文章编号:1674-2869(2013)06-0050-05

# 利用堆排序优化路径搜索效率的分析

孙玉昕,章 瑾
(武汉工程大学计算机科学与工程学院,湖北 武汉 430074)

**摘 要**:随着计算机技术的发展,路径搜索算法在许多领域内得到广泛的应用,对搜索时间要求提出更高的要求.为了解决这一问题采用基于人工智能的启发式搜索算法,利用网络拓扑图给出的信息动态地调整搜索方向,并利用二叉堆进行算法优化,从而达到提高搜索效率的要求.常规使用启发式搜索算法进行路径搜索计算,其时间复杂度是$O(n_2)$($n$为网络节点数量),即当面临百万节点的复杂网络拓扑时,启发式搜索算法的搜索耗时将会呈指数级快速增长,无法完全满足工程技术需求.通过理论分析与实验数据证明应用二叉堆的启发式搜索算法对于长路径,大搜索空间的搜索应用时表现出良好的时间线性,其时间复杂度是$O(\log n)$($n$为Openlist的节点数),没有出现常规启发式搜索算法应用时搜索时间爆炸式增长的情况,具有较高的性能和效率,对工程实践有一定的实用参考实用价值.

**关键词**:路径搜索;启发式搜索算法;排序;二叉堆
**中图分类号**:TP391.9 **文献标识码**:A **doi**:10.3969/j.issn.1674-2869.2013.06.010

## 0 引 言

路径搜索的核心思想是利用计算机的处理能力,准确高效地在网络中任意两个或多个节点之间寻找出最佳路径.路径搜索算法在计算机工程中有着广泛的应用价值,比如地理信息系统、位置服务、智能交通、智能机器人等领域[1].在工程实践中,路径搜索的时间效率和空间效率是判断算法的优劣重要指标[2].启发式搜索和盲目搜索相比,可省略大量无谓的搜索路径,已能够极大提高搜索效率,但当面临百万节点的复杂网络拓扑时,启发式搜索算法的搜索耗时将会呈指数级快速增长,无法满足需求,因此考虑引入二叉堆进行进一步的算法优化,使得搜索速度进一步提升.

## 1 实验部分

### 1.1 算法选择

路径搜索的核心算法就是最短路径算法 ,它是计算机科学与地理信息科学等领域研究的热点.目前已知的最短路径算法主要有 Floyd (弗洛伊德)算法、矩阵算法和 Dijkstra(迪杰斯特拉)算法.其中 Dijkstra 算法是最为经典的最短路径算法,其主要特点是以起始点为中心逐层外推,直到推进至终点.Dijkstra 算法能确保求出最短路径的最优解,但由于它是逐层遍历的方式导致其效率比较低,无法满足工程实际的需求[3].

为了满足效率和灵活性要求,基于人工智能的启发式搜寻法(Heuristic Search Methods)的A*算法常被应用路径搜索[4].所谓启发式是在Dijkstra算法基础上,引入启发函数(Heuristic Function)来估算当前节点与目标节点的距离,通过启发函数的估算值不断动态调整搜索方向.与Dijkstra算法的盲目搜索过程,A*算法的启发式搜索方法充分利用网络拓扑图给出的信息来动态地调整搜索方向,因此搜索效率更高[5-6].

A*算法的核心计算公式表示为

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

其中 $f(n)$ 是途经节点 $n$ 从初始点到目标节点的路径距离的估价函数值,$g(n)$ 是在状态空间中从初始节点到 $n$ 节点的实际路径距离,$h(n)$是从 $n$ 到目标节点的距离的启发函数.$f(n)$值越小则表示该结点的路径越短[7-9].

A*算法流程图如图1所示.运用A*算法进行路径搜索计算.A*算法的时间复杂度是$O(n2)$,$n$为网络节点数量.因此,随着网络节点数和路径长度的增加,A*算法的搜索耗时呈指数级的快速增长.在工程实际中,经常需要对大规模的节点进行搜索,当面临百万节点的复杂网络拓扑时,A*算法的效率仍然无法满足需求[5,7].

