文章编号:1001 - 9081(2008)04 - 0951 - 03

复杂网络中最短 K 条路径问题的求解算法研究

刘　佳[1],夏少芳[2],吕亚男[2],陈立潮[2]
(1. 石家庄铁路职业技术学院 信息工程系,石家庄 050041； 2. 太原科技大学 计算机科学与技术学院,太原 030024)
(pplive518@126. com)

摘　要:以时间代价作为目标函数,针对复杂网络的优化问题进行研究,给出了目标评价函数模型的建立过程,提出了基于改进的 A * 算法求解复杂网络中最短 K 条路径问题的算法,并以城市交通为例,对算法进行了验证。实验结果表明所提出的算法可适用于一般多重图中最短 K 条路径问题的快速求解,具有广泛的应用价值。

关键词:多重图;A * 算法;最短 K 条路径
中图分类号: TP301.6　　**文献标志码:**A

Algorithm for solving K-shortest paths problem in complicated network

LIU Jia[1], XIA Shao-fang[2], Lü Ya-nan[2], Chen Li-chao[2]
(1. *Department of Information Engineering, Shijiazhuang Institute of Railway Technology, Shijiazhuang Hebei* 050041, *China*;
2. *Institute of Computer Science and Technology, Taiyuan University of Science and Technology, Taiyuan Shanxi* 030024, *China*)

Abstract: Focusing on the optimization problems about complicated network, an algorithm named KSPA (K-Shortest Paths based on A *) was proposed to solve the K-shortest paths problem in complicated network. The time cost was taken as target function and the establishment of the target function model was given. Experimental results show the KSPA algorithm proposed can be used to solve the K-shortest paths problems quickly in multi-graph.

Key words: multi-graph; A * algorithm; K-shortest paths

0　引言

最短路径问题是计算机科学与地理信息科学等领域的研究热点问题,人们继 Dijkstra 开创性的工作后提出了大量求解最短路径问题的算法[1-7]。实际上,在一些决策支持系统和咨询系统中,用户除了希望得到最短路径以外,还希望得到次短和第三最短路径作为决策参考[8],由此产生了求解第 K 最短路径问题。目前一些简单网络图是 K 条最短路径搜索问题的主要研究对象,因此出现的相关算法也都是针对简单网络的。David Eppstein 提出了一种在有向连通图中搜索第 K 最短路径的算法;Richard Bellman 与 Bobert Kalaba 从控制论的角度,通过一种离散过程的优化方法来计算给定两点之间的最短 K 条路径;文献[9-11]利用"背离路径"的性质给出了计算 K 条最短路径的递推公式,只是在求解第 K 最短路径时分别采用了不同的方法。然而,以上大部分算法本身比较复杂难以实现,时间复杂度不太理想,而且仅对网络节点个数较小的简单网络适用。

计算机网络与通信、分布式处理和智能交通系统的兴起给这个传统的研究课题带来了新的转折,这些要求解决最短路径问题的领域都以复杂网络图的结构呈现。因此,对复杂网络中最短 K 条路径问题求解算法的研究具有重大的意义。本文以城市公共交通为例,介绍复杂网络中最短 K 条路径问题的求解算法。

1　算法模型的建立

1.1　空间数据结构的建立

公共交通网络空间数据结构的建立主要是道路和停靠站点数据结构的建立。道路网是一个网状结构,道路数据结构采用拓扑型矢量表示法。整个道路网络可以看成是由多段道路连接而成,连接点即为道路和道路的交叉点。以道路和道路的交叉点为节点,节点和节点之间的道路为链,可建立链 - 节点式的拓扑数据结构,曲线道路可由多段道路进行拟合。每段道路由两个交叉点确定,且每段道路和每段道路只在端点处相交。公交停靠站点以点的形式存在,类似于道路交叉点,但是停靠站点必须位于某段道路上,而不能与道路的某个交叉点重合。道路及其交叉点和停靠站点组成的网络如图 1 所示。

