# 大规模邻域搜索算法求解时变车辆调度问题①

李妍峰[1]，李 军[1]，高自友[2]
(1. 西南交通大学经济管理学院，成都 610031;
2. 北京交通大学交通运输学院系统科学研究所，北京 100044)

**摘要**: 对时变网络车辆调度问题提出一种满足先入先出准则的时变处理方法，并建立相应的数学模型，提出一种基于大规模邻域搜索技术的智能优化算法进行求解，算法顶层采用动态规划算法搜索环状交换邻域以得到每辆车的最佳服务顾客集合；底层设计动态搜索算法用以安排每辆车的最佳服务路线. 在此基础上提出顶层加入虚拟顾客和底层嵌入 insert 两类改进策略. 通过实验仿真比较，验证了所提算法的有效性.

**关键词**: 时变网络车辆调度问题；先入先出；大规模邻域搜索；动态搜索算法
**中图分类号**: F253.4 **文献标识码**: A **文章编号**: 1007 - 9807(2012)01 - 0022 - 11

## 0 引 言

车辆调度问题(vehicle routing problem，VRP)是运筹学和组合优化领域里最经典的一类问题. 在静态网络中车辆在任意两节点间的行驶时间只与节点间距离有关，不随时间发生改变，而在实际城市交通网络中，由于受到交通管理、交通流量、交通事故、上下班高峰期等因素的影响，车辆行驶速度总在不停变化. 通常将一天分为几个不同时段，不同时段内车辆在任意两节点间的行驶时间不同. 研究时变网络 VRP 问题(time dependent vehicle routing problem，TDVRP) 具有更强的实际意义.

在 TDVRP 问题中，车辆行驶往往会从一个时段跨越到另一个(或几个) 时段，跨时段处理是时变特性处理的关键[1-11]. Malandraki 等[1-2] 采用阶段函数描述车辆在不同出发时段对应的行驶时间. 该策略是最常见的一种时变处理方法，但未考虑行驶时间在跨时段时的变化. 魏航等[12-13] 沿用该处理方法对有宵禁限制的最短路问题和有害物品运输问题进行了研究. Hill 和 Benton[4] 提出基于行驶速度阶段函数的处理方法，以上两种处理方法均不满足先入先出(first-in first-out，简写 FIFO) 准则. Ichoua 等[6] 调整了车辆在跨时段时的行驶速度，以保证行驶时间的连续性，该方法满足 FIFO 准则，但计算过程较复杂，需要已知任意两节点间的距离及每时段的旅行速度. Haghani 等[8]、Donati 等[9] 沿用了该时变处理方法. 李妍峰等[10-11] 对时变旅行商问题提出一种满足 FIFO 准则的跨时段处理方法，推导出旅行商在走行过程中跨单时段和多时段对应的旅行时间，处理更简便.

由于 TDVRP 问题受时间影响较大，且问题非常复杂，算法需要适应时变网络性质，目前相关算法研究很少. 最优化算法只能求解某些小规模 TDVRP 问题，其应用受到很大的限制. 大部分研究者考虑如何构造启发式算法求解 TDVRP 问题. 求解算法主要包括最近邻算法[1]、禁忌搜索算

① 收稿日期: 2010 - 06 - 07; 修订日期: 2011 - 09 - 22.
基金项目: 国家自然科学基金资助项目(71001005)；中国博士后科学基金资助项目(20090460196)；中国博士后特别资助项目(201003043)；中央高校基本科研业务费专项资金资助项目(SWJTU11CX087)；四川省教育厅社会科学研究项目(09SB066).
作者简介: 李妍峰(1980—)，女，四川乐山人，博士后，讲师. Email: yanwaa@126.com

© 1994-2012 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

法[6]、遗传算法[8]和蚁群算法[9]等. 在邻域搜索领域,Malandraki 和 Daskin[1]提出最近邻算法,求解了 10 ~ 25 个顾客的 TDVRP 问题. 该方法求解问题规模很小,且未考虑行驶时间在跨时段时的变化,问题不满足 FIFO 准则,不能为实际 VRP 问题提供决策支持.

