软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2012,23(Suppl.(2)):204−212

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

基于匹配度聚类算法的单车辆合乘问题*

邵增珍[1,2+], 王洪国[1], 刘 弘[1,2], 孟春华[3]

[1](山东师范大学 信息科学与工程学院,山东 济南 250014)

[2](山东省分布式计算机软件新技术重点实验室,山东 济南 250014)

[3](山东师范大学 管理科学与工程学院,山东 济南 250014)

Single Carpooling Problem Based on Matching Degree Clustering Algorithm

SHAO Zeng-Zhen[1,2+], WANG Hong-Guo[1], LIU Hong[1,2], MENG Chun-Hua[3]

[1](Institute of Information Science and Engineering, Shandong Normal University, Ji'nan 250014, China)

[2](Shandong Provincial Key Laboratory for Distributed Computer Software Novel Technology, Ji'nan 250014, China)

[3](Institute of Management Science and Engineering, Shandong Normal University, Ji'nan 250014, China)

+ Corresponding author: E-mail: shaozengzhen@163.com, http://www.sdnu.edu.cn

Shao ZZ, Wang HG, Liu H, Meng CH. Single carpooling problem based on matching degree clustering algorithm. *Journal of Software*, 2012,23(Suppl.(2)):204−212 (in Chinese). http://www.jos.org.cn/1000-9825/12040.htm

Abstract: The effect of carpooling is very significant in many aspects such as reducing the cost of logistics and traffic congestion. In this paper, a clustering heuristic strategy based on matching degree is introduced to assign the demand of services to one specific vehicle. In a single vehicle problem, a prior clustering idea to reduce the number of insertion operation is induced which may improve the efficiency of the algorithm. Additionally, migration operator is proposed to improve the success rate of matching and to reduce total costs. Real examples show that the algorithm not only cuts down a vehicle's idling rate, but also reduces the vehicle operation cost to some degree.
Key words: carpooling problem; matching degree clustering; prior clustering; heuristic algorithm

摘 要: 车辆合乘在降低物流成本和减少交通拥塞等方面作用显著.针对单车辆合乘问题,提出基于匹配度的聚类算法,用于将服务需求分配到具体某一辆车.借鉴"先验聚类"思想,算法中的单车辆合乘匹配过程的插入次数可大大减少,从而提高了算法效率.另外,为进一步提高车辆搭乘效率,降低运行成本,通过迁移算子对匹配度聚类过程进行微调.算例证明,该算法可显著降低车辆空载率,在一定程度上降低了车辆运行成本.
关键词: 车辆合乘匹配;匹配度聚类;先验聚类;启发式算法

车辆合乘匹配问题也称为拼车问题(carpooling problem)[1],是指相近路线的服务需求乘坐同一车辆出行,合乘产生的费用分摊.美国、新加坡[2,3]等国家早在 20 世纪 70 年代就从政策上开始支持合乘出行方式,因为该方式不仅有利于降低个体出行成本,对环境保护、交通拥塞、噪声污染等的控制也具有重要意义.文献[4]基于自适应插入算法解决了单车辆 DARP 问题,本研究与拼车问题具有一定相似性.我国近期也开始关注该问题,但相

* 基金项目: 国家自然科学基金(60970004); 山东省自然科学基金(ZR2011FQ029)
收稿时间: 2012-05-15; 定稿时间: 2012-11-09

关研究尚处于起步阶段.文献[5]提出利用模拟退火算法解决出租车的动态解决方案;文献[6]在区域-区域匹配算法的基础上提出基于交通路网的合乘路径匹配算法;文献[7]提出了基于量子进化算法的车辆共享模型.以上研究和政策探索大多是政策及理论研究,有关车辆合乘的实用研究成果尚不多见.

在物流领域及私家车交通领域,合乘的服务需求非常旺盛.2012年3月,北京市政府出台政策,鼓励多人合乘一辆出租车出行.而在物流领域中,各大物流企业的物流路线的整合利用,也属于车辆合乘的应用范畴.但是,要提高合乘效率,就需要寻找一种科学方法,建立车辆与服务需求的合理匹配关系.本文正是针对该实际问题展开研究,提出一种基于匹配度的聚类算法.该算法可实现车辆及服务需求的双向优选,不仅可以提高合乘成功率,还能有效降低车辆承担的运行成本,具有较高的实用价值.

1 单车辆合乘匹配问题数学模型

1.1 问题描述

某区域有一车辆可提供搭乘服务,其出发地点、到达地点确定且有具体的时间窗口要求.区域内有 n 个搭乘服务需求,其出发地点、到达地点确定,也有具体的时间窗口要求.在满足车辆自身时间约束前提下,可搭乘服务需求以降低其运行成本.假设车辆的出行信息及服务需求信息确定且保持不变,各服务需求没有差异性.假设以下条件:(1) 车辆及服务需求需在规定时间窗口内到达指定上车位置;(2) 如车辆到达某一上车点的时刻早于服务时间窗口下界,则需等待至该时间窗口下界,并在同一时刻离开;(3) 任何服务需求的搭载量不能大于车辆剩余搭载量,且任何时刻车辆不能超载;(4) 不考虑服务时间.问题的目标是在满足车辆时间约束前提下,提高搭乘服务需求数量,降低车辆运行承担成本.