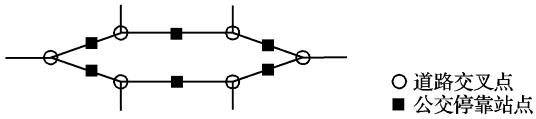


图 1　道路及其交叉点和停靠站点组成的网络

1.2　时空搜索模型的建立

最优换乘方案数学模型的评价函数规定为乘客从出发点到目的地所花费的时间最少。在决定换乘所需时间的诸因素中和时间有关的因素为首发车和末班车时间、发车间隔以及车速。发车间隔分为高峰间隔、平峰间隔和晚间间隔,相应的车速分为高峰车速、平峰车速和晚间车速。这样在一个确定的时刻,即用户的咨询时间,总可以确定有哪几路车正在运营及这些车在该时刻的发车间隔和车速。因此,在某个时刻 t_0 最短路径数学模型的目标函数 f 为:

$$\min f(p) = \sum_{i=1}^{n}\left(h_k + \frac{d_{kj}}{s_{kj}} + \Psi\right)$$
$$\text{s.t. } t_1 \leqslant t_0 \leqslant t_2,\ N_{\min} \leqslant n \leqslant N_{\max},$$
$$k, j \in Q,\ Q \in p,\ p \in L \qquad (1)$$

其中 $f(p)$ 是从出发点沿着路径 p 到达目的地所需的时间, n

收稿日期:2007 - 11 - 02;**修回日期:**2007 - 12 - 10。　**基金项目:**山西省自然科学基金资助项目(20051044)。
作者简介:刘佳(1980 -),男,河北石家庄人,硕士研究生,主要研究方向:人工智能；　夏少芳(1983 -),女,河北邢台人,硕士研究生,主要研究方向:人工智能；　吕亚男(1981 -),女,山东淄博人,硕士研究生,主要研究方向: 人工智能；　陈立潮(1961 -),男,山西万荣人,教授,博士,主要研究方向:模式识别、人工智能、数据挖掘。

是总换乘次数，h_k 是第 i 次换乘时在节点 k 的平均候车时间，可以认为是第 i 次换乘时公交车在 t_0 时刻的发车间隔。d_{kj} 是公交网络中 k，j 两点之间的路程，即第 i 次换乘时公交车所走的距离。s_{kj} 是公交网络中 k，j 两点之间的车速，即第 i 次换乘时公交车在 t_0 时刻的速度。Ψ 是常数，因为每次换乘可能会遇到许多不可知因素，如过马路需要绕行天桥等。所以每次换乘除了所需的换乘时间外，还应额外添加一段时间。t_1 是所有可能换乘公交车的最晚首发车时间。t_2 是所有可能换乘公交车的最早末班车时间。$N_{\min}$ 是最少换乘次数。$N_{\max}$ 是最多换乘次数。k 是第 i 次换乘时的起点。j 是第 i 次换乘时的终点。Q 是公交线路中的节点，由公交站点抽象而成。L 是出发点和目的地之间的可通行路径集。p 是 L 中的一条可行路径。由以上分析可知，最短路径的搜索不但是空间的搜索，而且要和具体的搜索时间结合，是一个同时涉及时间和空间的问题。

1.3 模型分析

站点间的换乘有两种可能，一种是从这路车换乘另外一路车，一种是从坐车换成步行或从步行换成坐车。两次换乘都是从一个站点到另一个站点，这样的两个站点或者坐同一路车可以到达，或者步行可以到达。如果将所有站点看作是节点，站点之间坐同一路车或者步行可以到达看作一条有向边，所花费的时间看作边权，则 t_0 时刻的公共交通状态便形成了一个网络。因为站点与站点之间可能有多种到达方式，所以该网络是一个不规则的多重图[12]，如图 2 所示。

