杨素琼

基于A*算法的地图路径搜索的实现

杨素琼 林碧琴 何 伟

摘 要 最短路径问题(SP)是人工智能的一个活跃方向,本文介绍了人工智能中的一般启发式搜索算法的原理及算法的优点,搜索过程,并将其应用到公园导游系统的路径搜索中,给出了一种搜索公园导游地图最短路径的算法及其实现。

关键词 人工智能 启发式搜索算法 公园导游地图 最短路径

Implementation of Search for Map Path Based on A* Algorithm

Yang Suqiong Lin Biqin He Wei
(Northern Jiaotong University, Beijing, 100044)

Abstrast The problem of the shortest path is a active aspect of artificial intelligence. This paper introduces the theory of artificial intelligence heuristic search algorithm, the advantage and search process of A* algorithm. A algorithm and implementation of search for the shortest path on park guide map is presented on the basis of applying A* algorithm to the path search for park guide system.

Keywords artificial intelligence, heuristic search algorithm, park guide map, shortest path

模仿人类问题求解的能力是人工智能最基本最重要的一项任务,知识表示和搜索技术是人工智能问题求解领域的两大基本课题。搜索是人工智能中最基本的使用技术,由于人工智能涉及的已知信息往往是不完全的,这就要求对问题的已知空间进行搜索,快速找到正确答案。搜索包括对知识块的匹配、选择和解释;匹配和解释的结果往往引起再搜索。

1 启发式搜索法

尼尔逊(N. J. Nillson)指出,在智能过程中,搜索是不可避免的。搜索是人工智能中最基本的问题之一,许多问题的求解都可以归结为搜索。搜索方法主要有两大类,一类是盲目搜索,另一类是启发式搜索。盲目搜索是指在不具有对特定问题的任何有关信息的条件下,按固定的步骤进行的搜索;而启发式搜索则是利用与问题有关的信息,尽量减少不必要的路径,以求尽快到达目标。

与问题有关的信息,称为启发信息。启发信息通常用在待扩展的节点上,使得搜索总是沿着那些被认为是最有希望的区段来扩展。

我们用公式:

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

其中 $f^*(n)$表示从始节点 s 通过节点 n 到目标节点的最佳路径的代价,$g^*(n)$表示从始节点到节点 n 的最佳路径代价,$h^*(n)$表示从 n 到目标节点的最佳路径的代价。

$f^*(n)$通常是未知的,我们用 $f(n)$来作为它的近

杨素琼 北方交通大学电子与信息工程学院 在读硕士研究生 100044 北京市
林碧琴 北方交通大学 副教授 100044 北京市
何 伟 北方交通大学 在读博士研究生 100044 北京市

似估计，即

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

这里 $g(n)$ 表示迄今为止搜索已产生的从 s 到 n 的所有路径中最佳路径的代价，$h(n)$ 表示对 n 到目标节点代价的估计，$h(n)$ 称为启发函数。$h(n)$ 涉及对未搜索路径的估计，它的精确程度要靠我们前面一再强调的依赖于问题领域的启发信息。估计值越小的节点，被认为希望度越高，应该优先扩展。

启发式搜索的基本原理是：对于搜索过程中遇到的每个新状态（或者说新节点），按估价函数计算出它的最佳代价估计值，然后选出当时估计值最小的状态，从该状态开始继续搜索。这种搜索实际上是以节点的代价估计值为标准的最佳优先搜索。

2 启发式 A* 搜索算法

在介绍算法之前，先说明算法中用到的几个数据结构：结构 G，表示当前已生成的显式搜索图；一张 OPEN 表，存放已生成而尚未进行扩展处理的节点；一张 CLOSED 表，存放已生成且进行过处理的节点。

启发式搜索算法（A* 算法）：

(1)建立一个只由初始节点 s 组成的搜索图 G；把始节点 s 送 OPEN 表，OPEN：=(s)；CLOSED：=()；即置 $f(s) = 0 + h(s)$。

(2) LOOP：if OPEN=() then EXIT，即失败结束。

(3) n：=FIRST(OPEN)，使 f(n) 最小；REMOVE(n,OPEN)；add(n,CLOSED).