1.2 问题形式化

令 P 为区域内服务需求集合,$P=\{0,1,2,\ldots,n\}$,定义 p_0^+, p_0^- 为车辆的起点和终点; p_i^+, p_i^- 分别为服务需求的起点和终点编号,$i=1,2,\ldots,n$.令集合 $N^+=\{p_i^+ \mid i\in P\}$, $N^-=\{p_i^- \mid i\in P\}$ 表示服务需求的起点编号集合和终点编号集合.定义 $N^+=\{0,1,2,\ldots,n\}$, $N^-=\{n+1,n+2,\ldots,2n+1)\}$.记 $N=N^+\cup N^-$,表示区域内所有起点和终点的集合. $\forall x\in N^+$,定义上车点编号函数 $Up(x)=x$,下车点编号函数 $Down(x)=n+x+1$.定义所有服务需求的上、下车点的时间窗口为 $[e_x,l_x]$, $x\in N$.

设研究区域内点的集合为 $V=N\cup V'$,其中 V' 表示车辆可经过的其他道路节点的集合,令 $|V'|=s$.V 中任意两点的道路距离已知.定义图 $G=(V,E)$,E 为图 G 所有边的集合,$E=\{(x,y)|x,y\in V\}$.定义 $A=\{a_{x,y}|x,y\in N\}$ 为图 G 的距离矩阵.不考虑路径方向性问题,有 $a_{x,y}=a_{y,x}$, $a_{x,x}=0$.基于距离矩阵 A,定义车辆在任意两点间所用的行驶时间矩阵 $T=\{t_{x,y}|x,y\in N\}$,其中 $t_{x,y}=\vec{d}_{x,y}/v$,v 为车辆平均速度,$\vec{d}_{x,y}$ 为点 x 到点 y 的最短路径长度.定义车辆的固定运行成本为 fc,单位里程内每增加一单位运载量增加成本为 ac,车辆最大装载量为 Q,当前已装载货物量为 q.任意 $i\in N^+$,定义上货量 Δq_i, $i\in N^+|\{0\}$;任意 $i\in N^-$,定义卸货量为 Δq_i, $i\in N^-|\{n+1\}$.简单起见, $\forall i\in N$, 令 $\Delta q_{down(i)}=-\Delta q_{up(i)}$.

1.2.1 变量定义

1) $X_{x,y}$:二进制变量,$x\in N^+$,$y\in V$,$x\neq y$.$X_{x,y}=1$,说明车辆行驶到了某服务需求上车点,即成功搭乘该服务需求,否则 $X_{x,y}=0$.

2) T_x:时刻变量,$x\in V$,为车辆到达顶点 x 的实际时刻.

3) B_x:时刻变量,$x\in V$,为车辆离开顶点 x 的时刻.当 $x\in N$ 时,需满足服务时间窗口约束,有

$$B_x=\max(T_x,e_x),x\in N \tag{1}$$

4) q_x:货品数量,$x\in V$,为车辆离开顶点 x 时的货品数量,有 $q_x\leqslant Q$.如 $X_{x,y}=1$,则 $q_y=q_x+\Delta q_y$.

5) $Cost$:为车辆运行全程所承担的成本.设车辆离开出发点 x_0 时的状态为 $Status_{x_0}$ 状态.每到达一个顶点后离开,就转化为一种新的状态.记 $Status_{x_u}$ 是车辆的第 u 个状态, $Status_{x_u}=(q_{x_u},l_u)$, q_{x_u} 是车辆在边 (x_{u-1},x_u) 上运行时的货品数量;l_u 是顶点 x_{u-1} 到顶点 x_u 之间的路径长度,即 $l_u=a_{x_{u-1},x_u}$.车辆于该路段的变动成本为

$cost_u = ac \cdot l_u \cdot q_{x_u}$. 设车辆到达终点时共经历 U+1 个状态(U 个路段),路段总长度为 L,按照成本均担原则,车辆承担的运行成本为

$$Cost = \sum_{u=1}^{U+1}\left[\left(cost_u + fc \cdot \frac{l_u}{L}\right) \cdot \frac{q}{q_{x_u}}\right] \tag{2}$$

