无人机赛课第4讲

--路径规划与A*算法

李潇翔

联系方式: lxx17@mails.tsinghua.edu.cn

时间：2019.10.26

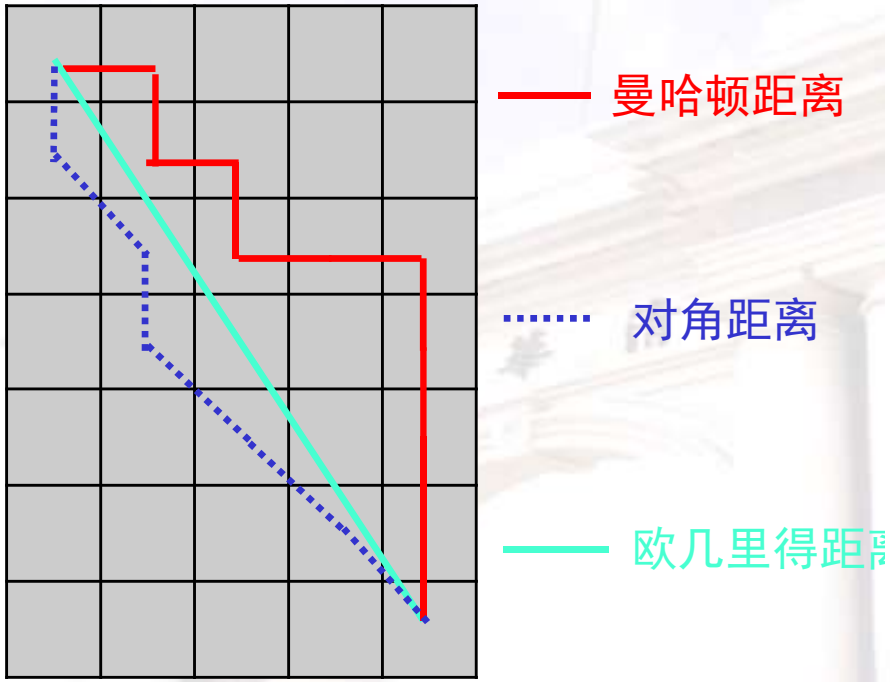
目 录

1 路径规划

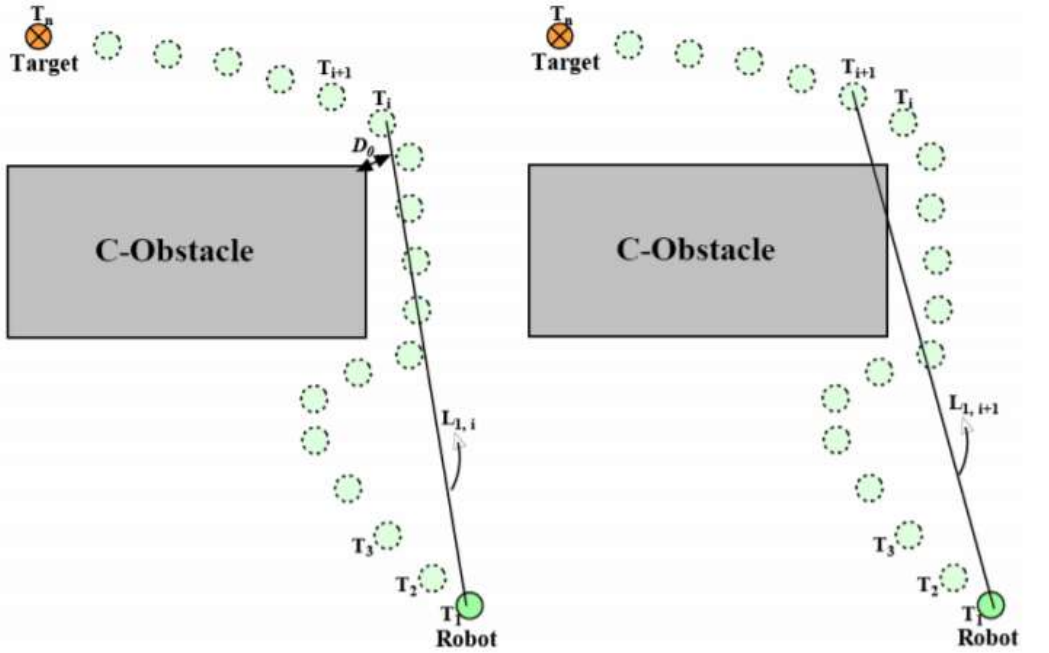
- 基本概念
  - 路径规划是运动规划(Motion planning)的重要研究课题
    - 物体的表示（Object representation）：点、多面体、操作臂等
    - 配置空间（Configuration Space）：一组表征对象位置的独立的参数
    - 精确式或启发式（Exact or Heuristic）
    - 全局规划或局部规划（Global or Local）
  - 路径（Path）： Cspace中起点到终点的曲线（或序列点）
  - 轨迹（trajectory）：路径 + 时间分配

