

# A\*算法

A\*算法与寻找最短路径的Dijkstra算法有相似之处。（不如说A\*其实是Dijkstra的改版，A\*算法的应用场景仅限寻找起始状态到目标状态之间的最短路径，Dijkstra能找到起始状态到所有节点的最短路径）。

## 原理

A\*算法最特别之处就是估价函数 $F(n)=G(n)+H(n)$

其中 $F(n)$ 表示起始状态经过状态 $n$ 到目标状态的估计代价

$G(n)$ 表示起始状态到状态 $n$ 的最小代价

$H(n)$ 表示状态 $n$ 到目标状态的估计代价

与Dijkstra算法一样，所有的状态（点）可分为三类：

黑色：已经确定了起始状态到其的最短路径

灰色：即将访问

白色：尚未访问

从起始状态开始（起始状态标黑），将其能到达的所有状态标灰，确定他们的 $G$ 值（暂时忽略 $H$ 值，并算出 $F$ 值）。

接下来重复一下操作：

1. 寻找灰色状态中 $F$ 值最小的，成为当前状态
2. 当前状态标黑(即 $G$ 值已确定)
3. 对于当前状态能到达的状态：
  1. 如果它是黑色，则略过。
  2. 如果它是白色，计算它的 $G+H=F$ 值，并标灰，把当前状态作为其父状态。
  3. 如果它是灰色，检查 $G$ 值能否更新，若能，则更新 $G$ 值并将其父状态改为当前状态。
4. 若目标状态被标黑，则最短路径已找到。
5. 若目标未黑而已没有灰色状态，则路径不存在。

## 与Dijkstra比较

Dijkstra仅以最短路径(即A\*中的 $G$ )为优先搜索的判断依据，A\*加入了 $H(n)$ 更关注到目标状态的最短路径。Dijkstra可应用于抽象的图模型，而A\*因为需要构建 $H(n)$ 估价函数，若图比较抽象，则构建有困难。

## 关于估价函数 $H(n)$

有几种不同的构建方法。

1. 直线距离。
2. 曼哈顿方法，即在瓦块图中，水平距离加垂直距离。

From:  
<https://wiki.cvbbacm.com/> - CVBB ACM Team

Permanent link:  
[https://wiki.cvbbacm.com/doku.php?id=2020-2021:teams:no\\_morning\\_training:a\\_%E7%AE%97%E6%B3%95&rev=1590314193](https://wiki.cvbbacm.com/doku.php?id=2020-2021:teams:no_morning_training:a_%E7%AE%97%E6%B3%95&rev=1590314193)

Last update: 2020/05/24 17:56