6) WT_x:时间变量,$x \in P$,为服务需求 x 的等待时间,即从最早上车时刻到实际上车时刻的时间,有

$$WT_x = T_x - e_x \tag{3}$$

7) RT_x:时间变量,$x \in P$,是服务需求 x 从上车点到下车点所用的时间,即乘车时间.有

$$RT_x = T_{n+x+1} - B_x \tag{4}$$

1.2.2 目标函数定义

首先考虑车辆本身承担的运行费用最少,然后考虑最大化搭乘数量,有

$$\min\ Cost \tag{5}$$

$$\max \sum_{x \in N^+}\sum_{y \in V\setminus\{x\}} X_{x,y} \tag{6}$$

1.2.3 约束条件说明

C1) 时间窗口约束:车辆须在规定时间窗口内到达上下车点.有

$$e_x \leqslant T_x \leqslant l_x, x \in N \tag{7}'$$

特殊情况下会出现车辆早于时刻 e_x 到达顶点 x 的情况,则(7)′可修正为

$$T_x \leqslant l_x, x \in N \tag{7}$$

C2) 一次访问约束:车辆到某一个上、下车点,每一个点只能被服务 1 次.有

$$\sum_{y \in V\setminus\{x\}} X_{x,y} \leqslant 1, x \in N^+ \tag{8}$$

$$\sum_{x \in V\setminus\{y\}} X_{x,y} \leqslant 1, y \in N^- \tag{9}$$

C3) 车辆出发点、到达点约束:车辆必须从规定地点出发且到达指定终点.有

$$\sum_{y \in N^+\setminus\{0\}} X_{0,y} = 1 \tag{10}$$

$$\sum_{x \in V\setminus\{n+1\}} X_{x,n+1} = 1 \tag{11}$$

C4) 成对约束:搭乘服务需求的上车点 x 和下车点 n+x+1 若被服务,则必须都被服务.有

$$\sum_{y_1 \in V\setminus\{x\}} X_{x,y_1} - \sum_{y_2 \in V\setminus\{n+x+1\}} X^j_{y_2,n+x+1} = 0, x \in N^+ \tag{12}$$

C5) 访问顺序约束:搭乘服务需求的上车时刻加乘车时间,应该在需求的下车时刻之前.有

$$T_x + RT_x \leqslant T_{m+n+x}, x \in N^+ \tag{13}$$

C6) 车辆搭载容限约束:行驶过程中,任意时刻不能超过车辆容限.有

$$q_0 = q_{n+1} = q \tag{14}$$

$$q_x \leqslant Q, x \in V \tag{15}$$

2 匹配度聚类启发式算法

本文提出基于匹配度的聚类启发式(matching degree clustering,简称 MDC)算法,用于解决从若干服务需求中筛选与搭乘车辆最为匹配的服务需求.其基本思想是:以车辆为聚类中心,将与车辆匹配度较大的服务需求聚为一类,然后借鉴“先验聚类”思想[4],执行单车辆插入过程,以寻求最搭乘方案.由于匹配度聚类过程属于概率性聚类,为提高效果,算法采取了反复迭代策略,直到满足问题精度要求或者满足停止条件.

2.1 几个定义

定义 1. 松弛时间 RTv 是指服务需求 x 完成从上车点 $x^{\uparrow}$ 到下车点 $x^{\downarrow}$ 所用时间 $t_{x^{\uparrow},x^{\downarrow}}$ 与 $x^{\uparrow}$ 的时间窗口开始时刻到 $x^{\downarrow}$ 时间窗口结束时刻之间的时间差值,有 $RTv_x=(l_{x^{\downarrow}}-e_{x^{\uparrow}})-t_{x^{\uparrow},x^{\downarrow}}$.

定义 2. 线路拟合度 H 是指服务需求 x 的上车、下车点所确定的直线段与车辆和 x 邻近的原行驶路径之间的线路拟合情况.

图 1 中,有向线段 SD 是由服务需求 x 的上车点 S 和下车点 D 所确定的路线,有向曲线 L 是车辆初始行驶路径(假设车辆从左向右行驶),中间结点为车辆途经点.作如下处理:1) 从曲线 L 的各途经点集合中分别找出与点 S,D 距离最近的点 S',D',定义两点确定的直线为 L'; 2) 定义 SD 与 L'的夹角为α,按照式(16)计算服务需求 x 与车辆的线路拟合度:

$$H_x=\frac{1}{2}\left(e^{-\frac{1}{2C}(|S'-S|+|D'-D|)}+\cos|\alpha|\right) \tag{16}$$

其中,$|S'\text{-}S|$,$|D'-D|$为两点之间的距离;$C>0$ 为常量,一般可取所研究凸平面空间的半径.显然,$H_x\to 1$ 说明拟合度越大,匹配成功的可能性也越大,反之则越小.

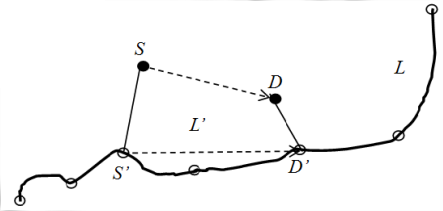

Fig.1 Illustration of fitting between service demand route and vehicle route

图 1 服务需求路线与车辆路线的拟合示例

定义 3. 匹配度 MP 是服务需求 x 与车辆匹配成功与否的可能性,取值范围为[0,1],用 MP_x 表示. MP_x 越大,说明匹配成功的可能性越大.匹配度是一个综合性指标,与服务需求的松弛时间、线路拟合度、服务需求货物量等关系密切.定义

$$MP_x=\begin{cases}0, & (l_{x^{\uparrow}}<e_0)\ \text{or}\ (e_{x^{\downarrow}}>l_{n+1})\\ \beta_1\cdot\dfrac{RTV^x}{\sum\limits_{i\in P\setminus\{0\}}RTV^i}+\beta_2\cdot\dfrac{H_x}{\sum\limits_{i\in P\setminus\{0\}}H_i}+\beta_3\cdot\left(1-\dfrac{\Delta q_x}{\sum\limits_{i\in P\setminus\{0\}}\Delta q_i}\right), & \text{else}\end{cases} \tag{17}$$

式(17)表明:若需求 x 的上车最晚时刻早于车辆出发的最早时刻,或者下车最早时刻晚于车辆到达终点的最晚时刻,就失去搭乘资格;否则将以一定资格拟搭乘,$\beta_1,\beta_2,\beta_3\in[0,1]$,且$\beta_1+\beta_2+\beta_3=1$.在满足时间约束的前提下,松弛时间越大、线路拟合度越高,则匹配度越大;而服务需求容量越小,匹配度越大.