1 路径规划

- 分类
  - 全局规划(大量先验信息) vs. 局部规划（局部传感信息）
  - 动态规划（在线） vs. 静态规划（离线）
  - 离散域 vs. 连续域



栅格法 A*搜索



人工势场法 [1]

[1] Li G, Tamura Y, Yamashita A, et al. Effective improved artificial potential field-based regression search method for autonomous mobile robot path planning[J]. *International Journal of Mechatronics and Automation*, 2013, 3(3): 141-170.

1 路径规划

- 路径规划问题的一般表示（以无人机为例）
  - 位姿表示：$P(x, y, z, \theta, \psi)$ 注意此处的坐标系选取方法！绝对坐标或相对坐标
  - 路径$r(q)$：$P_{\mathrm{s}} \xrightarrow{r(q)} P_{\mathrm{f}}$
    - 其中q为路径参数，表征长度、角度变量等
    - $P_{\mathrm{s}}, P_{\mathrm{f}}$为初末位姿，起点$P_{\mathrm{s}}$和终点$P_{\mathrm{f}}$由路径$r(q)$连接
  - 添加约束条件Π：$P_s\left(x_s, y_s, z_s, \theta_s, \psi_s\right) \xrightarrow{\Pi, r(q)} P_f\left(x_f, y_f, z_f, \theta_f, \psi_f\right)$
    - 可飞行性：运动学约束、机动性条件等
    - 安全性：避障、避撞等

$$\Pi = \begin{cases} \Pi_{\kappa}, & \kappa \leqslant \kappa_{\max} \\ \Pi_{\tau}, & \tau \leqslant \tau_{\max} \\ \Pi_{\mathrm{safe}}, & r_i(q) \cap r_j(q) = \varnothing, \quad i \neq j \end{cases}$$

曲率限制、挠率限制 } 动力学约束

避撞，安全性约束

1 路径规划

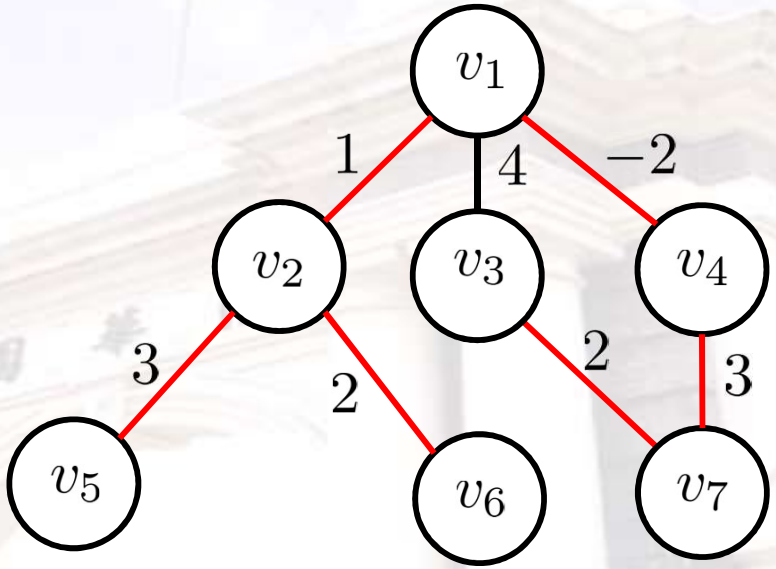
- 路径规划问题的一般表示（以无人机为例）

$$P_s\left(x_s, y_s, z_s, \theta_s, \psi_s\right) \xrightarrow{\Pi, r(q)} P_f\left(x_f, y_f, z_f, \theta_f, \psi_f\right)$$