邻域搜索算法是一类应用广泛的改进算法,在邻域搜索中,邻域越大,邻域解的质量越好,从而最终解的质量也就越好,但搜索邻域的时间就会越长. 所谓大规模领域( very large scale neighborhood,VLSN) 搜索技术就是探索有效的搜索策略,使其既能保持大规模邻域的寻优能力,又能克服低效耗时的弱点,从而提高算法的竞争能力. Congram[14]应用 dynasearch 对旅行商问题和单机调度问题进行了研究. Ahuja 等人[15]首次对 VLSN 研究进行了综述,标志着这一研究已成为智能优化搜索领域极具潜力的重要分支和新的研究方向. 利用 VLSN 搜索技术求解 VRP 研究非常少,Thompson 和 Psaraftis[16]构造了基于 VLSN 搜索技术的环状交换方法求解 VRP 问题. 目前还未见 VLSN 搜索技术求解 TDVRP 问题的相关研究.

本文对 TDVRP 问题建立了数学模型,提出一种满足 FIFO 准则的跨时段时变处理方法,并构造基于 VLSN 搜索技术的环状交换算法求解该问题,在此基础上提出两类改进策略,最后通过实验仿真进行算法性能比较.

## 1　时变处理方法

在一天中,车辆的行驶时间受到交通管理、交通流量、交通事故、上下班高峰期等因素的影响而发生变化,通常根据这些因素把一天分为几个时段,在同一时段内路段特征近似相同,如具有相同的拥挤程度、相同的流量等. 车辆在行驶过程中可能会从当前时段跨越到后一个( 或几个) 时段. 假设一天被分为 $P$ 个时段, $[B_p, B_{p+1}]$ 表示时段 $p(p \in [1,P])$ 的区域. 节点 $i$ 到节点 $j$ 的距离为 $d_{ij}$,车辆行驶 $d_{ij}$ 需跨 $K(K < P)$ 个时段,其长度被分为 $d_{ij}^1, d_{ij}^2, \cdots, d_{ij}^{K+1}$,分别为车辆于每一时段所行驶的距离. 在不考虑跨时段情况下,车辆于第 $l(l = 1, 2, \cdots, K+1)$ 时段行驶 $d_{ij}$ 需要时间 $c_{ijl}$.

假设车辆于 $t_i$ 时刻( $t_i \in [B_1, B_2]$) 从节点 $i$ 出发,总行驶时间为 $\sum_{l=1}^{K+1} \frac{d_{ij}^l}{d_{ij}} c_{ijl}$. 在第 1 时段行驶 $d_{ij}^1$ 所用时间为 $B_2 - t_i$,在第 $l(l \in [2,K])$ 时段行驶 $d_{ij}^l$ 所用时间为 $B_{l+1} - B_l$,则 $d_{ij}^1 = d_{ij}/c_{ij1} \times (B_2 - t_i)$, $d_{ij}^l = d_{ij}/c_{ijl} \times (B_{l+1} - B_l)$, $d_{ij}^{K+1} = d_{ij} - \sum_{l=1}^{K} d_{ij}^l = d_{ij} - d_{ij}/c_{ij1} \times (B_2 - t_i) - \sum_{l=2}^{K} d_{ij}/c_{ijl} \times (B_{l+1} - B_l)$, 因此车辆在第 $K+1$ 时段行驶 $d_{ij}^{K+1}$ 所用时间 $\frac{d_{ij}^{K+1}}{d_{ij}} c_{ij,K+1} = (1 - \frac{B_2 - t_i}{c_{ij1}} - \sum_{l=2}^{K} \frac{B_{l+1} - B_l}{c_{ijl}}) c_{ij,K+1}$. 则车辆从节点 $i$ 出发,到达节点 $j$ 的总行驶时间为 $(B_2 - t_i) + \sum_{k=2}^{K} (B_{k+1} - B_k) + (1 - \frac{B_2 - t_i}{c_{ij1}} - \sum_{l=2}^{K} \frac{B_{l+1} - B_l}{c_{ijl}}) c_{ij,K+1} = B_{K+1} - t_i + (1 - \frac{B_2 - t_i}{c_{ij1}} - \sum_{l=2}^{K} \frac{B_{l+1} - B_l}{c_{ijl}}) c_{ij,K+1}$. 到达节点 $j$ 的时刻 $t_j = t_i + B_{K+1} - t_i + (1 - \frac{B_2 - t_i}{c_{ij1}} - \sum_{l=2}^{K} \frac{B_{l+1} - B_l}{c_{ijl}}) c_{ij,K+1}$. 经整理得

$$t_j = \frac{c_{ij,K+1}}{c_{ij1}} t_i + B_{K+1} + (1 - \frac{B_2}{c_{ij1}} - \sum_{l=2}^{K} \frac{B_{l+1} - B_l}{c_{ijl}}) c_{ij,K+1} \tag{1}$$