基于式(17),可得到所有服务需求对搭载车辆的匹配度向量 MP.设定阈值 $T\in(0,1)$,按照轮赌原则对满足 $MP_x\geqslant T$ 的需求进行选择,可确定当期车辆的拟搭乘服务需求子集 N_m,有 $N_m\in N$ 且$|N_m|<|N|$.

2.2 基于先验聚类的单车辆合乘匹配算法

确定子集 N_m 后,问题规模变小,寻优将会更具有针对性.单车辆合乘匹配的目标是寻找一个有序服务序列,使得 N_m 内的服务需求能够尽可能地实现搭乘,同时降低车辆承担的运行成本.问题的可行解是由多个成对的上、下车点组成的有序服务序列,如 $(x^{\uparrow},y^{\uparrow},z^{\uparrow},z^{\downarrow},x^{\downarrow},y^{\downarrow})$ 是一个包含 3 个服务需求的服务序列.在不考虑时间、容量等约束时,备选服务序列数量过大.传统的解决算法包括单对插入、路径内插入及路径间插入等,基本原则是将新服务需求的上、下车点插入已有服务序列中并保持序列依旧可行,有文献采用路径交换等随机原则生成多种新解,但效果并不理想.芬兰学者 Hame[4]提出了基于先验知识的“先验聚类(prior clustering)”单车辆插入算法,在解决 DARP 问题时作用明显,本文借鉴了此算法.

路径寻优过程是一个迭代过程,包括两个主要步骤:(1) 将新服务需求 i 插入由所有已存在的服务需求(已插入搭乘服务需求 1,2,…,i–1)组成的所有可行服务序列集合 S_{i-1} 中;(2) 确定所有服务序列的可行性,生成一个新的可行服务序列集 S_i.迭代多次生成 S_n 后,计算目标函数值即可找到当前最优方案.但问题的关键在于"插入"操作太多,导致计算量剧增,难以在较短时间内得到较好的方案.不考虑任何约束,最差情况下,将搭乘服务需求 i 插入前 i–1 个搭乘服务需求所组成的有序服务序列集合 S_{i-1} 的次数为 $m_i=(2^i)!/2^i$,则 m 个服务需求总的插入操作次数最多为 $\sum_{i=1}^{m}\frac{(2i)!}{2^i}$.如果在插入操作之前,根据各种约束条件及先验知识对"插入可行性"进行验证,则可排除大量无效插入操作,这正是"先验知识"的作用.

$\forall x,y\in N$, 在满足各种约束条件及先验知识的前提下,如果顶点 x 可到达顶点 y,记为 $x\rightarrow y$,则称"x 可达 y",有 $F_{x,y}$=1;否则,$F_{x,y}$=0.

定义 4(优先关系). $\forall x\in C_l\subseteq N,\forall y\in C_k\subseteq N$,如 $F_{x,y}$=0,则称集合 C_k 优先于 C_l,记为 $C_k<C_l$.假设有集合 C_k,C_l,C_m,则有如下特点:1) $C_k\prec C_k$;2) 如果 $C_k<C_l$,则 $C_l\prec C_k$;3) 如果 $C_k<C_l,C_l<C_m$,则 $C_k<C_m$.

结合问题的先验知识(如"$x^{\uparrow}\rightarrow x^{\downarrow}$"),在插入操作之前,可把一个包含多名服务需求的上、下车点集合分成多个具有优先关系的聚类,该过程称为先验聚类过程.假设 1~4 号服务需求的上、下车点集合为{$1^{\uparrow},1^{\downarrow},2^{\uparrow},2^{\downarrow},3^{\uparrow},3^{\downarrow},4^{\uparrow},4^{\downarrow}$},其可达关系见表 1.执行如下操作:

(1) 对表中各行求和,并按"行和"进行降序排列,然后按照由行排序所确定的顺序对列顺序进行相应变动,得到表 2.

2) 按照"行元素内容相同"的原则将所有顶点分组.基于优先关系定义,有

$$\{1^{\uparrow}\}<\{1^{\downarrow},2^{\uparrow},3^{\uparrow}\}<\{2^{\downarrow}\}<\{3^{\downarrow},4^{\uparrow}\}<\{4^{\downarrow}\}.$$

子集{$1^{\uparrow}$},{$2^{\downarrow}$},{$4^{\downarrow}$}中的元素位置已经确定,因此在后面的插入操作中,只需对子集{$1^{\downarrow},2^{\uparrow},3^{\uparrow}$}和{$3^{\downarrow},4^{\uparrow}$}中的元素进行内部排序即可.可以看出,经过上面的行列变换,我们得到了重要结论:各顶点的位置已经相对确定,凡是不符合以上规则的服务序列均无需执行插入操作,这将大大减少插入次数.对形如{$1^{\downarrow},2^{\uparrow},3^{\uparrow}$}的待排序序列,可充分考虑各限制条件,并采用遗传算法[8]、K-OPT[9]等方法选出符合条件的序列,从而获得车辆当前最优方案.