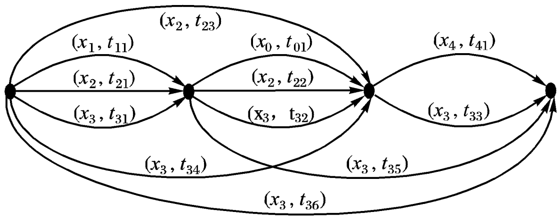


图2　公共交通多重图

图2 中有序对(x_s, t_{kj}) 表示乘客在第 i 次换乘时所花费的时间为 t_{kj}，即为多重图中某条有向边的权值，x_s 表示乘客第次换乘时选择的换乘方式（$x_s \geqslant 1$ 表示公交车号，$x_s = 0$ 表示步行，x_s 值相同说明乘坐的公交车相同）。由此可知，时空搜索模型的解即为该多重图中某两点之间的最短路径。

2　算法的设计与实现

通过以上分析，对模型的求解可以分成两步：第一步是确定图 2 所示的公共交通多重图，即确定哪些节点之间有边，有几条边，边的权值是多少；第二步是求所确定的多重图中由出发节点到目的节点的最短 K 条路径。

1）公共交通多重图生成

图2 所示公共交通多重图中节点间的有向边通过两种方式形成：一种是乘坐同一路车可以达到；一种是步行可以达到。

通过检索各路公交车行走路线（往返相反方向的同一路车按两路车进行处理）所经过的站点，可以确定哪些站点节点之间通过乘坐同一路车可以到达，生成有向边，边权为 $h + d/s + \Psi$，其中 h 表示在 t_0 时刻该路公交车可能的等候时间，即发车间隔；d 表示按照该路车的行车路线两车站之间的距离；s 表示该路公交车在 t_0 时刻的车速。

通过步行从一个站点走到另一个站点的距离如果不大于某一阈值(δ)，则这两个站点相互步行可到达，可以生成两条有向边，边权 $= \dfrac{\text{步行距离}}{\text{步速}} + \Psi$。然而站点位于道路上，步行也必须沿道路行走，所以两站点间是否步行可以到达是无法直接求出的。对此问题，用图 1 就可以解决：站点位于道路上，道路由两个交叉点确定，所以可由两点间距离公式直接计算出站点到相应两个道路交叉点的距离。将所有的站点和道路交叉点看作节点，节点间的权值为节点之间的距离，这样整个道路和站点便构成一个赋权无向图。在图 1 的基础上就可以运用 Dijkstra 算法求解任意两个公交站点间的最短路径，如果两个站点间的最短路径小于给定的阈值，就认为这两个站点相互步行可以到达，否则两个站点间就不能生成有向边。

2）最短 K 条路径问题求解

鉴于公共交通多重图的特点，本文采用人工智能领域里面的 A＊算法[12]求解最短 K 条路径。启发式搜索 A＊算法是用于路径搜索和规划的经典方法，该算法简单，可以利用已有的知识和数据进行启发搜索，比较灵活。但它在搜索过程中暴露的不足却很明显，搜索空间越大，耗时越多，以及由于估计函数设计的不合理不能充分利用求解过程中的启发信息等。在 A＊算法中只要 $h(x) \leqslant h^*(x)$，就一定能找到最短路径，在一般情况下所设计的启发函数 $h(x)$ 为两个节点间的欧式距离，这种启发函数往往远远低于实际的最短路径，尽管保证了算法的可纳性，却影响了算法的搜索效率。从 A＊ 算法的性质可以看到启发函数 $h(x)$ 的值越大，搜索的效率就越高。因此为了突出启发信息的重要性，我们对估计函数进行如下改进：