由式( 1) 可知,车辆到达节点 $j$ 的时刻 $t_j$ 为车辆从节点 $i$ 出发时刻 $t_i$ 的增函数,满足 FIFO 准则. 在该式中,不涉及两节点间的距离和旅行速度,处理起来更容易. 当 $K=1$ 时, $\sum_{l=2}^{K} \frac{B_{l+1} - B_l}{C_{ijl}} = 0$, 该式变为跨单时段情形,有 $t_j = \frac{c_{ij2}}{c_{ij1}} t_i + (1 - \frac{c_{ij2}}{c_{ij1}}) B_2 + c_{ij2}$.

车辆从节点 $i$ 出发的时刻 $t_i$( $t_i \in [B_p, B_{p+1}]$) 需满足以下两个条件才能跨越 $K$ 个时段: 车辆从 $p$ 时段出发,且从 $p$ 时段到第 $p+K$ 时段未行驶完该路段. 经整理得 $B_{p+1} - c_{ijp} + \sum_{l=p+1}^{p+K-1} \frac{c_{ijp}}{c_{ijl}} (B_{l+1} - B_l) \leqslant t_i < B_{p+1}$. 当 $K = 1$ 时, $\sum_{l=p+1}^{p+K-1} \frac{c_{ijp}}{c_{ijl}} (B_{l+1} - B_l) = 0$, 变为跨单时段情形,有 $B_{p+1} - c_{ijp} \leqslant t_i < B_{p+1}$.

© 1994-2012 China Academic Journal Electronic Publishing House. All rights reserved.　http://www.cnki.net

车辆于时段 $p$ 从节点 $i$ 出发到达节点 $j$，会出现不跨时段和跨时段两种可能性，对应表达式为

$$t_j=\begin{cases} t_i+c_{ijp} & (B_p\leqslant t_i<B_{p+1}-c_{ijp},\ p=1,\cdots,P)\\ \dfrac{c_{ij,p+K}}{c_{ijp}}t_i+B_{p+K}+\left(1-\dfrac{B_{p+K}}{c_{ijp}}-\sum\limits_{l=p+1}^{p+K-1}\dfrac{B_{l+1}-B_l}{c_{ijl}}\right)c_{ij,p+K} & \left(B_{p+1}-c_{ijp}+\sum\limits_{l=p+1}^{p+K-1}\dfrac{c_{ij,p+K}}{c_{ijl}}(B_{l+1}-B_l)\leqslant t_i<B_{p+1},p=1,\cdots,P-K\right)\end{cases} \tag{2}$$

## 2 TDVRP 问题数学模型

文中考虑 $N$ 个顾客，$M$ 辆车的 TDVRP 问题，目标函数为最小化所有车辆的总行驶时间. 假设所有车辆同时于 $t_0(t_0\in[B_p,B_{p+1}])$ 时刻从车场(编号为 0) 出发，目标函数也可转换为最小化所有车辆服务完顾客后返回车场的时刻. 模型如下

$$\min\sum_{m=1}^{M}t_{N+m} \tag{3}$$

$$\text{s.t.}\quad\sum_{\substack{i=0\\ i\neq j}}^{N}\sum_{p=1}^{P}x_{ijp}=1\qquad j=1,\cdots,N+M \tag{4}$$

$$\sum_{\substack{j=1\\ i\neq j}}^{N+M}\sum_{p=1}^{P}x_{ijp}=1\qquad i=0,\cdots,N \tag{5}$$

$$t_j-Lx_{ijp}\geqslant S_{ijp}-L\qquad i=0,\cdots,N;\ j=1,\cdots,N+M;\ i\neq j;\ p=1,\cdots,P \tag{6}$$

$$S_{ijp}=\begin{cases} t_i+c_{ijp} & B_p\leqslant t_i<B_{p+1}-c_{ijp},\ p=1,\cdots,P;\\ \dfrac{c_{ij,p+K}}{c_{ijp}}t_i+B_{p+K}+\left(1-\dfrac{B_{p+K}}{c_{ijp}}-\sum\limits_{l=p+1}^{p+K-1}\dfrac{B_{l+1}-B_l}{c_{ijl}}\right)c_{ij,p+K} & B_{p+1}-c_{ijp}+\sum\limits_{l=p+1}^{p+K-1}\dfrac{c_{ij,p+K}}{c_{ijl}}(B_{l+1}-B_l)\leqslant t_i<B_{p+1},p=1,\cdots,P-K\end{cases} \tag{7}$$