- 协同规划与任务规划
  - 集群编队的优势：费效比和容错性
  - 核心在于信息共享
  - 任务分配、协调执行——分布式算法
  - 挑战：信息增多（$O(N^2)$）、相互干扰、一致性、稳定性等问题

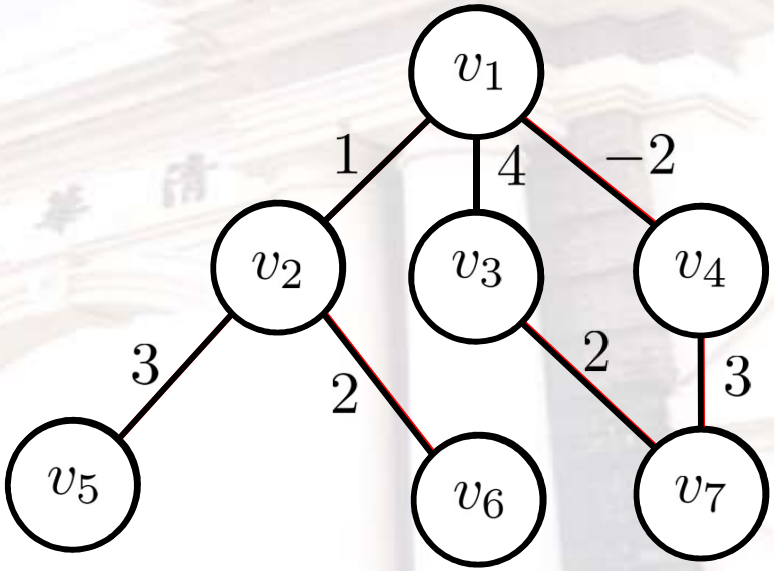
2 图论相关知识补充

- 基本概念 $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$
  - 顶点：$\mathcal{V} = \{v_1, v_2, \cdots, v_n\}$ ，顶点的度
  - 边：$\mathcal{E} = \{e_1, e_2, \cdots, e_m\}, \mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ ，有向图、无向图
  - 边的权重：赋权图、**最小生成树、最短路径树**



2 图论相关知识补充

- 搜索算法：利用计算机的高性能来穷举一个问题解空间的部分或所有可能情况；以图搜索为例。
  - 无权图：广度优先搜索($v_1, v_2, v_3, v_4, v_5, v_6, v_7$)
    深度优先搜索($v_1, v_2, v_5, v_6, v_3, v_7, v_4$)
  - 赋权图：**Dijkstra算法**[2]($v_1, v_4, v_2, v_6, v_5, v_7, v_3$)



[2] Dijkstra E W. A note on two problems in connexion with graphs[J]. *Numerische mathematik*, 1959, 1(1): 269-271.

附：Dijkstra算法伪代码

```
1  function Dijkstra(Graph, source):
2
3      create vertex set Q
4
5      for each vertex v in Graph:
6          dist[v] ← INFINITY
7          prev[v] ← UNDEFINED
8          add v to Q
10     dist[source] ← 0
11
12     while Q is not empty:
13         u ← vertex in Q with min dist[u]
14
15         remove u from Q
16
17         for each neighbor v of u:
18             alt ← dist[u] + length(u, v)
19             if alt < dist[v]:
20                 dist[v] ← alt
21                 prev[v] ← u
22
23     return dist[], prev[]
```