$$f(x) = \alpha g(x) + \beta h(x) \tag{2}$$

对于任意一个节点 j，则有：

$$f(j) = \alpha g(j) + \beta h(j) \tag{3}$$

启发函数 $h(x)$ 设计如下：

$$h(x) = T + \rho\varphi \tag{4}$$

对于任意一个节点 j，则有：

$$h(j) = T + \rho\varphi_{ij} \tag{5}$$

其中 α、$\beta(0 < \alpha, \beta \leqslant 1,\ \alpha + \beta = 1)$ 为两个参数因子，实验中设定 $\beta \in (0.6,\ 0.9)$。两个参数分别用于调整 $g(x)$ 和 $h(x)$ 在估计函数中所占的比例。$g(j)$ 为从出发点 S 到 j 点实际所花费的时间，可以看到 $h(j)$ 为两部分之和；T 为从 j 点到终点 E 的直线距离 D_{jE} 除以最大车速 $v_{\max}$ 得到的时间，$T = \dfrac{D_{jE}}{v_{\max}}$；$\varphi_{ij}$ 为两条直线确定的夹角，取弧度值。其中一条直线由出发点 S 与终点 E 确定，另一条直线由节点 i 与节点 j 确定。其中节点 i 为当前所扩展的节点，节点 j 为节点 i 的后继节点。$\rho(0 < \rho \leqslant 1)$ 为常数因子，用于调整 φ_{ij} 在 $h(j)$ 中所占的比例。$f(i)$ 为从起点 S 经过节点 i 到达终点 E 所花费时间的估计值，所以对于任意一个节点 i，它的任意一个后继节点 j 的估价函数的值为：

$$f(j) = g(j) + h(j) \Rightarrow f(j) = \alpha g(j) + \beta(T + \rho\varphi_{ij}) \tag{6}$$

最短路径的子路径也是最短路径，也就是说由改进的 A＊算法求得的第一条最短路径中必定包括某些节点序列或者某些子路段在第二、第三甚至第 K 最短路径中。利用这个性质可以来求解最短 K 条路径。在求解之前，通过对实际问题的分析，可以做如下的规定（以后称为条件 R）：

1）最多换乘次数为 4 次；

2）换乘车次不能重复；

3）相邻两次换乘都不能为步行；

4）换乘站点不能重复。

由于在公共交通多重图中求解最短 K 条路径，因此这里约定 $arc(i, j, v)$ 表示起点为 i，终点为 j，权值为 v 的弧段，使用"＜＞"符号表示连接两个节点序列；使用 $next(i, R)$ 表示在满足条件 R 的情况下 i 的后继邻接点集合；$succeed(j, P)$ 表示在路径 P 中 j 的后继节点；使用 $path(S, E, P)$ 表示沿路

径 P 从 S 节点到 E 节点的节点序列, $path(S, j, P <> path(j, i, P) = path(S, i, P)$;使用 $path\_wasf(S, E, G)$ 表示通过 A * 算法得到的在多重图 G 中从 S 到 E 的最短路径, S 表示起点, E 表示终点。另外约定使用 $current\_path$ 表示当前分析路径; $result\_path$ 表示已经得到的最短路径的集合; $optional\_path$ 表示候选路径的集合,是所有最可能成为下一条较优路径的集合。

求最短 K 条路径算法(K-Shortest Paths based on A *, KSPA)的步骤如下:

1) $optional\_path = \varnothing$, $current\_path = path\_wasf(S, E, G)$, $result\_path = result\_path \cup current\_path$, $k = 1$(用整数 k 表示已经得到的第 k 最短路径)。

2) 如果 $current\_path = \varnothing$ 或者 $k \geqslant K$ 则算法结束,退出;否则转步骤 3)。

3) 按顺序从 S 到 E 遍历 $current\_path$ 中所有节点,这里把当前遍历到的节点称为当前分析点,记为 j。

4) 针对当前分析点 j,如果 j 为终点,则转步骤 7),否则考察 $next(j, R)$ 集合。对于任何一个 $m \in next(j, R)$,判断是否满足 $(path(S, j, current\_path) <> arc(j, m, v)) \in (result\_path) \cup optional\_path)$,如果条件成立,就从图 G 中暂时删除 $arc(j, m, v)$,以保证不会再次得到当前分析路径作为候选路径。

5) 利用前面设计的 A* 算法计算从 j 到终点的最短路径 $path\_wasf(j, E, G)$,如果 $path(j, E, path\_wasf(j, E, G))$ 不为空,则 $optional\_path = optional\_path \cup (path(S, j, current\_path) <> path(j, E, path\_wasf(j, E, G)))$,不管 $path(j, E, path\_wasf(j, E, G))$ 是否为空,从 G 中暂时删除所有 $arc(j, m, v)$, $m \in next(j, R)$。

