下：

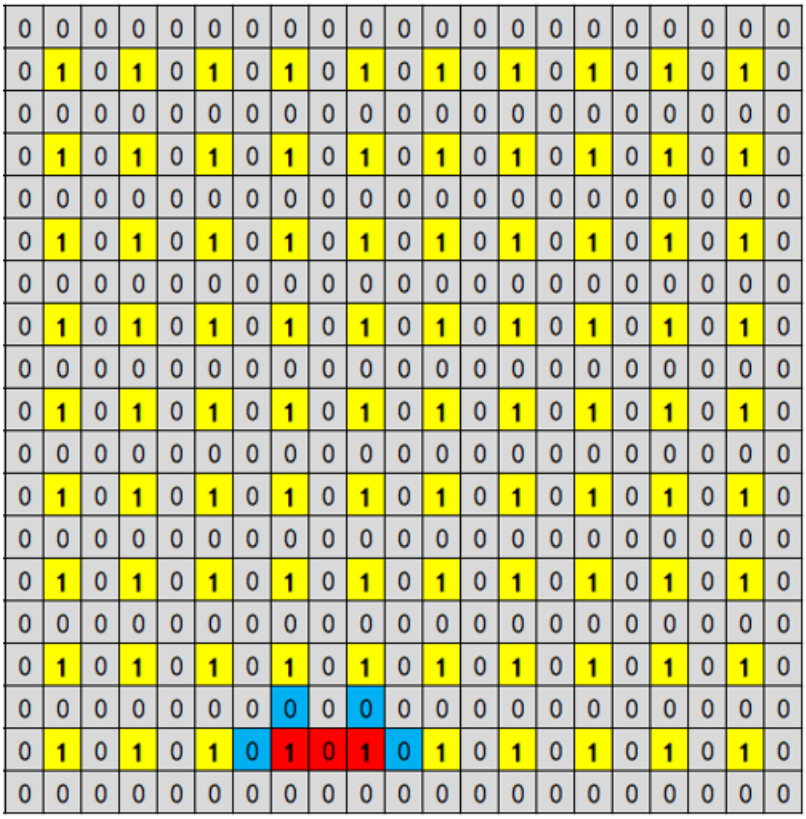
普利姆算法生成随机迷宫



3、随机选择一个蓝色0，然后看红色1隔着这个蓝色0对面的格子，是否是黄色的1，

1）如果是，则把对面的黄色1标记成红色1，即变成通路，然后把蓝色0变成红色的，即也变成通路；

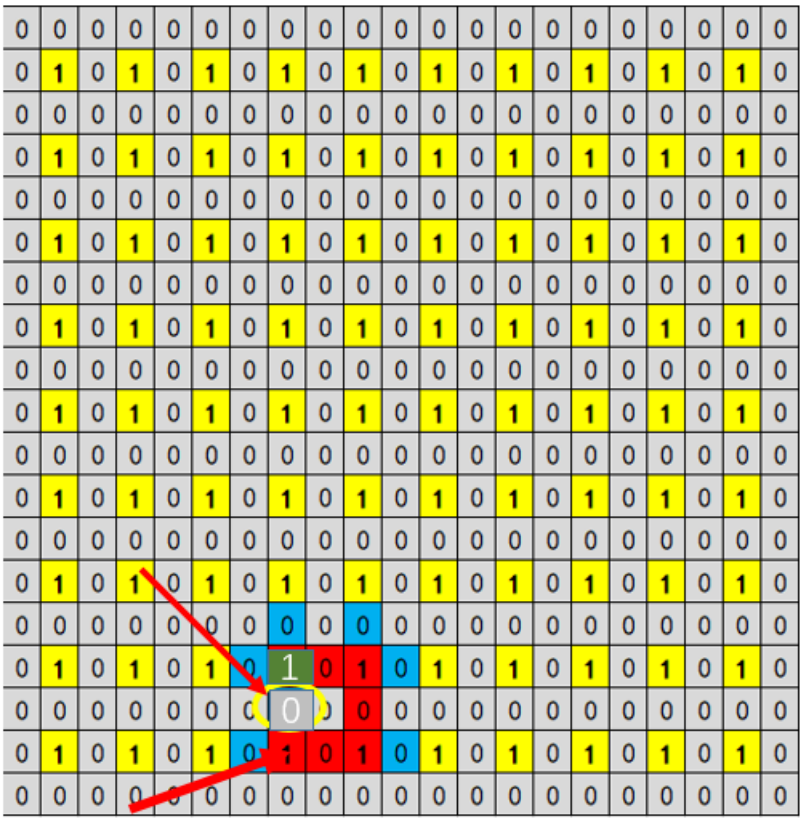
2）如果不是，就把这个蓝色的0变成灰色的；



4、继续为红色通路集合寻找待定的蓝色0，然后随机选择一个蓝色0，然后看红色1隔着这个蓝色0对面的格子，是否是黄色的1，

1）如果是，则把对面的黄色1标记成红色1，即变成通路，然后把蓝色0变成红色的，即也变成通路；

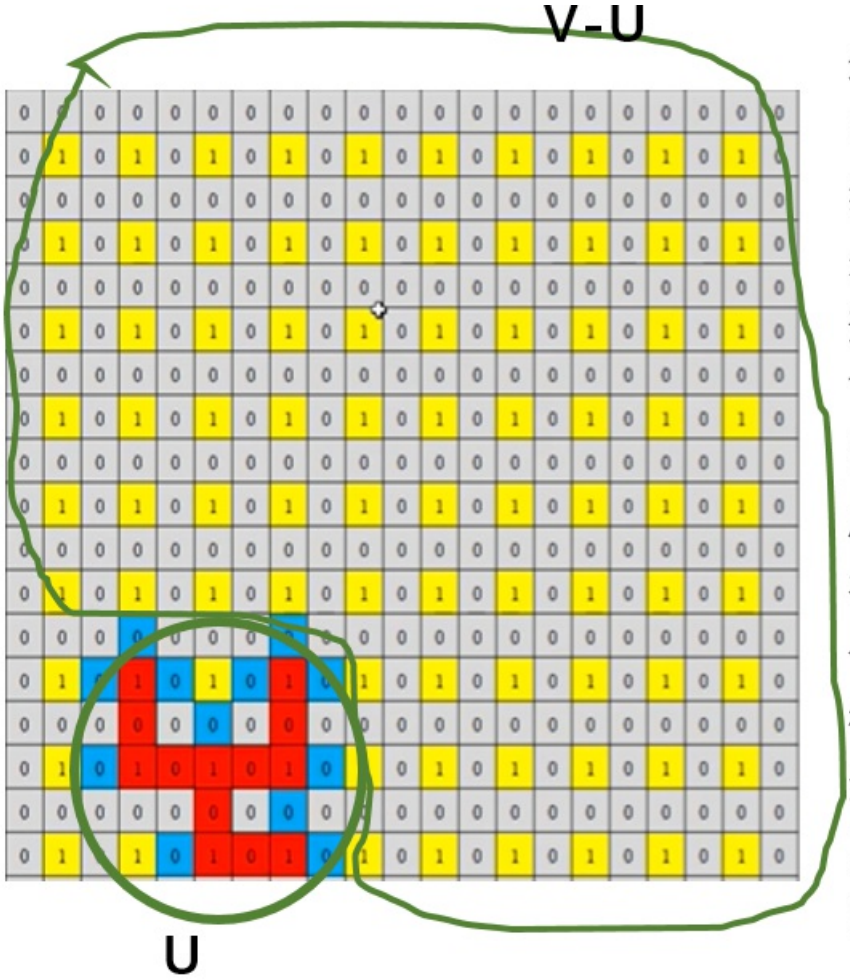
2）如果不是，就把这个蓝色的0变成灰色的；



2）如果随机选的蓝色的0隔着红色1对面的不是黄色1，说明这不是一条通路，就把这个蓝色的0变成灰色0，变成确定的障碍物；

接下来就继续随机找另一个蓝色的0；

即不停地重复上面的步骤3，直到整个地图都没有蓝色0了，地图就生成完毕。



这就是用的prim算法思想，把图中预置的通路节点分成两个集合，已经在通路里的属于U，未添加到通路里的属于V-U，通过这样的添加节点策略，把节点间的蓝色0当成最小生成树的边，依次添加到TE里，直到所有节点都添加到U里，这样，在任意两个节点间就形成了一个通路。prim算法形成的主路相对于深度优先算法，比较自然，但迷宫的分岔比较多，所以迷宫会更复杂，玩家需要做的选择次数可能会比较多。

当然，还有更多更复杂的算法生成更复杂的迷宫，在这里就不做过多介绍了。主要是本人知识理解也有限，最后附上迷宫项目的大体完成步骤吧。

用A*算法求解迷宫

第一步：

寻路算法

1）广度优先搜索算法

2）dijkstra(迪杰斯特拉)算法

3）Greed-Best-First（最好优先贪婪算法）

4）A*算法

路径规划指的是，依据某个优化准则（如工作代价最小、行走路径最短、行走时间最短等），在工作空间中找到一个从起始状态到目标状态能避开障碍物的最优路径。

总结

基于A*算法搜索迷宫路径的大概预习知识就是这样啦~