Table 1 Illustration of reachable relationship

表 1 可达关系示例

顶点	$1^{\uparrow}$	$1^{\downarrow}$	$2^{\uparrow}$	$2^{\downarrow}$	$3^{\uparrow}$	$3^{\downarrow}$	$4^{\uparrow}$	$4^{\downarrow}$	行和
$1^{\uparrow}$	1	1	1	1	1	1	1	1	8
$1^{\downarrow}$	0	1	1	1	1	1	1	1	7
$2^{\uparrow}$	0	1	1	1	1	1	1	1	7
$2^{\downarrow}$	0	0	0	1	0	1	1	1	4
$3^{\uparrow}$	0	1	1	1	1	1	1	1	7
$3^{\downarrow}$	0	0	0	0	0	1	1	1	3
$4^{\uparrow}$	0	0	0	0	0	1	1	1	3
$4^{\downarrow}$	0	0	0	0	0	0	0	1	1

Table 2 Changed reachable relationship

表 2 变换后的可达关系示例

顶点	$1^{\uparrow}$	$1^{\downarrow}$	$2^{\uparrow}$	$3^{\uparrow}$	$2^{\downarrow}$	$3^{\downarrow}$	$4^{\uparrow}$	$4^{\downarrow}$	行和
$1^{\uparrow}$	1	1	1	1	1	1	1	1	8
$1^{\downarrow}$	0	1	1	1	1	1	1	1	7
$2^{\uparrow}$	0	1	1	1	1	1	1	1	7
$3^{\uparrow}$	0	1	1	1	1	1	1	1	7
$2^{\downarrow}$	0	0	0	0	1	1	1	1	4
$3^{\downarrow}$	0	0	0	0	0	1	1	1	3
$4^{\uparrow}$	0	0	0	0	0	1	1	1	3
$4^{\downarrow}$	0	0	0	0	0	0	0	1	1

2.3 启发式迁移规则

匹配度聚类算法具有一定的局限性,某些匹配良好的服务需求有可能没有进入子集 N_m,从而影响匹配效果.为缓解该局限性,算法在完成单车辆匹配过程后,采取启发式修正策略,将某些劣质服务需求迁出 N_m,同时适当地从集合 N–N_m 中寻找新的服务需求,用以扩展寻优范围,提高解的质量.文献[10]发现,长弧在高质量解中出现的几率较小.借鉴该结论,结合先验聚类效果,本文对一次聚类方案进行启发式分析,对那些搭乘失败及导致长弧的服务需求执行选择性"迁出"操作,同时从差集 N–N_m 中选择新的服务需求.

对 N_m 中匹配失败的服务需求 x,迁出概率为 1.对虽然匹配成功但导致长弧的服务需求有如下操作:定义弧长阈值 $\delta=\theta\cdot S^*/(|Path^*|-1))$,其中 $\theta\in[0.5,1.5]$ 为调节因子,S^*为本次迭代中生成的最优路径的路径总长度,即 $S^*=\sum_{x\in V}\sum_{y\in V}X_{x,y}\cdot a_{x,y}$;$Path^*$为当前最优路径,$|Path^*|$为路径上顶点的个数.设 $Lx_l^{\uparrow},Lx_r^{\uparrow}$ 分别表示上车点 $x^{\uparrow}$ 到其前续结点、后续结点的弧长度,$Lx_l^{\downarrow},Lx_r^{\downarrow}$ 分别表示下车点 $x^{\downarrow}$ 到其前续结点、后续结点的弧长度.定义

$L_x=\max(Lx_l^{\uparrow},Lx_r^{\uparrow},Lx_l^{\downarrow},Lx_r^{\downarrow})$ 为服务需求 x 所连接的弧的最大长度.计算得到 $Path^*$ 中各服务需求的弧的最大长度的前 r(r≤5,根据需求自动调整)个服务需求 mx,组成子集 LN_m,$|LN_m|=r$.定义 emP_{mx} 为从集合 N_m 中将 mx 迁出的概率,有:

$$emP_{mx}=\begin{cases}0, & L_{mx}-\delta<0\\ \min\left(\dfrac{L_{mx}}{\delta}-1,1\right), & L_{mx}-\delta\geqslant 0\end{cases} \tag{18}$$

如果 emP_{mx} 较大,说明元素 mx 形成了长弧,则 mx 迁出的可能行就较大.实际应用中,可设置当 emP_{mx} 超出某阈值(例如 0.8)时,即将 x 迁出.根据迁出服务需求的数量,算法根据概率从差集 N–N_m 中嵌入新的服务需求.

3 数值实验及结果分析

为验证 MDC 算法的有效性及性能,本文设计了一组实验.以中国济南市环路以内的主干道路组成的方形路网为研究对象,假设该地域内有一辆车提供搭乘服务,有 n=30 个服务需求,$|V|$=105.实验基于如下平台:Intel(R) Celeron(R) CPU E3300,2.50GHz,4G 内存,操作系统为 Windows XP Professional.

