% 将必经节点联结成无间断路径，如果结点间不连续，则插入节点使其连续。

function [single\_new\_pop] = [generate\_continuous\_path](forGA_word.xls)(single\_pop, Grid, x)

i = 1;

single\_new\_pop = single\_pop; %传入的某行的初始路径，有20个路径节点

[~, single\_path\_num] = size(single\_new\_pop);

%遍历该行的所有节点，使其连续

while i ~= single\_path\_num

%%定位第i、i+1个节点的坐标

% 路径中第i个栅格在地图的列（从左到右编号1.2.3...）

column\_now = mod(single\_new\_pop(1, i), x) + 1;

% 路径中第i个栅格在地图的行（从上到下编号行1.2.3...）

row\_now = fix(single\_new\_pop(1, i) / x) + 1;

% 路径中第i+1个栅格在地图的列、行

column\_next = mod(single\_new\_pop(1, i + 1), x) + 1;

row\_next = fix(single\_new\_pop(1, i + 1) / x) + 1;

% 初始化最大迭代次数

max\_iteration = 0;

%% 判断点i和i+1是否连续,若不连续插入值(如果前后两节点的X坐标与Y坐标的差中较大值不等于1，说明不连续)

while max(abs(column\_next - column\_now), abs(row\_next - row\_now)) ~= 1

%取两节点的中点作为插入点,见forGA\_word.xls-sheet1

%插入点的横坐标 x\_insert，纵坐标 y\_insert

x\_insert = floor((column\_next + column\_now) / 2);%Y = floor(X) 将 X 的每个元素四舍五入到小于或等于该元素的最接近整数。

y\_insert = floor((row\_next + row\_now) / 2);

% 插入栅格为自由栅格

if Grid(y\_insert, x\_insert) == 0

% 插入的栅格序号

num\_insert = (x\_insert - 1) + (y\_insert - 1) \* x;

% 插入新序号（将当前的栅格序号中间插入一个新栅格序号 其他保持不变）

single\_new\_pop = [single\_new\_pop(1, 1:i), num\_insert, single\_new\_pop(1, i+1:end)];

% 插入栅格为障碍物栅格

else

% 往左走(如果当前待插入格（障碍物格)的左邻格不是障碍物 且 左邻格不是当前研究的两个格中任意一个)

if Grid(y\_insert, x\_insert - 1) == 0 && ((x\_insert - 2) + (y\_insert - 1) \* x ~= single\_new\_pop(1, i)) && ((x\_insert - 2) + (y\_insert - 1) \* x ~= single\_new\_pop(1, i+1))

x\_insert = x\_insert - 1;

% 栅格序号

num\_insert = (x\_insert - 1) + (y\_insert - 1) \* x;

% 插入新序号

single\_new\_pop = [single\_new\_pop(1, 1:i), num\_insert, single\_new\_pop(1, i+1:end)];

% 往右走 (如果当前待插入格（障碍物格)的右邻格不是障碍物 且 右邻格不是当前研究的两个格中任意一个)

elseif Grid(y\_insert, x\_insert + 1) == 0 && (x\_insert + (y\_insert - 1) \* x ~= single\_new\_pop(1, i)) && (x\_insert + (y\_insert - 1) \* x ~= single\_new\_pop(1, i+1))

x\_insert = x\_insert + 1;

% 栅格序号

num\_insert = (x\_insert - 1) + (y\_insert - 1) \* x;

% 插入新序号

single\_new\_pop = [single\_new\_pop(1, 1:i), num\_insert, single\_new\_pop(1, i+1:end)];

% 向上走

elseif Grid(y\_insert + 1, x\_insert) == 0 && ((x\_insert - 1) + y\_insert \* x ~= single\_new\_pop(1, i)) && ((x\_insert - 1) + y\_insert \* x ~= single\_new\_pop(1, i+1))

y\_insert = y\_insert + 1;

% 栅格序号

num\_insert = (x\_insert - 1) + (y\_insert - 1) \* x;

% 插入新序号

single\_new\_pop = [single\_new\_pop(1, 1:i), num\_insert, single\_new\_pop(1, i+1:end)];

% 向下走

elseif Grid(y\_insert - 1, x\_insert) == 0 && ((x\_insert - 1) + (y\_insert - 2) \* x ~= single\_new\_pop(1, i)) && ((x\_insert - 1) + (y\_insert-2) \* x ~= single\_new\_pop(1, i+1))

y\_insert = y\_insert - 1;

% 栅格序号

num\_insert = (x\_insert - 1) + (y\_insert - 1) \* x;

% 插入新序号

single\_new\_pop = [single\_new\_pop(1, 1:i), num\_insert, single\_new\_pop(1, i+1:end)];

% 如果各方向都无法插入则舍去此路径

else

%break\_pop = single\_new\_pop

single\_new\_pop = [];

break

end

end

column\_next = x\_insert;

row\_next = y\_insert;

max\_iteration = max\_iteration + 1;

%如果可以不断的增加新节点，但增加次数超过20000次，则舍弃此路径

if max\_iteration > 20000

single\_new\_pop = [];

break

end

end

if isempty(single\_new\_pop)

break

end

[~, single\_path\_num] = size(single\_new\_pop);

i = i + 1;

end