**收稿日期**:2013-04-03
**作者简介**:孙玉昕(1977-),女,黑龙江大庆人,讲师,硕士.研究方向:地理信息系统.

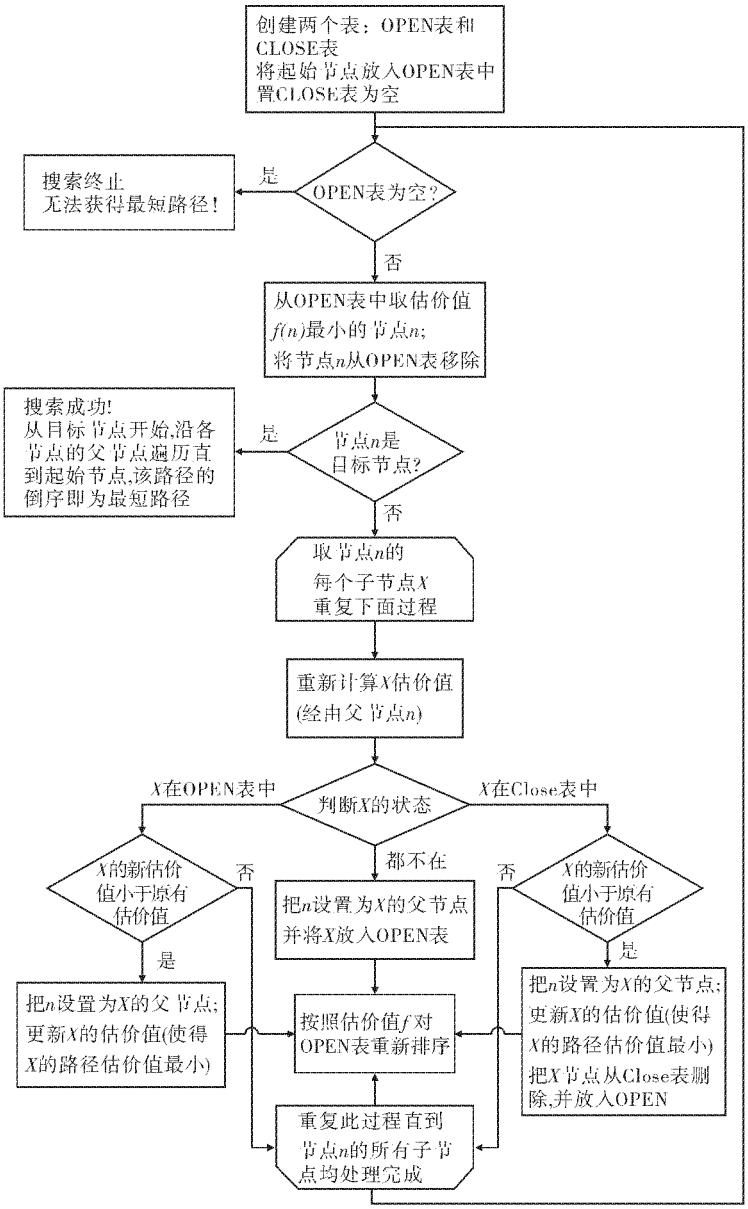


**图 1　A* 算法流程图**

Fig. 1　A* algorithm flowchart

## 1.2　优化分析

针对上述大规模网络的路径搜索需求，重点研究了应用 A* 算法进行路径搜索时的效率优化问题. A* 算法的数据结构的核心是两个表：Open 表(Open 表中存放已计算出估计值的结点)与 Closed 表(Closed 表存放已访问过的结点)，并且还没有访问过的 A* 结点. Open 表是 A* 算法访问最频繁的表，因为需要从 Open 表中取出最小的估计值，作为本次访问节点.

大量的操作是针对 Openlist 列表(Openlist 表存放可能成为最短路径的队列)进行的，重复不断的 OpenList 进行节点增加，删除，查找，按 $f$ 值排序等操作. 可以说 Open 表的数据结构设计很大程度上决定了算法的效率. 最易于实现 Open 表数据结构是链表结构，采用链表或者哈希表的数据结构来处理 Openlist. 这样算法的过程比较清晰，设计难度也较低. 但是当需要处理大规模的网络时，OpenList 的列表长度会增长到有数千个节点组成. 在重复对大规模的节点进行排序，搜索，操作，计算量会非常大. 其中最主要的是对 Openlist 的排序操作，如果使用常规的排序算法如冒泡法等，需要重复不断的遍历整个列表，算法复杂度是 $O(n2)$[2].