3.1 数据生成方法及参数选择

固定车辆起点和终点分别为 p_0^+, p_0^-,车辆总容量为 Q、当前载货量为 q,以及服务时间窗口为$[e_0,l_0]$,$[e_{n+1},l_{n+1}]$.要求满足 $\left|p_0^+-p_0^-\right|\geqslant C$, C 为所研究凸空间的半径.根据 Poisson 分布生成上下车点 p_0^+ 和 p_0^-,以及对应的时间窗口.图 2 是根据以上方法生成的路网图.

Fig.2 Test road network based on Jinan City of China ($|V|$=105)

图 2 基于中国济南城市路网生成的路网($|V|$=105)

车辆生成信息为 p_0^+=(164,436), p_0^-=(1650,376),对应时间窗为$[e_0,l_0]$=[8,18],$[e_{31},l_{31}]$=[40,50];平均速度 v=60,车辆固定成本 fc=80,可变成本 ac=0.02;车辆容量 Q=800,当前载货量 q=250.表 3 是 30 个服务需求的基本信息.

3.2 实验过程及分析

本实验设定 MDC 算法执行时,聚类次数不超过 30 次,搭乘插入寻优次数不超过 1 000 次,且当搭乘服务需求数量无明显变化时算法结束.对 MDC 算法连续运行 20 次,所有对比参数取平均值.

首先确定匹配度影响因子β_1,β_2,β_3 的取值.表 4 说明,当$\beta_1=\beta_2=\beta_3$=1/3,即 3 个影响因子具有相同影响力时,算法具有较高的性能;当$\beta_1=\beta_2$=0.4,β_3=0.2 时,算法性能最高.这说明,服务需求的松弛时间、服务需求与车辆原始路径之间的线路拟合度对匹配度的影响较大,而需求容量对匹配度的影响相对较小,这在一定程度上符合实际规律.后文实验中均设定$\beta_1=\beta_2$=0.4,β_3=0.2.对算法执行过程进行跟踪,得到匹配度向量 MP,见表 5.按照 MDC 中的聚类规则,编号为 15,16,17 的服务需求(R_{15},R_{16},R_{17})被匹配到车辆上.

Table 3 Basic information of service demand

表 3 服务需求基本信息

需求编号	需求起始点坐标及时间窗	所需容量	需求编号	需求起始点坐标及时间窗	所需容量
1	(126, 256), (601, 531) (5,10) , (20, 25) (310, 425), (712, 629)	150	16	(601, 443), (1151,443) (15, 20) , (30, 35) (813, 441), (1413, 397)	120
2	(10, 15) , (25, 30) (601, 443), (902, 717)	120	17	(20, 25) , (35, 40) (618, 324), (971, 381)	140
3	(15, 29) , (30, 35) (448, 41) , (797, 46)	80	18	(10, 15) , (20, 25) (749, 406), (1156, 352)	100
4	(10, 15) , (18, 25) (697, 45) , (1071, 17)	90	19	(15, 20) , (25, 30) (873, 390), (1354, 298)	90
5	(15, 20) , (25, 30) (577, 639), (1411, 459)	100	20	(20, 25) , (30, 35) (736, 533), (961, 499)	80
6	(5, 10) , (30, 35) (712, 629), (1307, 496)	160	21	(1, 5) , (10, 15) (822, 539), (1102, 507)	110
7	(5, 10) , (25, 30) (835, 720), (1226, 504)	130	22	(5, 10) , (10, 25) (588, 71) , (736, 533)	150
8	(10, 15) , (25, 30) (585, 207), (1438, 70)	70	23	(10, 15) , (20, 25) (585, 257), (712, 629)	110
9	(15, 20) , (35, 40) (795, 247), (1278, 59)	90	24	(15, 20) , (25, 30) (662, 374), (835, 720)	140
10	(20, 25) , (30, 35) (967, 247), (1291, 111)	80	25	(15, 20) , (25, 30) (697, 45) , (1084, 225)	120
11	(25, 30) , (35, 40) (452, 363), (904, 298)	110	26	(15, 20) , (25, 30) (952, 146), (1152, 409)	110
12	(10, 15) , (25, 30) (678, 397), (1250, 248)	130	27	(20, 25) , (30, 35) (1152, 316), (1413, 397)	90
13	(15, 20) , (30, 35) (809, 330), (1105, 272)	90	28	(30, 35) , (35, 40) (749, 406), (1053, 380)	100
14	(20, 25) , (30, 35) (310, 425), (912, 410)	100	29	(45, 50) , (60, 66) (442, 185), (1080, 150)	150
15	(10, 15) , (20, 25)	80	30	(20, 26) , (39, 55)	160

Table 4 Influence of β_1, β_2, β_3

表 4 参数β_1,β_2,β_3取值对算法的影响

parameters			第一次获得最优解的聚类次数	平均算法运行时间(s)	成功搭乘的服务需求数量	最优解的平均成本(Cost/L)
β1	β2	β3				
1/3	**1/3**	**1/3**	5	32.119	**3**	**84.2381**
0.2	0.4	0.4	7	47.001	1	87.6507
0.2	0.6	0.2	4	29.684	2	85.1247
0.2	0.2	0.6	11	54.335	1	87.3254
0.4	0.2	0.4	8	51.394	**3**	83.2414
0.4	**0.4**	**0.2**	**3**	27.335	**3**	**82.9173**
0.6	0.2	0.2	5	33.415	2	86.1275

