

路径搜索策略研究

杨 泳, 严余松, 户佐安

(西南交通大学 交通运输与物流学院, 四川 成都 610031)

摘 要: 针对城市道路网车辆导航系统中经典 Dijkstra 最短路径搜索算法中存在的计算效率问题, 研究基于启发式策略和双向搜索策略的双向启发式优化搜索算法, 并探讨路网的分层搜索策略. 采用启发信息减少搜索范围、双向搜索分解搜索空间, 从而提高了算法的执行效率. 实际路网仿真结果表明: 相比经典 Dijkstra 算法, 启发式策略搜索效率可提升 70%~80%, 双向搜索策略在不损失搜索精度下进一步提高搜索效率 5%~10%, 而分层搜索策略可以极大提高大规模路网车辆导航长距离下路径搜索效率.

关键词: 最短路径规划; Dijkstra 算法; 启发式策略; 双向搜索策略; 分层搜索策略; 路网

中图分类号: TP18; U491

文献标识码: A

文章编号: 1000-7180(2013)10-0042-04

Path Searching Strategies Research

YANG Yong, YAN Yu-song, HU Zuo-an

(School of Transportation & Logistics, Southwest Jiaotong University, Chengdu 610031, China)

Abstract: Computational efficiency is widely recognized to be the critical issue in the shortest route searching algorithm on urban road traffic network vehicle navigation system using classic Dijkstra algorithm. Heuristic search strategy and bi-directional search strategy are studied, including the exploration with multi-level road networks search strategy. Heuristic information is utilized to limit the search area and bi-directional search method is proposed to decompose the search space, the efficiency of computing shortest path was improved and acceptable results were given with the algorithms. Comparative simulation results on traffic network indicate 70%~80% computation efficiency improvement by heuristic strategy against classical Dijkstra algorithm and continual 5%~10% efficiency gains without losing accuracy by bi-directional search strategy. Finally, multi-level search strategy indicates promising efficiency boost in long distance vehicle navigation system under real traffic networks.

Key words: shortest route searching algorithm; Dijkstra algorithm; heuristic strategy; bi-directional search strategy; multi-level search strategy; traffic network

1 引言

最短路径规划是车辆导航系统的基本问题, 网络中两点之间的最短路径规划问题可以归纳为图论中求解带权有向图的最优路径问题. 在车辆导航系统中, 最短路径规划算法用于计算车辆在原节点与目标节点之间最优行驶路径, 由于系统实时性要求高、实际路网的规模庞大, 而车载计算机的处理能力和系统的存储资源有限, 从而要求最短路径算法的

执行效率很高^[1].

在计算能力确定的前提下, 主要有三种优化策略^[2]: 存储数据结构优化、搜索算法优化及分层技术优化. 杨瑜君^[3]针对超大路网下 Dijkstra 最短路径问题展开研究, 指出采用分层搜索技术可以较大提高路径搜索效率, 文献[4-6]中基于 Dijkstra, A*, D* Lite 等算法提出了一些改进算法. 文献[7]中针对城市快速路的交通紧急事件给出了宏观的动态交通模型, 证明双向搜索算法的可行性、有效性. 大多

收稿日期: 2012-12-26; 修回日期: 2013-02-18

基金项目: 国家自然科学基金项目(61104175); 四川省软科学研究计划项目(2012ZR0022)

数最短路径规划研究都基于某种算法展开,只考虑网络的拓扑特征或阶段特征,从而忽略了网络的空间分布特征,对搜索策略研究缺乏系统性,难以融合各种搜索策略形成完备的车辆导航最短路径算法。

本文在经典 Dijkstra 算法的基础上,研究基于启发策略的 A* 算法及双向搜索策略的双向 A* 算法,几种算法的效率和精度对比表明了搜索策略的优越性。同时,对分层搜索策略进行了研究,表明分层搜索策略可以极大提高大规模路网车辆导航长距离下路径搜索效率。

2 算法描述

2.1 用堆实现路网存储结构

表示路网的数据结构有很多,例如邻接矩阵、邻接表、十字链表、队列,实际路网的结构属于典型的边稀疏网络,表示路网的数据结构中最有效的是二叉堆^[2]。Williams 在 1964 年提出了堆排序方法,该方法引入“堆”这种特定的数据结构。这里堆结构可以被视为一颗完全二叉树,可以作为高效的优先队列来使用。本文在计算过程中,采用优先队列与完全二叉树相结合的存储方式。