A* 算法并不需要 Openlist 完全有序，只需要找出 $f$ 值最小的节点即可，而对其他节点的位置无要求. 因此采用二叉堆可满足算法对 Openlist 的操作需求.

首先二叉堆是完全二叉树，并满足堆的特性：父节点的键值总是小于其子节点的键值. 最小键值的节点总在堆的顶端. 因此在算法中查找 $f$ 值

最小的节点，堆顶端的节点就是需要的节点. 用一维数组来表示二叉堆时，数组中第 $n(n>=1)$ 节点，其子节点位于数组的 $2n$ 和 $2n+1$ 的位置，其父节点位于数组 $n/2$ 的位置. 如图 2 所示(数组下标从 1 开始).

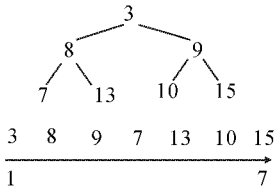


**图 2 二叉堆的数组表示**

Fig. 2 Binary heap array representation

显然数组第 1 个节点就是堆顶节点，就是 $f$ 值最小节点. 利用二叉堆的特性，可直接获取 Open 表中 $f$ 值最小的节点. 从 openlist 列表中取出 $f$ 值最小的节点，需要将其从堆移除. 在移除堆顶节点后，将堆的最后一个节点移动到顶点，再将新堆顶与其两个子节点比较，如父节点比子节点大，就交换二者，直到父节点比两个子节点的 $f$ 值都低.

当向二叉堆增加节点时，可将新节点放在数组末尾. 然后与其父节点比较，如果新节点的 $f$ 值更低，就交换父子的位置. 直到新节点不再比它的父节点低，或者这个节点已经到达顶端. 在搜索过程中，节点的 $f$ 值可能会改变，这时要重新对 Openlist 进行有序化处理：从发生改变的节点开始，将其与父节点比较，如果子节点的 $f$ 值小于其父节点，就将他们交换. 直到到达堆顶为止.

运用二叉堆可直接获取 Openlist 的 $f$ 值最小的节点，计算量与 Openlist 大小无关. 向 Openlist 插入节点，删除节点，排序操作的算法时间复杂度都是 $O(\log n)$. $n$ 为 Openlist 的节点数. 从理论分析可知，二叉堆对于大规模的 A* 路径搜索是效率比较优化的算法.

## 2 结果与讨论

应用 A* 算法进行最短路径搜索，用于测试的路网模型是一个拥有 100 万个节点(1000 * 1000)的超大型二维网格. 为了网络拓扑模型更加清晰化和结构化，网络中的每个节点(处于边缘的节点除外)都拥有 8 个不同方向的相邻节点，节点分为两类：可通行与不可通行. A* 算法需要在网格中指定的起始节点和目标节点之间，快速搜索出一条由可通行的节点组成的而且是代价最小的通行路径. 在测试网络中按照一定比例(30%)随机地设置的不可通行节点，当路径搜索遇到不可通行的节点时，是不能直接通行的，必须要绕过.

路径代价如图 3 中，节点与方向正对的四个相邻节点之间的路径代价为 10，与斜向方向的相邻节点的四个子节点的路径代价为 14. 这样的路网结构可以很好的模拟二维平面路网的拓扑结构和数学模型，同时尽量采用整形数值计算，降低运算难度，提高计算效率.

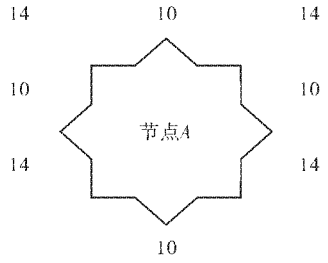


**图 3 网络结构模型**

Fig. 3 Network structure model

A* 算法中的关键估价函数值 $f(n)=g(n)+h(n)$，其中 $g(n)$ 表示为从节点 $n$ 到起始节点 $s$ 的已知最短路径长度. $H(n)$ 标准从节点 $n$ 到目标节点 $o$ 的路径估计值. 估价函数 $f(n)=g(n)+h(n)$ 代表经过该节点的路径的估计值，在 A* 算法运行中，$f(n)$ 值越小那么此节点的路径越好. 估价函数通过判断父节点和子节点的相对位置来进行累加计算，从起始节点开始逐步外推，直到到达目标节点位置.