$$\sum_{m=1}^{M}z_{im}=\begin{cases}1 & i=1,\cdots,N\\ M & i=1\end{cases} \tag{8}$$

$$\sum_{i=1}^{N}d_iz_{im}\leqslant Q\qquad m=1,\cdots,M \tag{9}$$

$$\sum_{j=1}^{N}y_{jim}=\sum_{j=1}^{N}y_{ijm}=z_{im},\quad i=0,\cdots,N;\ m=1,\cdots,M \tag{10}$$

$$t_i\geqslant 0\qquad i=1,\cdots,N+M \tag{11}$$

$$x_{ijp}\in\{0,1\}\qquad i=0,\cdots,N;\ j=1,\cdots,N+M;\ p=1,\cdots,P \tag{12}$$

$$y_{ijm}\in\{0,1\}\qquad i=0,\cdots,N;\ j=1,\cdots,N;\ m=1,\cdots,M \tag{13}$$

$$z_{im}\in\{0,1\},\quad i=0,\cdots,N;\ m=1,\cdots,M \tag{14}$$

目标函数(3) 为最小化所有车辆访问完所有顾客之后返回车场的时刻，其中 $t_{N+m}$ 表示第 $m(m\in[1,M])$ 辆车返回车场的时刻；约束(4) 保证在任一时段节点 $j$ 紧前只有一个节点，其中 $x_{ijp}=\begin{cases}1 & \text{若车辆在时段 } p \text{ 内从节点 } i \text{ 出发到节点 } j\\ 0 & \text{否则}\end{cases}$；约束(5) 保证在任一时段节点 $i$ 紧后只有一个节点；约束(6) 计算车辆从节点 $i$ 出发，到达节点 $j$ 的时刻；其中 $S_{ijp}$ 为车辆于 $p$ 时段从节点 $i$ 出发，到达节点 $j$ 的时刻，$L$ 为一非常大的正数，若 $x_{ijp}=1$，约束(6) 应取“=”号，此时 $t_j=S_{ijp}$；若 $x_{ijp}=0$，约束(6) 应取“>”号；约束(7) 分别考虑了跨时段与不跨时段两种情况计算 $S_{ijp}$；约束(8) 保证每个顾客只能被某一辆车访问，而车场则被所有车辆访问，其中 $z_{im}=\begin{cases}1 & \text{若第 } m \text{ 辆车在任意时段访问节点 } i\\ 0 & \text{否则}\end{cases}$；约束(9) 为车的能力约束，其中 $d_i$ 为顾客 $i$ 的需求量，$Q$ 为每辆车的装载能力；约束(10) 保证路径的连续性，其中 $y_{ijm}=\begin{cases}1 & \text{若第 } m \text{ 辆车在任意时段从节点 } i \text{ 出发到节点 } j\\ 0 & \text{否则}\end{cases}$；约束(11) 保证到达节点 $i$ 的时刻非负.

## 3 基于 VLSN 搜索技术的环状交换算法

基于 VLSN 搜索技术的环状交换算法包括顶层和底层两层策略：顶层构造环状交换辅助图，采用动态规划算法搜索环状交换邻域以得到每辆车服务的顾客集合；底层设计动态搜索算法用以安排每辆车的最佳顾客服务路线. 为了描述方便，文中只考虑跨单时段情形(跨多时段情形可类推).

© 1994-2012 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

### 3.1　顶层策略

对TDVRP问题而言,环状交换就是在不同车辆之间以环状方式交换一定数目的顾客,从而减少所有车辆的总返回车场时刻. 设$[I^1, \cdots, I^M]$表示环状交换前的一个可行路线集合,其中$I^i$表示第$i$辆车的路线,$J = \{J^1, J^2, \cdots, J^M\}$表示交换后的路线集合,$f(I)$表示路线$I$经过底层策略求解得到的车辆返回车场时刻,环状交换值为环状交换引起的车辆返回车场时刻的变化,表示为$\sum_{i=1}^{M}[f(J^i) - f(I^i)]$,只要当该变化值为负时才能改进.

