启发式搜索

尽管广度优先搜索、深度优先搜索加上有效的剪枝方法，可以解决很多问题，但这两种搜索都是盲目的，它们不管目标在哪里，只管按照自己的方式搜索，会存在很多没必要的搜索。有没有一种启发式搜索算法，可以**启发程序朝着目标的方向搜索**，从而提高搜索效率呢？

启发式搜索算法对每一个搜索状态都进行评估，选择估值最好的状态，从该状态进行搜索直到目标。如何对一个状态进行评估呢？一个状态的当前代价最小，只能说明从初始状态到当前状态的代价最小，不代表总的代价最小，因为余下的路还很长，未来的代价有可能更高。因此**评估需要考虑两部分：当前代价和未来估价。**

评估函数 f(x)：f(x)=g(x)+h(x)，其中，g(x)表示从初始状态到当前状态 x 的代价，h(x)表示从当前状态到目标状态的估价，**h(x)被称为启发函数。**

常用的启发式搜索算法有很多，例如 A*、IDA*、模拟退火算法、蚁群算法、遗传算法等。

一　A*算法

A*算法是**带有评估函数的优先队列式广度优先搜索算法。**在广度优先搜索时维护一个优先队列，每次都从优先队列中取出评估值最优的状态进行扩展。第 1 次从优先队列中取出目标状态时，即可得到最优解。A*算法提高搜索效率的关键在于启发函数的设计，不同的启发函数，其搜索效率不同。启发函数 h(x)越接近当前状态到目标状态的实际代价 h′(x)，A*算法的效率就越高。启发函数的估值不能超过实际代价，即 h(x)≤h′(x)。

如果启发函数的估值超过实际代价，则失去意义。例如，如果当前节点到目标的实际最短距离为 30，当前节点的启发函数估值为 50，另一个节点的启发函数估值为 100，则在两个节点已走过路径长度 g(x)相同的情况下，不能说明当前节点就一定比另一个节点优，也没有比较的意义，反正两个都不优。

如果令所有状态的 h(x)都为 0，则退化为普通的优先队列式广度优先搜索算法，不再有启发式搜索的作用。

二　IDA*算法

IDA*算法是带有评估函数的迭代加深 DFS 算法。深度优先搜索有可能跌入一个无底深渊，搜索了很多步也无法找到问题的解，因此要对搜索的深度加以限制，超过该深度便不再搜索，立即回溯。迭代加深 DFS 算法是深度优先搜索算法的一种变形，事先限定一个深度 depth，在不超过该深度的情况下进行深度优先搜索，如果找不到解，则增加深度限制，重新进行搜索，直到找到目标。IDA*算法设置了一个评估函数 f(x)：当前深度+未来估计步数，当 f(x)>depth 时立即回溯，避免无效搜索，提高效率。在很多情况下，IDA*算法的效率更高，代码更少。