对二维网络拓扑，图中节点由坐标 $(X,Y)$ 表示，那么节点 $A$ 到目标节点 $B$ 的启发函数可以用两点间最短路径的方法来设计，即

$$h(n)=\sqrt{(X_n-X_o)^2+(Y_n-Y_o)^2}$$

但考虑到计算机在处理开平方根运算会增大计算量，采用另外一个近似的公式取代

$$H(n)=\mathrm{abs}(X_n-X_o)+\mathrm{abs}(Y_n-Y_o)$$

其中，$X_n$，$Y_n$ 为节点 $n$ 的横向和纵向坐标值；$X_o$，$Y_o$ 为目标节点 $o$ 的横向和纵向坐标值.

abs()为绝对值函数. 图 4 为路网搜索的过程实例，图中每个像素点代表路网拓扑中一个节点，白色点为可通行节点，黑色点为不可通行节点. 深灰色点为搜索过的节点(Close 表)，白色点组成的线条即为所需的最短路径结果.

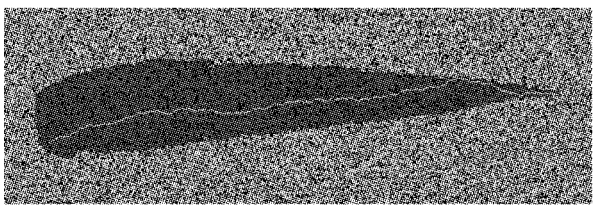

**图 4 网络搜索过程**

Fig. 4 Road network search process

通过常规的链表数据结构实现的 Openlist 和采用二叉堆进行优化后的 Openlist 的算法进行多次对比分析，结果如表 1 和图 5 所示.

**表 1　引入二叉堆算法后实际耗时对比**

Table 1　Introduction of binary heap algorithms compare the actual time-consuming

| | 最短路径节点数 | 总搜索空间（含重复搜索） | 常规算法耗时/s | 二叉堆算法耗时/s |
|---|---|---|---|---|
| 1 | 292 | 13 328 | 0.047 | 0.031 |
| 2 | 423 | 40 008 | 0.156 | 0.047 |
| 3 | 520 | 80 384 | 0.265 | 0.079 |
| 4 | 713 | 275 624 | 0.735 | 0.187 |
| 5 | 926 | 698 400 | 2.656 | 0.453 |
| 6 | 1 095 | 1 132 232 | 3.891 | 0.703 |
| 7 | 1 309 | 1 606 376 | 6.281 | 0.985 |
| 8 | 1 526 | 2 409 928 | 8.656 | 1.437 |

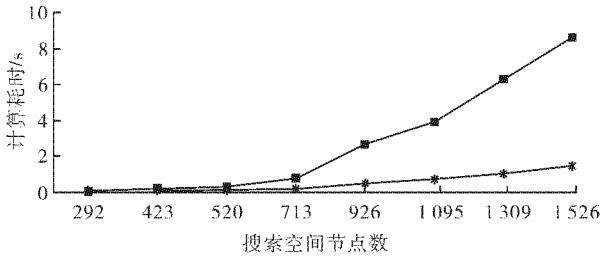


**图 5　引入二叉堆算法后实际耗时对比**

Fig. 5　Introducing binary heap algorithm actually consuming comparison chart

**注**：—■— 常规算法；—✱— 二叉堆算法

## 3　结　语

从两种算法的计算量和耗时对比可知，常规算法在进行长路径、大搜索空间的搜索时，计算时间迅速增加，而二叉堆算法则表现出良好的时间线性，没有出现搜索时间爆炸式增长的情况. 而针对数千个路径节点，在需要搜索几百万的海量节点空间的情况下，应用二叉堆数据结构的 A* 算法具有良好的时间线性度，具备工程应用价值.

## 致谢

论文在资料收集和实验数据采集过程中，得到黄珂副教授的大力支持，在此谨向她表示衷心的感谢.