6) $j = succeed(j, current\_path)$,然后转步骤 4)。

7) 从候选路径 $optional\_path$ 中取出权值最小的一条路径作为 $current\_path$,执行 $result\_path = result\_path \cup current\_path$, $optional\_path = optional\_path - current\_path$, $k = k + 1$。

8) 恢复所有已经删除的弧段,然后转步骤 2)。

3 算法验证

表 1　两种算法结果比较

OD 对	传统的 A * 算法 $h(x)$ = 两点间欧氏距离/ v_{max}		改进的 A * 算法 $\alpha = 0.13, \beta = 0.87, \rho = 1$		找到的路径是否相同
	花费时间/s	扩展的节点个数	花费时间/s	扩展的节点个数	
1	0.5684	123	0.1371	28	相同
2	1.6052	312	0.1373	96	相同
3	0.1368	27	0.1370	27	相同
4	2.2445	443	0.1470	153	相同
5	2.3205	458	0.1471	175	相同
6	0.1013	20	0.1280	12	相同
7	2.6047	476	0.1472	188	相同
8	2.6397	521	0.1472	222	相同
9	2.9691	586	0.1491	326	相同
10	2.7056	534	0.1472	254	相同
平均	1.7866	350	0.1424	148	

为了验证本文提出的模型及算法的有效性,这里对基于 A * 算法求解公交网络中最短 K 条路径问题由 Delphi 7.0 在 Athlon 3000 + 环境下实现,算法的步骤如 KSPA 算法所描述。所采用的数据是基于城市公共交通网络的,其中包括将近 90 条公交线路,大约 400 个公交站点;用本文提出的模型进行描述转化成的网络包括 13 847 条边。在城市的公共交通网络中,随机选取了若干 OD 节点对进行实验。

表 1 是分别运用 A * 算法和改进的 A * 算法求解 OD 对间的最短路径(第一条最短路径)相同的情况下所花费的时间和扩展的节点数之比。

在同样的条件下,运用 KSPA 算法求解它们之间最短 K 条路径($K = 3$),然后再运用基于动态剪枝技术的深度优先搜索算法进行求解,这两种算法所花费的时间如表 2 所示。

表 2　两种算法结果比较

OD 对	花费时间/s 基于动态剪枝技术的深度优先搜索算法	花费时间/s KSPA 算法 $\alpha = 0.13, \beta = 0.87, \rho = 1$	找到的路径是否相同
1	4.2259	0.1870	相同
2	5.5281	0.1871	相同
3	4.1460	0.1671	相同
4	4.1559	0.1881	相同
5	5.0569	0.1970	相同
6	4.1461	0.1570	相同
7	6.1089	0.2080	相同
8	5.2469	0.1970	相同
9	6.7089	0.2171	相同
10	5.6280	0.2060	相同
平均值	5.0952	0.1965	

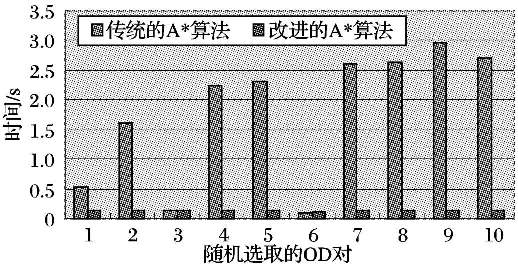


图 3　传统的 A * 算法与改进的 A * 算法时间对比

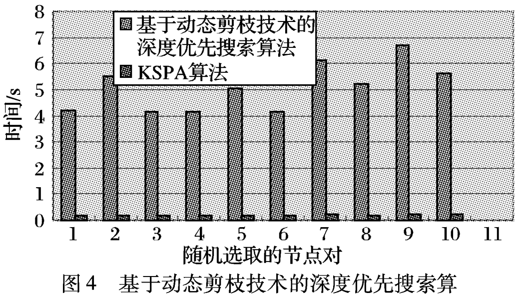


图 4　基于动态剪枝技术的深度优先搜索算法与 KSPA 算法($K = 3$)时间对比

从表 1 和表 2 可以看到,改进后的 A * 算法的效率远远高于改进前的 A * 算法,基于改进的 A * 算法而设计的求解最短 K 条路径的 KSPA 算法的时间效率远远高于深度优先搜索的时间效率,具有更广泛的应用价值。