初始化

到初始点最短的路径

如果只关心source点到target点的最短路径，可以在u == target时中止循环

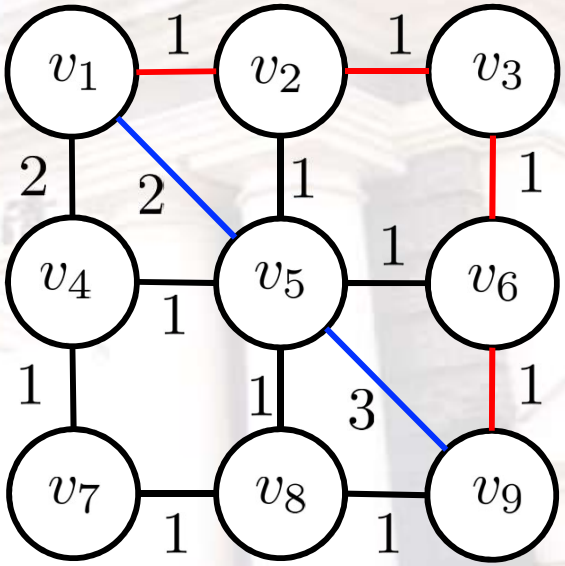
如果v到初始点的路径经过u能更短的话，替换掉

将路径的点逐一记录下来

* Acknowledgement to **Wikipedia** from *https://en.wikipedia.org/wiki/Dijkstra%27s_algorithm#cite_note-2*

图论相关知识补充

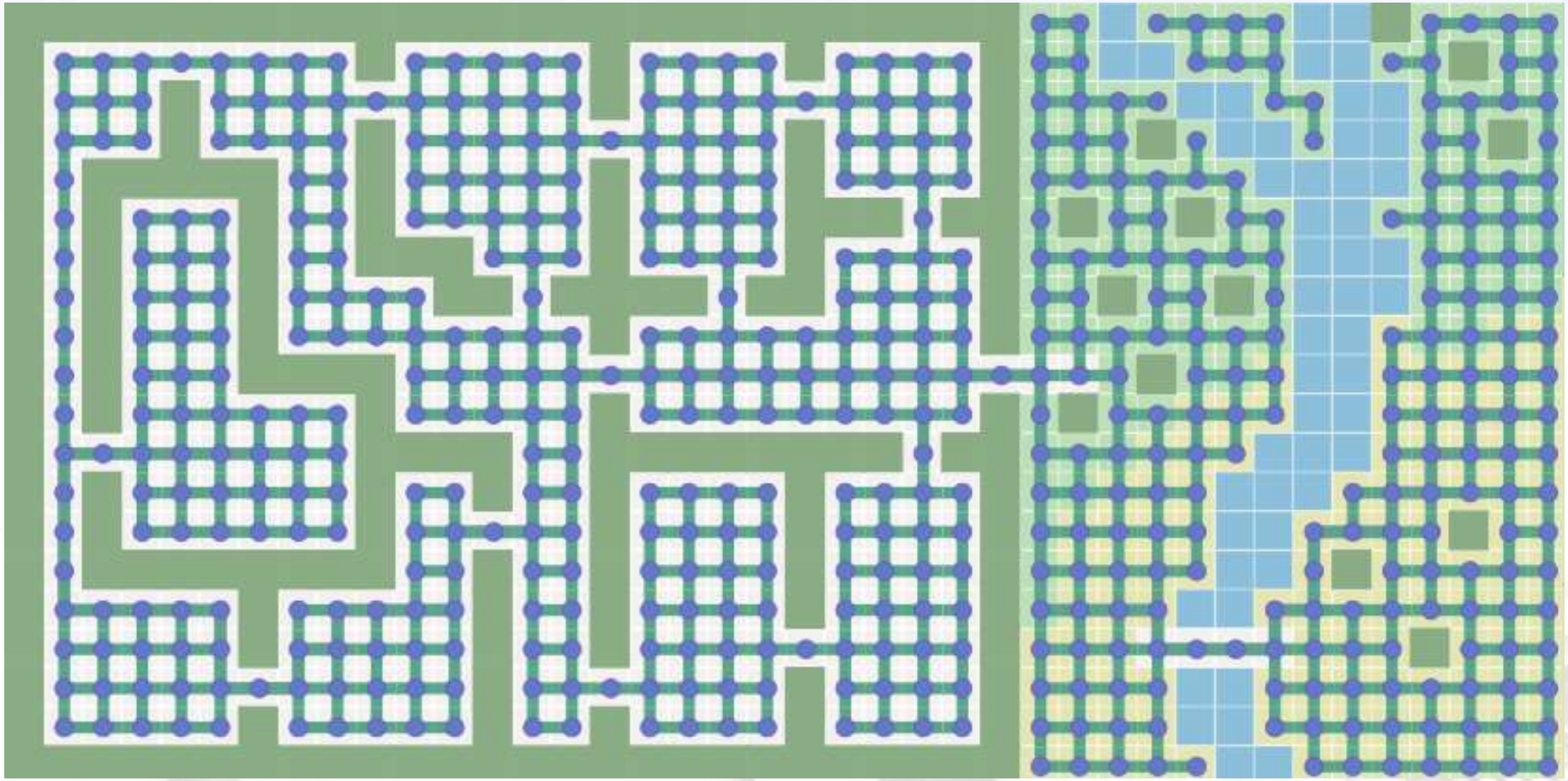
- 图的搜索
  - **Dijkstra算法**
  - 其他启发式算法：**Greedy Best-First Search（BFS）**
    - “沿着节点数最少的方向”
    - 如果 v_5 到 v_9 的权重改为1结果如何?