**参考文献：**

[1] 陈伟亚，刘芳芳. 地理信息系统在水污染控制规划中的应用[J]. 武汉工程大学学报，2013，35(1)：21-25.
CHEN Wei-ya, LIU Fang-fang. Applieation of geographic information system teehnology in planing of water pollution control[J]. Jouranal of wuhan institute of teehnology, 2013,35(1):21-25. (in Chinese)

[2] 柳林，张继贤，唐新明，等. LBS 体系结构及关键技术的研究[J]. 测绘科学，2007，32(5)：144-146.
LIU Lin, ZhANG Ji-xian, TANG Xin-ming, at al. LBS architecture and Research of key technologies [J]. Science of Surveying and Mapping, 2007, 32 (5):144-146 . (in Chinese)

[3] 顾运筠. 最短路径搜索算法的几种优化改进[J]. 计算机应用与软件，2008，25(4)：246-247.
GU Yun-jun. Shortest path search algorithm several optimized and improved[J]. Computer Applications and Software,2008,25(4): 246 -247. (in Chinese)

[4] 刘浩，鲍远律. A* 算法在矢量地图最优路径搜索中的应用[J]. 计算机仿真，2008，25(4)：253-257.
LIU Hao, BAO Yuan-lv. The application of the A* algorithm searching the optimal vector path map[J]. Computer Simulation, 2008, 25 (4): 253-257. (in Chinese)

[5] 陈和平，张前哨. A* 算法在游戏地图寻径中的应用与实现[J]. 计算机应用与软件，2005，22(12)：118-120.
CHEN He-ping, Zhang Qian-shao. Application and Implementation of the A* algorithm pathfinding in game maps [J]. Computer Applications and Software, 2005, 22(12):118-120. (in Chinese)

[6] 史辉，曹闻，朱述龙，等. A* 算法的改进及其在路径规划中的应用[J]. 测绘与空间地理信息，2009，32(6)：208-211.
SHI Hui, CAO Wen, ZHU Shu-long, et al. A* algorithm and its application in path planning[J]. Geomatics & Spatial Information,2009,32 (6): 208-211. (in Chinese)

[7] 朱福喜，傅建明. 人工智能原理[M]. 武汉：武汉大学出版社，2002.
ZHU Fu-xi, FU Jian-ming. Principle of artificial intelligence[M]. Wuhan: Wuhan University press, 2002. (in Chinese)

[8] 龚劬. 图论与网络最优化算法[M]. 重庆：重庆大学出版社，2009.
GONG Qu. Graph theory and network optimization algorithms[M]. Chongqing: Chongqing University Press,2009. (in Chinese)

[9] 鲁统伟，林芹，李熹，等. 记忆运动方向的机器人避障算法[J]. 武汉工程大学学报，2013，35(4)：66-71.
LU Tong-wei, LIN Qin, LI Xi, et al. Obestacle avoidance algorithm of robot based on recording moue direction. Jouranal of Wuhan instute of technology,2013,35(4):66-71. (in Chinese)

# Practical analysis of improving path searching efficiency by heap sort

*SUN Yu-xin, ZHANG Jin*

(School of Computer Science and Engineering, Wuhan Institute of Technology, Wuhan 430074, China)

**Abstract**: With the development of computer technology, the method of path search algorithm has been widely used in many fields, and the higher request has been put forward for the searching time. To meet the requirement, based on artificial intelligence (Heuristic Search Methods A*), the method of heuristic search algorithm was adopted to improve searching efficiency, the search direction was adjusted dynamically by using network topology information and the algorithm was optimized to improve searching efficiency requirements by binary heap. Generally, the heuristic search algorithm is used for path search, its time complexity is O (n2) (n is the number of the network nodes), while dealing with the complicated network topology of millions of nodes, the searching time of heuristic search algorithm shows with exponential growth, so it does not meet the requirements of engineering technology. A good linearity of time is shown when the heuristic search algorithm of binary heap is applied for the long path and big search space through the theoretical analysis and experimental data, time complexity of this algorithm is O (log n) (n is the number of nodes Openlist). Meanwhile, there is no explosive growth in the searching time. So it meets the requirement of the higher performance and efficiency by using this algorithm, and has certain practical value for engineering practice.

**Key words**: path search; heuristic search methods; sort; binary heap

本文编辑:陈小平