4 结语

本文在改进的 A * 算法及 KSPA 算法的基础上,开发了城市智能公交查询系统,并以某市的城市公共交通数据进行验证,实验结果证明了改进的 A * 算法及 KSPA 算法的正确性及有效性。本文提出的算法可以快速地求解一般复杂网络中的最短 K 条路径问题,具有广泛的应用价值。

(下转第 956 页)

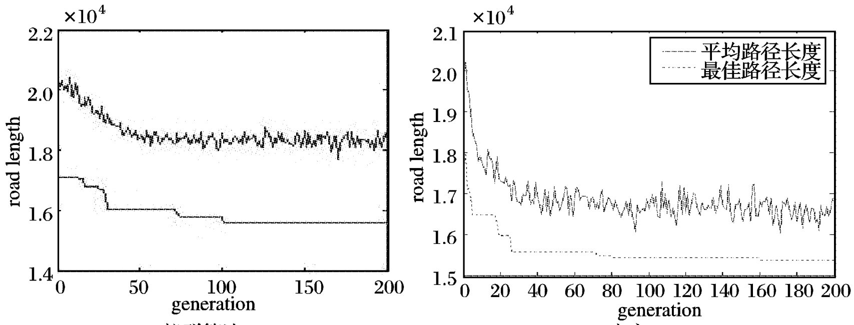
论,我们将改进前后的算法分别称为算法 1 和算法 2。当城市个数为 20 时,两种算法每次都能求得最优解,但算法 2 的收敛速度明显快于算法 1 的收敛速度;城市个数为 31 时,算法 2 有 3 次求得最优解 15 377.711 333,9 次求得次优解 15 380.515 324,相比之下,算法 1 分别只有 2 次和 5 次;对于 72 个城市的 TSP,算法 2 也明显优于算法 1,平均收敛代数不到算法 1 的二分之一。

表 1　算法改进前后各参数的设置

阶段	L_{size}	P_o	P_c	sL_{size}	$sGen$	sPr	sP_c	sP_m
改进前	8	0.15	0.7	3	10	0.15	0.0	0.15
改进后	8	0.15	0.7	3	10	0.15	0.9	0.05

表 2　算法改进前后求解效率的对比

城市个数	改进前			改进后		
	最优解	平均最优解	平均收敛代数	最优解	平均最优解	平均收敛代数
20	24.522 234	24.522 234 0	21	24.522 234	24.522 234 0	10
31	15 377.711 333	15 428.600 270 0	90	15 377.711 333	15 418.100 340 0	61
72	498.902 757	500.004 821 4	62	498.902 757	499.173 221 8	27



(a) 蚁群算法　　(b) 本文MAGA

图 4　蚁群算法与本文 MAGA 收敛曲线对比

从结果可以看出,与原算法相比,本文中的多智能体遗传算法在收敛速度和稳定性上都有较大提高,说明在邻域正交交叉算子中采用精英保留策略以及在自学习算子中引入交叉算子加快收敛速度的做法是行之有效的。

2.2　与其他算法的对比

在这个实验中,分别将本文中的多智能体遗传算法与 PBIL算法(城市个数为72)[5] 和蚁群算法(城市个数为 31)[6] 进行对比,各参数设置同上,结果如表 3 和图 4 所示。从表 3 可以看出,多智能体遗传算法在稳定性上明显好于 PBIL 算法,几乎每次都能找到最优解;而图 4 表明多智能体遗传算法的收敛速度要快于蚁群算法,而且得到的最优解(15 377.711 333)也好于蚁群算法的最优解(15 602)。