Q：搜索算法与路径规划有何联系？

2 图论相关知识补充

- 图的建模*



* Acknowledgement to **Red Blob Games** from *https://www.redblobgames.com/pathfinding/a-star/introduction.html*

3 A*算法介绍

- A* ≈ Dijkstra+ BFS
  - 作者[3]： Peter Hart, Nils Nilsson and Bertram Raphael
  - 表达式:

$$f(n) = g(n) + h(n)$$

从初始状态到状态n的实际代价

从状态n到目标状态的“最佳”路径的估计代价

到初始点的“贪心”

到终点的“贪心”

[3] Hart P E, Nilsson N J, Raphael B. A formal basis for the heuristic determination of minimum cost paths[J]. *IEEE transactions on Systems Science and Cybernetics*, 1968, 4(2): 100-107.

附：A*算法伪代码

```
function A*(start,goal)
    closedset := the empty set
    openset := set containing the initial node
    came_from := empty map
    g_score[start] := 0                    //g(n)
    h_score[start] := heuristic_estimate (start, goal)
    f_score[start] := h_score[start]

    while openset is not empty
        x := the node in openset having the lowest f_score[] value
        if x = goal
            return reconstruct_path(came_from,goal)
        remove x from openset
        add x to closedset
        for each y in neighbor_nodes(x)
            if y in closedset
                continue
            tentative_g_score := g_score[x] + dist_between(x,y)
```

已经估算完的集合

待估算的集合

初始化

f(x)最小的节点

这是节点的三个值！不仅仅和g、h函数有关，可能被更新的！

路径重构函数（后文介绍）

已经估算完毕，跳过

起点到y的（经过x的）距离

* Acknowledgement to **Wikipedia** from *https://zh.wikipedia.org/wiki/A*%E6%90%9C%E5%B0%8B%E6%BC%94%E7%AE%97%E6%B3%95*

附：A*算法伪代码

```
            if y not in openset
                tentative_is_better := true
            elseif tentative_g_score < g_score[y]
                tentative_is_better := true
            else
                tentative_is_better := false
            if tentative_is_better = true
                came_from[y] := x
                g_score[y] := tentative_g_score
                h_score[y] := heuristic_estimate (y, goal)
                f_score[y] := g_score[y] + h_score[y]
                add y to openset
    return failure
```