2.2 最短路径问题数学描述

最短路径问题是指在一个带权值的图中找出两个指定节点间的一条权值和最小的路径。数学上,将交通道路网络的拓扑关系模型记为赋权有向图 $G = (N, A)$ 。其中节点 $N = \{i\}$,节点数 $n = |N|$,弧集 $A = \{a \mid a = (i, j); i, j \in N\}$,弧数 $m = |A|$,对任意弧度 $a = (i, j)$,定义 c_a 或 $c_{i,j}$ 为路段权值(距离、行驶时间、行驶费用等),满足 $c_{i,j} > 0$ 。对 G 中任意路径 $P = (a_1, a_2, \dots, a_{l-1}, a_l)$,路径长度 $C(P) = c_{a_1} + c_{a_2} + \dots + c_{a_l}$ 为路径所经过的所有路段的弧长之和。任意给定两个节点 α, z 最短路径问题模型,描述为

$$\begin{aligned} & \text{Min}(c_{a,1} + \sum_{i=1}^{k-1} c_{i,i+1} + c_{k,z}), \\ & \text{s. t. } (a_{a,1}, \dots, a_{i,i+1}, \dots, a_{k,z}) \in A, \\ & k \in N, \\ & 0 < |k| < n. \end{aligned}$$

2.3 经典 Dijkstra 算法

Dijkstra 算法是求解最短路径问题的经典算法,Dijkstra 算法是一个按路径长度递增的次序产生最短路径的穷举算法,求解结果是原点到其它所有各节点的最短路径,搜索空间是以原节点为圆心

的圆(图1),算法比较简单,容易实现,适合小范围的最短路径的计算。

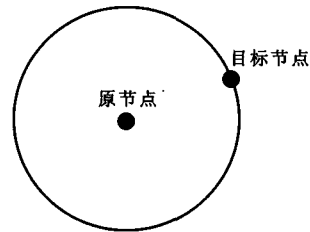


图1 Dijkstra 算法的搜索空间

Dijkstra 算法是已经被证明能得出最短路径的最优解,但它的效率是个问题,对于具有 n 个节点道路网,Dijkstra 算法的时间复杂度为 $O(n^2)$,一座大中型城市的路网节点数据就能达到几万个到几十万, Dijkstra 算法并不适合在大型交通网络中使用。

2.4 启发式搜索策略

启发式搜索是利用问题拥有的启发信息来引导搜索,以达到减少搜索范围,提高计算效率的目的。A* 算法的基本思想是引入启发函数 $h(n)$ 优先搜索最佳邻接路段节点,节点 n 的估价函数定义为 $f(n) = g(n) + h(n)$,式中 $g(n)$ 是开始节点到节点 n 的最小代价路径的代价, $h(n)$ 是节点 n 到目标节点之间代价的估计,称为启发函数, $f(n)$ 就代表节点 n 总的代价。本文 $h(n)$ 计算均采用曼哈顿(Manhattan)距离:

$$\text{Manhattan Dist}(A, B) = 2 * \pi * R * (|x_A - x_B| + |y_A - y_B|) / 360.$$

其中 R 为地球的半径,常量。

算法步骤描述如下:

1. 初始化。用链表数组存储交通网络图 G ;
2. 生成一个 Open 表和一个 Closed 表,把起点放入 Open 表;
3. 若 Open 表为空,则失败退出;
4. 选择 Open 表中 $f(n)$ 的节点,称为最佳节点,把它从 Open 表移入 Closed 表;
5. 若最佳节点是目标节点,获得解决方案,退出;
6. 扩展最佳节点,生成后续子孙节点均包含在 Open 表中;
7. 从小到大顺序重排 Open 表重新寻找最佳节点;
8. 返回第 3 步。

2.5 双向搜索策略

双向搜索就是将搜索过程分解成独立的两个搜索过程同时进行,即从原结点到目标节点正向搜索

过程和从目标结点向原节点的逆向搜索过程，其关键在于终止条件和切换条件的设定. 双向 A * 算法^[2]是结合双向搜索技术和启发信息提出的一种快速搜索算法，它由正向 A * 算法和逆向 A * 算法叠加进行. 正向过程启发算法的启发函数基于目标节点计算，而逆向过程函数则基于原节点计算.

理想状况下，正向搜索和逆向搜索在原节点和目标节点的几何中心相遇，这样可以减少 50% 的搜索空间，搜索空间如图 2 所示. 但在实际路网中，原节点和目标节点所处的路网密度、通达程度等均不同，搜索很可能不在中间点相遇，在极端情况下，甚至可能导致双向启发算法的搜索节点为单向启发搜索的两倍. 因此，设计一个好的双向搜索算法的关键是使其在搜索区域的中间部分相遇，而避免算法在中间部分不相遇，本文采用“迭代式最佳节点替换法”实现前向搜索和后向搜索切换来满足中间部分相遇和退出条件的要求.