为了更好地说明环状交换策略,文中构造环状交换辅助图,在交换过程中只考虑每条路线同时接收和移出一个顾客(交换多个顾客方法完全相同). 该图可表示为$G^p = (V^p, A^p, C^p)$,其中

阶段$p$: 表示第$p+1$条路线接收来自第$p$条路线移出的一个顾客,且第$p+1$条路线移出一个顾客;

$V^p$: 表示第$p$条路线的顾客集合,为了构成环状结构,令$V^{M+1} = V^1$;

状态$A^p = \{(i^p, j^{p+1}) : i^p \in V^p, j^{p+1} \in V^{p+1}$. 第$p+1$条路线接收来自第$p$条路线移出的顾客$i^p$,且第$p+1$条路线移出顾客$j^{p+1}$后,第$p+1$条路线对应车辆不会超载.$\}$

$C^p = \{c_{ij}^{pp+1} : (i^p, j^{p+1}) \in A^p\}$为弧费用,$c_{ij}^{pp+1} = f(I(j^{p+1}) + i^p - j^{p+1}) - f(I(j^{p+1}))$,表示第$p+1$条路线交换顾客前后引起的返回车场时刻变化.

对一个有$M$辆车,每辆车服务$n$个顾客的TDVRP问题,在每条路线只交换一个顾客的情况下,$G^p$中有$Mn^2$对弧. 由于每一对弧$(i^p, j^{p+1})$费用计算已非常复杂,即使在$M$、$n$很小的情况下,找到$G^p$中所有弧费用其代价非常大. 另外,在$G^p$中找负值环是一个NP难题[15]. 由于以上因素,使得环状交换邻域搜索非常困难. 文中利用动态规划算法搜索环状交换邻域,定义$T$为当前$p$个阶段所有可能的交换顾客序列,$F(p, T)$为阶段$p$交换序列为$T$对应的最小环状交换值.

初始化$p = 1$: $F(1, T) = C^1$;　　(15)

对$2 \leqslant p \leqslant M$: $F(p, T) = \min_{A^p \subset T}\{F(p-1, T - A^p) + C^p\}$　　(16)

### 3.2　底层策略

底层策略用以改进每辆车的路线,相当于求解一个时变旅行商问题(time dependent traveling salesman problem, TDTSP). Congram[14] 提出求解静态网络TSP问题的dynasearch策略,该策略包括3种算法: ds-2-opt、ds-2.5-opt和ds-3-opt,利用动态规划算法分别搜索TSP问题的2-opt, or-opt和3-opt邻域. 该策略的主要优点是利用动态规划算法搜索的快速性,能在多项式时间范围内搜索到指数大小的邻域.

本文利用该算法的思想,提出一类动态搜索算法计算时变网络中每辆车的最早返回车场时刻. 假设初始路径为$(\sigma_1, \sigma_2, \cdots, \sigma_{n+1})$,其中$\sigma_j$表示该路径第$j$个位置的节点,$\sigma_{n+1} = \sigma_1$. 为了表述方便,假设一天被分为2个时段,车辆从车场$\sigma_1$出发的时刻为$t_1(t_1 \in [B_1, B_2])$. $F(\sigma_j)$表示车辆从节点$\sigma_1$出发,到达节点$\sigma_j$的最早时刻,因不考虑在节点处停留,$F(\sigma_j)$也为车辆从节点$\sigma_j$出发的最早时刻. 在该子路径中加入节点$\sigma_{j+1}$,执行一系列独立的$k$-opt($k = 2, 3$)移动,得到$F(\sigma_{j+1})$,表示对应到达节点$\sigma_{j+1}$的最早时刻($j = 1, 2, \cdots, n$).

1) Ds-2-opt

2-opt移动以任意一条路径开始,断开其任意两边,以另外一种方式重新连接断开部分,构成一条新的路径,若车辆在该新路径行驶时间比原路径的时间短,则替换原路径. ds-2-opt算法对当前路径执行一系列独立2-opt移动.