y不是待估算的节点

暂时判断为更优

这个g不是单纯的y节点的g函数~

而是openlist里面可能会被更新的g

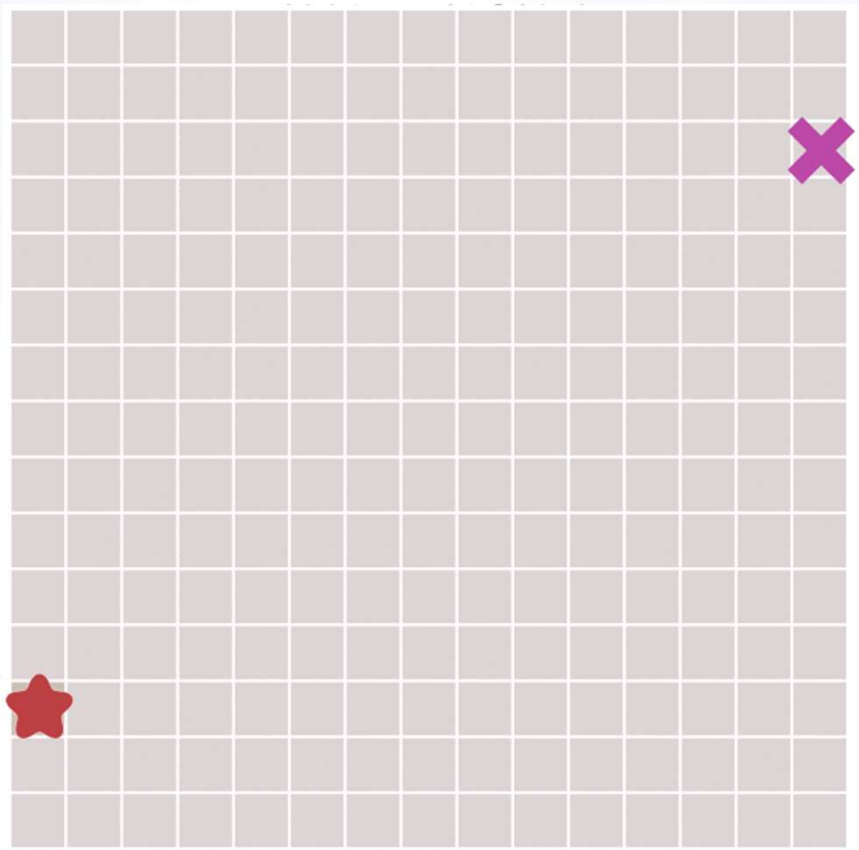
如果暂时判断更优（有必要更新）

y是x的子节点（记录路径）

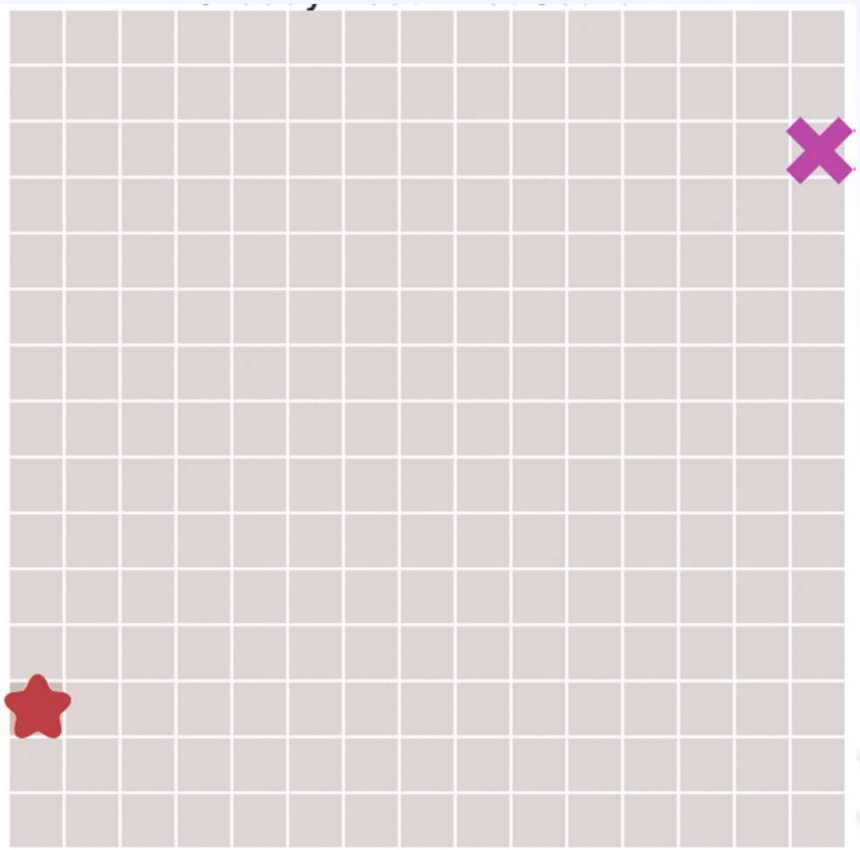
```
function reconstruct_path(came_from,current_node)
    if came_from[current_node] is set
        p = reconstruct_path(came_from,came_from[current_node])
        return (p + current_node)
    else
        return current_node
```