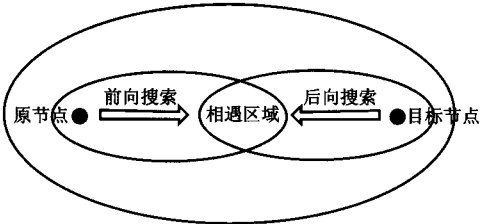


图 2 双向 A * 算法搜索空间

具体的改进方法是不使前向和后向搜索进行简单的交替切换执行，具体步骤如下：

- (1) 进行前向搜索，生成一个前向最佳节点.
- (2) 后向搜索以此前向最佳节点为目标节点，同时生成一个后向最佳节点，当执行前向搜索时以此后向最佳节点为目标节点.
- (3) 反复迭代，直至相遇退出.

前向和后向之间的切换取决于搜索空间最佳节点的估价函数，如果在前向搜索中基于目标节点的当前节点的估价函数小于后向搜索基于原节点的当前节点的估价函数，这就意味着在前向搜索中更偏向“中间区域”，此时执行后向搜索，反之，则执行后向搜索，这样就能保证前向和后向搜索中间区域相遇的理想情况.

2.6 分层搜索策略

在实际的路网中，道路是分等级的，而这种等级的区分在路径优化过程中发挥了很大的作用. 经验表明，在起讫点距离较远时，用户往往优先确定高等级道路的路线，然后确定起讫点与高等级路线之间的衔接路线进而完成对整个路线的确定. 文献[8]中

通过改变权值让高等级道路被优先选择，然而这样的诱导算法并没有提高诱导的效率，而且往往会产生在不同等级道路之间“上窜下跳”的现象. 为此本文提出道路网数据物理分层的观点，即将路网根据道路的等级分成不通的层次，高层路网是对原有路网的抽象，是原有路网节点的一个子集，层次越高，所包含的节点和路段数越少. 在基于分层路网的基础上，构造分层的诱导路线策略，可以提高诱导效率，吻合用户长距离出行的思想.

3 结果对比分析

为了验证搜索算法的有效性，我们在 Windows 操作系统平台下基于 .net 技术和 Visual Studio 工具开发小型 GIS 平台，能够显示、操作、分析道路网数据. 在此基础上，用基于面向对象的技术集成路径搜索算法在路网进行验证分析. 本文分别按经典 Dijkstra 算法、双向 A * 算法搜索最优路径. 实际测试数据选取成都市绕城以内道路网数据，共有 39 446 个节点和 28 067 条路段. 仿真程序的运行环境为 Windows server 2008R2，Intel 2.8 GHz 四核处理器，4 GB 主存，512 GB 硬盘. 测试分析分为两部分：(1) 测试算法的计算效率；(2) 测试算法结果的合理性.

3.1 算法计算效率对比

在路网上随机选取 10 个起点和终点对 (OD 对)，分别利用 Dijkstra 算法、A * 算法和双向 A * 算法进行最优路径搜索，利用 CPU 时间来衡量算法的效率. 由于计算时间较短，为消除和减少误差，对每组 OD 对分别计算 10 次，并取其平均值作为计算时间，结果如表 1 所示.

表 1 成都市道路网随机 10 组 OD 对计算时间 s

路径	CPU 时间		
	Dijkstra 算法	A * 算法	双向 A * 算法
1	0.265	0.029	0.022
2	0.321	0.069	0.066
3	0.065	0.015	0.011
4	0.314	0.05	0.031
5	0.231	0.074	0.054
6	0.265	0.033	0.031
7	0.232	0.031	0.035
8	0.263	0.02	0.022
9	0.203	0.034	0.021
10	0.178	0.026	0.013

可以看出，随着 OD 对距离的波动，搜索时间呈现相同趋势的波动. 对于同一 OD 对，搜索算法的平

均计算时间明显小于 Dijkstra 算法的平均计算时间,且不同的 OD 之间提高幅度有所区别,平均提高 70%~80%左右,如图 3 所示.而双向搜索算法在效率上更具优势,约提高 5%~10%,只在极少情况下双向搜索时间比 A* 算法高.