Table 5 Matching vector (n=30)

表 5 匹配向量(n=30)

编号	1	2	3	4	5	6
MP	0.76	0.6	0.81	0.32	0.15	0.51
	7	8	9	10	11	12
	0.87	0.71	0.07	0.75	0.26	0.62
	13	14	***15***	***16***	***17***	18
	0.46	0.81	***0.92***	***0.96***	***0.91***	0.17
	19	20	21	22	23	24
	0.01	0.37	0.2	0.56	0.77	0.76
	25	26	27	28	29	30
	0.12	0.48	0.31	0.67	0	0.51

算法运行完成后,得到车辆的搭乘之前和搭乘之后的路径如下:

车辆初始路径:3→10→8→13→35→43→44→55→89→62→68→65→79→81→83→85→86

车辆搭乘后路径:

3,$\underline{8T}$,( )→10,$\underline{11T}$,(R_{15})→8,$\underline{14T}$,(R_{15})→13,$\underline{17T}$,(R_{15},R_{16})→35,$\underline{19T}$,(R_{15},R_{16})→43,$\underline{21T}$,(R_{15},R_{16},R_{17})

→44,23T,(R_{15},R_{16},R_{17})→105,24T,(R_{16},R_{17})→104,25T,(R_{16},R_{17})→55,26T,(R_{16},R_{17})
→89,28T,(R_{16},R_{17})→62,29T,(R_{16},R_{17})→68,31T,(R_{17})→65,33T,(R_{17})→79,35T,(R_{17})
→81,37T,( )→83,39T,( )→85,43T,( )→86,45T,( )

其含义是:车辆在 8T 时刻从顶点 3 出发;在 11T 时刻到达顶点 10,并在该顶点搭乘需求 R_{15};在 14T 时刻到达顶点 8;在 17T 时刻到达顶点 13,并在该顶点继续搭乘需求 R_{16};在 19T 时刻到达顶点 35;在 21T 时刻到达顶点 43,并在该顶点继续搭乘需求 R_{17};在 23T 时刻到达顶点 44;在 24T 时刻到达顶点 105,在该顶点需求 R_{15} 下车;在 25T 时刻到达顶点 104;在 26T 时刻到达顶点 55;在 28T 时刻到达顶点 89;在 29T 时刻到达顶点 62;在 31T 时刻到达顶点 68,在该顶点需求 R_{16} 下车;在 33T 时刻到达顶点 65;在 35T 时刻到达顶点 79;在 37T 时刻到达顶点 81,在该顶点需求 R_{17} 下车;在 39T 时刻到达顶点 83;在 43T 时刻到达顶点 85;在 45T 时刻到达顶点 86,到达车辆终点,结束运行,具体如图 3 所示,路径总长度为 3 069.

可以发现,车辆搭乘服务需求后,其运行路线发生了一些变化:初始路径中,车辆由顶点 44 直接运行至顶点 55;而搭乘后的路径表示,为了能满足搭乘需求 R_{15} 的下车需求,车辆由顶点 44 开始,途经顶点 105,104 后,又绕回顶点 55.由已知条件知,需求 R_{15} 的下车时间窗口为[20,25],而车辆到达顶点 105(即 R_{15} 的下车点)的时刻是 25T,说明是满足服务需求的时间窗口约束的.

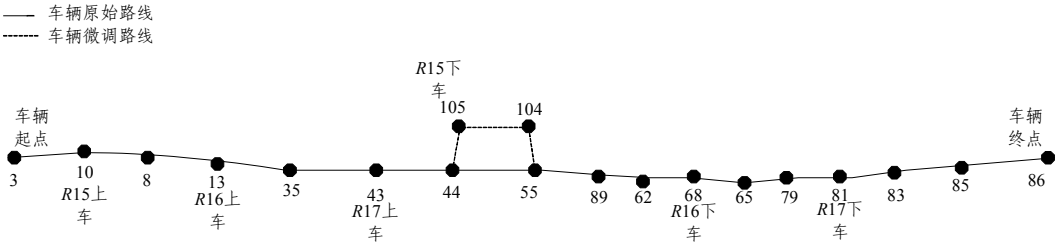

Fig3. Contrast between before riding and after riding

图 3 车辆搭乘前后路线对比

为说明车辆搭乘的有效性,按照以需求容量平摊运行成本的原则,图 4 给出了车辆及各服务需求在各路段承担成本曲线图.可以发现,车辆在 24T 时刻所承担的成本最小,这是因为此时同时搭乘了 R_{15},R_{16} 和 R_{17}.显然搭乘数量越多,车辆本身所承担的成本就越小.经过测算,车辆未搭乘时,其运行成本为 151.05;搭乘后,车辆运行总成本为 199.344,根据式(2),车辆本身承担 132.6,服务需求 R_{15} 承担 16.518,服务需求 R_{16} 承担 21.817,服务需求 R_{17} 承担 28.409,搭乘前后车辆本身所承担的成本降低了 18.45.为说明车辆搭乘前后成本变化,又对各路段中车辆搭乘前后的平均成本进行对比,如图 5 所示.显然,当车辆载货量达到最大值时,所经路段的平均成本也处于最低点.