路径重构函数

4 几种算法的对比



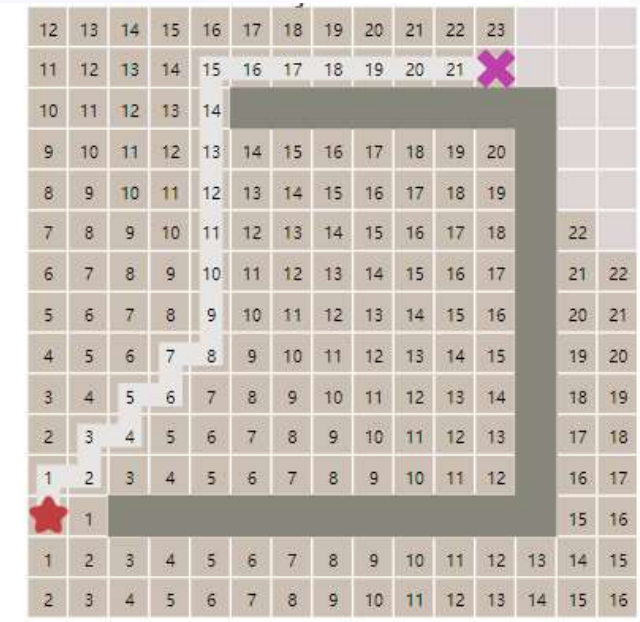
广度优先搜索



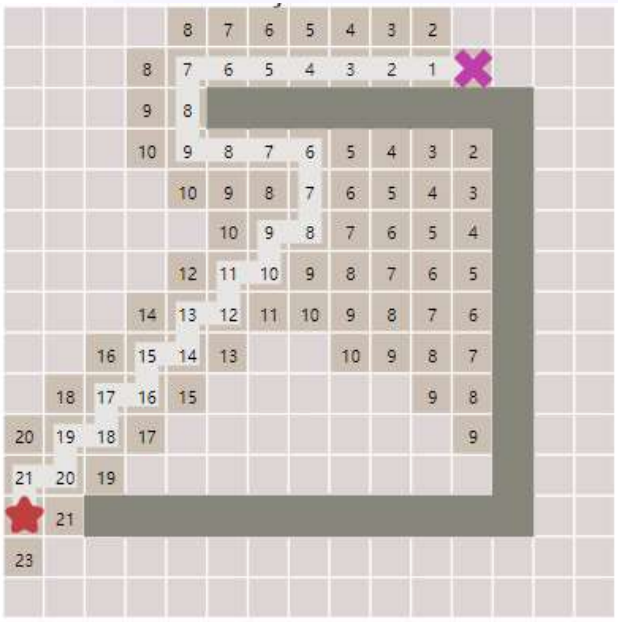
Greedy Best-First Search

* Acknowledgement to **Red Blob Games** from *https://www.redblobgames.com/pathfinding/a-star/introduction.html*

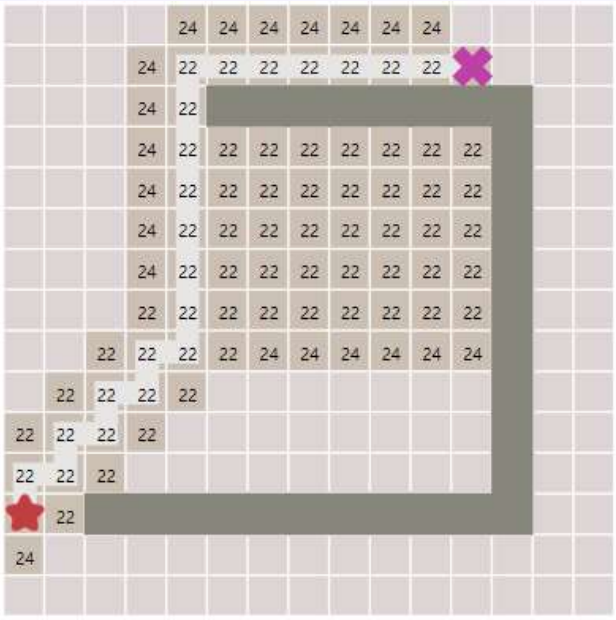
4 几种算法的对比



Dijkstra



Greedy Best-First Search



A*

* Acknowledgement to **Red Blob Games** from *https://www.redblobgames.com/pathfinding/a-star/introduction.html*

谢谢！