表 3　PBIL 算法与本文 MAGA 求解效率对比(城市个数为 72)

指标	PBIL 算法	本文 MAGA
最优解	498.902 9	498.902 757 0
平均最优解	502.783 1	499.173 221 8

3　结语

通过上面的实验分析可以看出,由于多智能体遗传算法借鉴了多智能体系统中智能体之间的竞争、合作和自学习行为,使得它具有很好的稳定性和很高的寻优效率,而改进后的多智能体遗传算法在性能上又有进一步的提高,在求解 TSP 等组合优化问题时明显优于原算法,具有很高的实用价值。

参考文献:

[1] 钟伟才,薛明志,刘静,等. 多智能体遗传算法用于超高维函数优化[J]. 自然科学进展, 2003, 13(10): 1078 - 1083.

[2] 薛志明,钟伟才,刘静,等. 用于函数优化的正交 Multi-Agent 遗传算法[J]. 系统工程与电子技术, 2004, 26(9): 1305 - 1311.

[3] 王小平,曹立明. 遗传算法——理论、应用与软件实现[M]. 西安: 西安交通大学出版社, 2002: 123 - 129.

[4] 李程俊,张求明. 求解 TSP 问题的多线程演化算法[J]. 计算机工程与设计, 2005, 26(7): 1744 - 1746.

[5] 金炳尧. 最优化计算中的若干新技术[J]. 科技通报, 2000, 16(2): 119 - 124.

[6] aiwa. 蚁群算法 TSP(旅行商问题)通用 matlab 程序[EB/OL]. [2007 - 06 - 20]. http://bbs.matwav.com/post/view?bid = 7&id = 529956.

(上接第 953 页)

参考文献:

[1] ZWICK L R. Replacement paths and k simple shortest paths in unweighted directed graphs[C]// LNCS 3580. Berlin: Springer, 2005: 249 - 260.

[2] COFFMAN T, GREENBLATT S, MARCUS S. Graph based technologies for intelligence analysis[J]. Communications of ACM, 2004, 47(3): 45 - 47.

[3] CHENY-L, YANG H-H. Finding the first k shortest paths in a time-window network[J]. Computers & Operations Research, 2004, 31(4): 499 - 513.

[4] SUBRAMANIAN S, MUTHUKUMAR V. Alternate path routing algorithm for traffic engineering in the Internet[C]// Proceedings of ITCC. Washington, DC, USA: IEEE Computer Society, 2003: 367 - 372.

[5] TAKAYUKI G, TAKESHI K, HIROSHI N. On the heuristics of a or A∗ algorithm in ITS and robot path-planning[C]// Proceedings of the 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems. Las Vegas, Nevada: IEEE Press, 2003: 1159 - 1166.

[6] LI JUN-JIE, ZHANG HAN-YI. A new solution to the k-shortest paths problem and its application in wavelength routed optical networks[J]. Photonic Network Communications, 2006, 12(3): 269 - 284.

[7] HOCEINI S, MELLOUK A, AMIRAT Y. K-shortest paths Q-routing: A new QoS routing algorithm in telecommunication networks[C]// LNCS 3421. Berlin: Springer, 2005: 164 - 172.

[8] DIAS L C, CLIMACO J. Shortest path problems with partial information: Models and algorithms for detecting dominance[J]. European Journal of Operation Research, 2000, 121(1): 16 - 31.

[9] HOFFMAN W, PAVLEY R. A method for the solution of the nth best path problem[J]. ACM, 1959, 6(4): 506 - 514.

[10] 袁红涛,朱美正. K 优路径的一种求解算法与实现[J]. 计算机工程与应用, 2004, 40(6): 51 - 53.

[11] 王明中,谢剑英,陈应麟. 一种新的 Kth 最短路径搜索算法[J]. 计算机工程与应用, 2004, 40(30): 49 - 50.

[12] 张永梅,韩焱,陈立潮. 城市公交查询系统的研究与设计[J]. 计算机应用, 2005, 25(2): 422 - 425.