(4)若 n 为目标节点，则成功，算法结束（若感兴趣的是目标节点，则给出 n 的状态；若关心的是路径，则通过追踪主链的指针，给出 s 到 n 的路径）。

(5)扩展节点 n，生成 n 的所有不是它的先辈节点的后继节点集 M={mi}，把 mi 作为 n 的后继节点添入 G。

(6)若 mi 没有在 OPEN 和 CLOSED 表中出现过，则把 mi 加入到 OPEN 中。

(7)若 mi 在 OPEN 表中有重复节点 k，且 g(mi)<g(k)，则 remove(k,OPEN)；add(mi,OPEN)。

(8)若 mi 在 CLOSED 表中有重复节点 k，且 g(mi)<g(k)，则

a. CLOSED 表的节点 k 改为节点 mi；

b. 按后继元表，修改 k 的所有在 OPEN 和 CLSOED 表中的后裔的 g、f 值。

(9)按 f 值从小到大的次序，对 OPEN 表中的节点重新排序。

(10) go LOOP.

在 OPEN 和 CLOSED 两张表中的节点，除含有它的状态描述外，还包含该节点的代价估计值 f 和 g，后继元表，以及一个主链的前趋指针，指明该节点在通向初始节点的最佳通路上的父节点。有了主链，使得找到目标节点后，只要沿着主链的指针追踪到初始节点，便可获得整个求解路径。

在 A 算法中，若对每一 n 均有 $h(n) \leqslant h^*(n)$，则算法 A 一定能找到一条到达目标节点的最佳路径。此时，算法 A 称为算法 A*。对任意图，如果存在从起始节点到目标节点的路径，则总是结束在一条从起始节点到目标节点的最佳路径的搜索算法被称为是可采纳的算法。A* 算法是可采纳的算法。

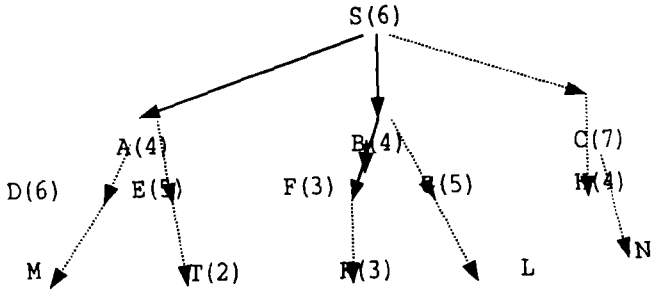


图1 A*算法搜索树

如图 1 有如下状态空间，可以看出 A* 算法的具体搜索过程，设起始位置是*S，终点位置是 T，字母后的数字表示该节点的评价函数值。

具体搜索过程如下：

(1)初始状态：

OPEN=[S(6)]；CLOSED=[]；

(2) 估算 S(6)，取得所有子节点，并放入 OPEN 表中；

OPEN=[A(4),B(4),C(7)]；CLOSED=[S(6)]；

(3) 估算 A(4)，取得所有子节点，并放入 OPEN 表中；

OPEN=[B(4),E(5),D(6),C(7)]；CLOSED=[A(4),s(6)]；

(4) 估算 B(4)，取得所有子节点，并放入 OPEN 表中；

OPEN=[F(3),G(5),E(5),D(6),C(7)]；CLOSED=[B(4),A(4),S(6)]；

(5) 估算 F(3),取得所有子节点,并放入 OPEN 表中;

OPEN =[T(2),K(3), G(5),E(5),D(6),C(7)];CLOSED=[F(3),B(4),A(4),S(6)];