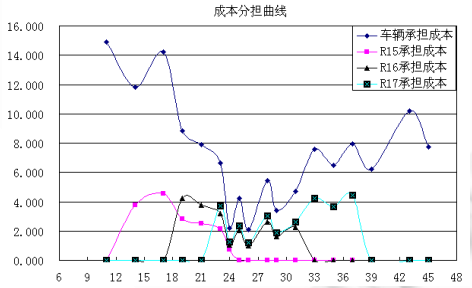

Fig.4 Curve of cost-sharing

图 4 成本分担曲线

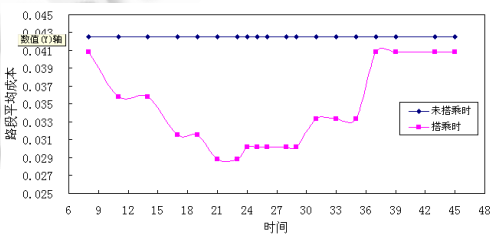

Fig.5 Curve of average cost

图 5 平均成本曲线

4 结 论

本文构建了单车辆合乘匹配问题的数学模型,并提出解决此问题的启发式算法 MDC.其思想是首先根据匹配度向量将搭乘服务需求聚集到某辆车,然后基于改进的“先验聚类”插入算法对单车辆匹配过程进行求解.为改善聚集效果,算法中引入迁移策略,对聚类效果进行优化调整.算例表明,MDC 算法能够以较高的准确性选择搭乘服务需求,能在较短时间内得到搭乘方案,并可有效降低车辆所承担的运行成本.算法没有考虑多车辆环境下搭乘成本的整体优化,也没有考虑路况变化对搭乘匹配的影响,后期将在以上几个方向重点展开研究.

References:

[1] Erik T. Ferguson and associates. The rise and fall of the American carpool: 1970–1990. Transportation, 1997,24(4):349–376.

[2] Dailey DJ, Loseff D, Meyers D. Seattle smart traveler: Dynamic ridematching on the World Wide Web. Transportation Research Part C, 1999,7(1):17–32.

[3] Buliung RN, Soltys K, Bui R, Habel C, Lanyon R. Catching a ride on the information super-highway: Toward an understanding of internet-based carpool formation and use. Transportation, 2010,37(6):849–873.

[4] Häme L. An adaptive insertion algorithm for the single-vehicle dial-a-ride problem with narrow time windows. European Journal of Operational Research, 2011,209(1):11–22.

[5] Zhang J, He RC. Solving dynamic taxipooling problem with simulated annealing algorithm. Journal of Lanzhou Jiaotong University, 2008,27(3):85–88 (in Chinese with English abstract).

[6] Zhai Y, Yang JL, Lian J, Fan MQ. Path matching algorithm for carpool information retrieval. Computer and Communications, 2007, 25(134):27–30 (in Chinese with English abstract).

[7] Wang WL, Huang HP, Zhao YW, Zhang JL. Dynamic customer demand VRP with soft time windows based on vehicle sharing. Computer Integrated Manufacturing Systems, 2011,17(5):1056–1063 (in Chinese with English abstract).

[8] Wang WL, Li JF, Wang J. Double population genetic algorithm for multi-objective job shop scheduling problem. Computer Integrated Manufacturing Systems, 2011,17(4):808–815 (in Chinese with English abstrct).

[9] Mladenović N, Urošević D, Hanafi S, Ilic A. A general variable neighborhood search for the one-commodity pickup-and-delivery traveling salesman problem. European Journal of Operational Research, 2012,220(1):270–285.

[10] Toth P, Vigo D. The granular tabu search and its application to the vehicle routing problem. INFORMS Journal on Computing, 2003,15(4):333–346.

附中文参考文献:

[5] 张瑾,何瑞春.解决动态出租车拼车问题的模拟退火算法.兰州交通大学学报,2008,27(3):85–88.

[6] 翟泳,杨金梁,连剑,樊铭渠.合乘出行信息检索的路径匹配算法.交通与计算机,2007,25(134):27–30.

[7] 王万良,黄海鹏,赵燕伟,张景玲.基于车辆共享的软时间窗动态需求车辆路径问题.计算机集成制造系统,2011,17(5):1056–1063.

[8] 王伟玲,李俊芳,王晶.求解多目标作业车间调度问题的双种群遗传算法.计算机集成制造系统,2011,17(4):808–815.

邵增珍(1976－),男,山东寿光人,博士生,副教授,CCF 会员,主要研究领域为智能计算,路径优化.

刘弘(1955－),女,博士,教授,博士生导师,CCF 高级会员,主要研究领域为智能计算.

王洪国(1966－),男,博士,教授,博士生导师,主要研究领域为电子政务,运筹优化.

孟春华(1989－),女,硕士生,CCF 学生会员,主要研究领域为路径优化,电子商务.