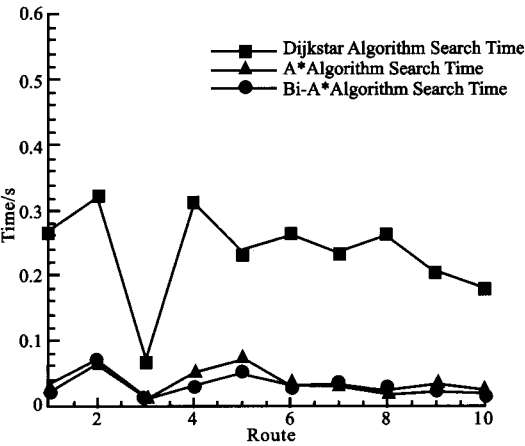


图 3 Dijkstra、A*、双向 A* 算法效率对比

3.2 算法计算结果对比

首先对比路网上 10 组 OD 对两种不同算法之间的搜索路径结果,如图 4 所示,可以看出算法规划出来的路径长度都比 Dijkstra 算法规划的路径长度稍长,但吻合程度很高,说明搜索算法计算出的路线是较好的“次优路”,而双向 A* 算法在计算结果略微比 A* 算法略长.

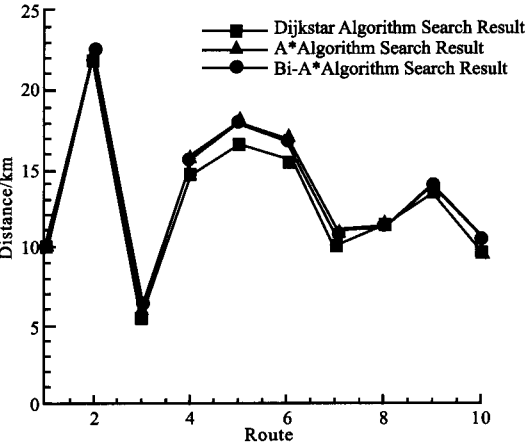


图 4 Dijkstra、A*、双向 A* 算法结果对比

其次,进一步量化搜索路径精度,随机选取 1 000 组 OD 点,分别使用经典 Dijkstra 算法、A* 算法和双向 A* 算法进行最优路径搜索计算,结果如表 2 所示.以 Dijkstra 最短距离为基准,采用 A* 算法进行计算平均会有 6.02% 的误差,采用双向则有 6.03% 的误差.对于最长距离 33.55 km 的结果,分别产生 3.21 km 和 3.32 km 的偏差,均能满足出行者的出行要求.

表 2 成都市道路网随机 1 000 组 OD 点结果对比 km

	路径长度		
	Dijkstra	A *	双向 A *
平均值	13.849	14.69	14.70
路径最小值	1.088	1.088	1.088
路径最大值	33.55	36.76	36.87

3.3 分层计算结果对比

本文采用两级分层:底层路网包括路网中所有的道路和交叉口,而顶层路网只包含主干道和主干道形成的交叉口.本文采用两级分层后,高等级道路数 112 条,是底层数据(28 067 条)的 0.4%,在高等级道路的路径搜索时间可以忽略不计.

通过大量测试表明,在长距离大范围内采用分层搜索策略效果非常明显,路径往往优先选择高等级公路,其搜索时间至少可以下降至 20%~30%,符合实际预期.在测试结果分析过程中,也发现一些问题,例如高等级入口不是最优路口、绕“路”进入高等级路网等现象,这将是本文进一步研究的重要课题.

4 结束语

结合经典 Dijkstra 算法、A* 算法和双向 A* 算法搜索可以发现,启发式策略在运算时间上具有明显优势,可使计算效率平均提高 70%~80%,规划产生的“次优路径”也能较好的满足出行者的出行要求,具有一定的实用价值;采用双向搜索策略,计算效率上有一定幅度的提升,计算结果变化并不明显,可以预计在更大规模路网及动态条件下,双向搜索策略优势更为明显.实际中,由于司机主观偏好及路网信息认可程度等因素的影响,本策略也为研究“多准最优路径”问题提供新的研究思路.本文提出的分层启发式算法新思路可以实现长距离下快速搜索路径,这对于实现实时动态路径诱导的智能交通领域具有非常好的应用前景,但如何提高其准确性将是本文进一步研究的重点方向之一.

参考文献:

[1] 张起善. 智能车辆定位导航系统及应用 [M]. 北京: 科学出版社, 2002.

[2] Fu L, Sun D, Rilett L R. Heuristic shortest path algorithms for transportation applications: state of the art [J]. Computers & operations research, 2006, 33(1): 3324-3343.

(下转第 49 页)