(6) 估算 T(2),已得到解。

图 1 中的实线路径即是所要搜寻的结果,T(2)是所要找寻的目标,顺着主链指针追踪到初始节点,便可得到整个求解路径 S(6)→B(4)→F(3)→T(2)。

3 应用 A* 算法寻找地图路径

导游系统软件有着广泛的应用前景,而地图路径搜索是导游系统必不可少的一部分,占有极其重要的位置。要找出两个景点之间的路径,如果只应用一般的广度搜索或宽度搜索等盲目搜索法,从原则上讲该问题是可以求解的。只要找出图中各种可能的路径,再进行比较,取最短的那条即可。但是对于象颐和园这样景点路径比较多的导游系统,此法就存在着很多的实际缺陷。一是可能需要过长的搜索时间,二是有可能出现我们前面说过的"爆炸"问题。因此必须采用启发式搜索,以求尽快找到一条两个景点之间的最短路径。

对于估价函数,$f(n) = g(n) + h(n)$,$g(n)$ 为从起始点到目前节点的最佳路径,$h(n)$ 为启发函数,对启发式函数的讨论如下:

$$h(n_j) = \rho(n_i,n_j) + W\{D(s,t)d(n_i,n_j)\}$$

其中,n_i 是当前节点,n_j 是当前节点的后继子节点之一。启发函数 h_j 中包含有角度和距离两个因素,以起始节点和目标节点之间的连线为零度基准线,$\rho(n_i,n_j)$ 表示 ni 与 n_j 之间的连线与基准线之间的角度,用弧度来表示。$d(n_i,n_j)$为 n_i 与 n_j 的距离,为 $D(s,t)$起始节点与目标节点之间的距离,W 为加权系数。启发函数包括了方位和距离两个因素,加大了启发的信息量,加快了搜索的过程和准确度。

路径的搜索过程可以描述如下:

```
search()
{
OPEN =[起始节点];CLOSED= [];
While (OPEN 表非空)
{
  从 OPEN 中取得一个节点 S,并将其从 OPEN 表中删除
  if(S 是目标节点)
  {
    求得路径 path,并返回路径 path;
  }
  else
  {
    for(S 的每一个子节点 Y)
    {
      根据估价函数计算节点 Y 的估价值;
      if(Y 不在 OPEN 表和 CLOSED 表中)
      {
        if(Y 的估价值小于 OPEN 表的估价值)
          更新 OPEN 表中的估价值;
      }
      else
      {
        if(Y 的估价值小于 CLOSED 表的估价值)
          {
            更新 CLOSED 表中的估价值;
            从 CLOSED 表中移出节点,并放入 OPEN 表中;
          }
        }
      将节点 S 插入 CLOSED 表中;
      按估价值将 OPEN 表中的接点排序;
    }//end for
  }//end else
}//end while
}//end function
```

4 算法搜索效果举例

该算法可用 Visual C++实现,根据公园地图的特点,公园地图的表示可用景点、交叉路口为节点,相邻的节点和它们之间的连接来表示,其中所有的节点和连接都有不同的属性,最基础的部分是点的信息,导游图由大量的点对象通过一定的连接而组成。

图 2 是颐和园导游地图的一部分,我们应用上述启发式搜索算法对其进行路径搜索。要求寻找从乐寿堂到多宝塔的路径,图中的黑线即为搜索的结果。

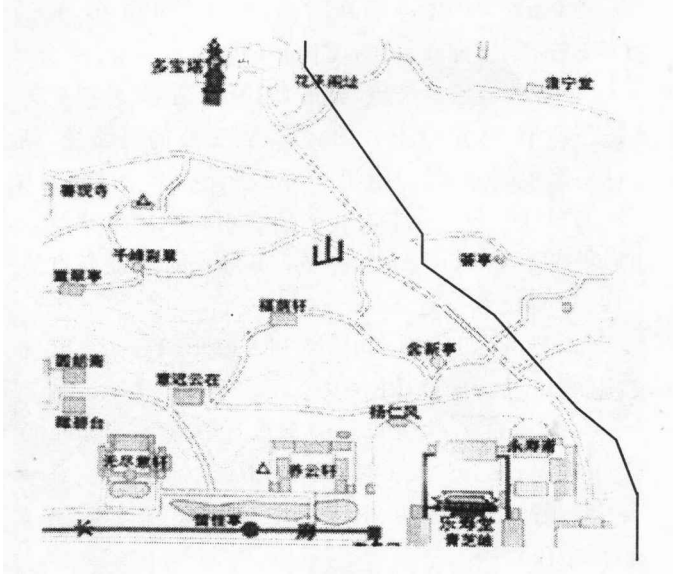
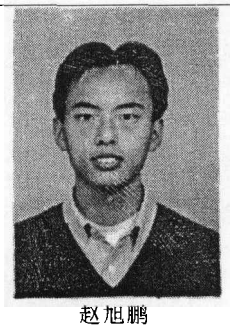


图 2 A* 算法效果搜索实例


赵旭鹏

多重系统引导的原理与实例分析

赵旭鹏

摘 要 作者首先论述了微机上多重操作系统引导的实现过程和工作原理，然后通过实例来进行两种实现方式间的比较与分析。

关键词 多重 操作系统 引导 分析

1 引言

随着计算机应用的日益普及，与计算机相关的硬件及软件都有了迅猛的发展。目前主流配置的PC机的硬盘达到8G以上，内存32M以上和赛扬以上级别的CPU。这样一来，除了安装一个操作系统以及各种软件以外，还会闲置大量的硬盘空间。这就为在同一台计算机上安装多个操作系统提供了可能。尽管Windows 9x是当今PC机的主流平台，但程序运行要求的"包袱"也较多。而出于实际运行"简洁"的需要和为了维护以往投资等因素的考虑，DOS仍然还有其用武之地。在各类微机实验室或在某些应用场合，还需要经常的进行多种操作系统，如UNIX、LINUX、NT/2000、Windows 9x/3x、NOVELL、DOS、BeOS等平台的更换，以便于进行教学、研究之用。然而人们不太可能为每一种系统都单独准备出相当数量的机器，因此出于节约开支并充分利用资源的考虑，在同一台计算机上进行多个操作系统的安装就成为必要的了。

下面让我们先从理论多重系统引导的实质入手，然后再以作者所做实例为蓝本，对两种主要的多重系统管理方式进行分析与比较，以期为读者的选择提供参考。

2 系统引导过程

为了理解多重系统引导的实现，我们首先要知道微机的系统引导过程，一般PC机的系统引导过程(从硬盘引导)大致分4步：

a. 机器加电检测进行BIOS调用并执行硬盘主引导扇区中的主引导程序；

一个完整的硬盘主引导记录共有512个字节，它们在硬盘中占用一个扇区，称为主引导扇区，该扇区位于硬盘的0面0道1扇区。它分为3个主要部分：主引导程序、4个硬盘分区表、硬盘赋权标记。其中主引导

赵旭鹏 北方交通大学 在读硕士研究生 100044 北京市

5 结束语

从算法的时间复杂度和空间复杂度来看，本算法是可取的，对导游系统来说，它可以尽快的找到所需的最短路径。对于公园导游这样一个特殊的系统，游客会有不同的需求，比如说有的在从一个景点到另一个景点的途中还希望顺带游览其它的景点，这样需要找出来的可能就不是最短路径，而是满足附带要求的尽可能短的路径。这时对这一算法就需要做一些改进，如引入感兴趣的集，启发式搜索算法恰当地应用感兴趣的集，则可以满足不同的需求。

6 参考文献

1 Nilsson N J. Principles of artificial intelligence. NewYork: Tioga Publishing Co, 1980

2 俞瑞钊，史济建. 人工智能原理与技术. 杭州：浙江大学出版社，1993. 201－211

3 吴泉源，刘江宁. 人工智能与专家系统. 北京：国防科技大学出版社，1995. 27－36

4 杨宪泽. 基于图搜索算法的探讨. 西南民族学院学报，自然科学版，1998，24(2)：117－122

(责任编辑：张树增 收稿日期：2000－04－14)