初始条件: $F(\sigma_1) = t_1$;

$j = 1, 2$时,递归方程为

$$F(\sigma_{j+1}) = \begin{cases} F(\sigma_j) + c_{\sigma_j,\sigma_{j+1},1} & B_1 \leqslant F(\sigma_j) < B_2 - c_{\sigma_j,\sigma_{j+1},1} \\ \dfrac{c_{\sigma_j,\sigma_{j+1},2}}{c_{\sigma_j,\sigma_{j+1},1}} F(\sigma_j) + \left(1 - \dfrac{c_{\sigma_j,\sigma_{j+1},2}}{c_{\sigma_j,\sigma_{j+1},1}}\right) B_2 + c_{\sigma_j,\sigma_{j+1},2} & B_2 - c_{\sigma_j,\sigma_{j+1},1} \leqslant F(\sigma_j) < B_2 \\ F(\sigma_j) + c_{\sigma_j,\sigma_{j+1},2} & F(\sigma_j) \geqslant B_2 \end{cases} \quad (17)$$

© 1994-2012 China Academic Journal Electronic Publishing House. All rights reserved.　http://www.cnki.net

对 $j = 3, 4, \cdots, N$,在当前路径中有两种方式加入节点 $\sigma_{j+1}$:

方式 ①: 在节点 $\sigma_j$ 后加入节点 $\sigma_{j+1}$,并保持原路径顺序不变,如图 1 所示. $F_1(\sigma_{j+1})$ 表示第 $j+1$ 阶段对应方式 ① 到达节点 $\sigma_{j+1}$ 的时刻. 递归方程同式(17).

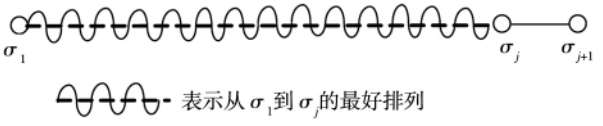


**图 1**　ds-2-opt 方式 ① 邻域变换示意图

Fig. 1 The first form for ds-2-opt

方式 ②: 在节点 $\sigma_j$ 后加入节点 $\sigma_{j+1}$,断开弧 $(\sigma_i, \sigma_{i+1})$ 和弧 $(\sigma_j, \sigma_{j+1})$,并执行 2-opt 移动($i = 1, \cdots, j-2$). 如图 2 所示. $F_{2,i}(\sigma_{j+1})$ 表示第 $j+1$ 阶段对应方式 ② 第 $i$ 种可能到达节点 $\sigma_{j+1}$ 的时刻. 变化后的路径统一用 $(e_1, e_2, \cdots, e_{j+1})$ 表示,$\forall k \in [1, i], e_k = \sigma_k$; $\forall k \in [i+1, j], e_k = \sigma_{j+i+1-k}$; $e_{j+1} = \sigma_{j+1}$.

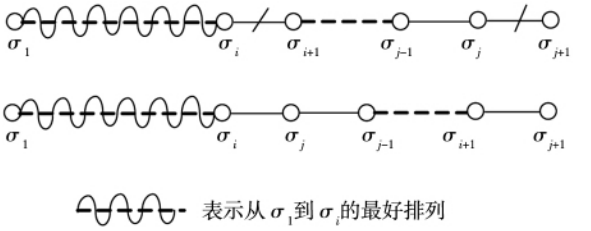


**图 2**　ds-2-opt 方式 ② 邻域变换示意图

Fig. 2 The second form for ds-2-opt

递归方程为

$$F_{2,i}(\sigma_{j+1}) = \begin{cases} F(\sigma_i) + \sum_{k=i}^{j} c_{e_k,e_{k+1},1} & B_1 \leqslant F(\sigma_i) < B_2 - \sum_{k=i}^{j} c_{e_k,e_{k+1},1} \\ t_{e_1,e_g} + t_{e_g,e_{g+1}} + t_{e_{g+1},e_{j+1}} & B_2 - \sum_{k=i}^{j} c_{e_k,e_{k+1},1} \leqslant F(\sigma_i) < B_2 \\ F(\sigma_i) + \sum_{k=i}^{j} c_{e_k,e_{k+1},2} & F(\sigma_i) \geqslant B_2 \end{cases} \tag{18}$$

式中 $t_{e_1,e_g}$ 表示跨时段前到达节点 $e_g$ 的时刻,其中 $e_g$ 为跨时段弧的前节点 ($g \in \{i, i+1, \cdots, j\}$),$t_{e_1,e_g} = F(\sigma_i) + \sum_{k=i}^{g-1} c_{e_k,e_{k+1},1}$ ($i = 1, \cdots, j-2$). 特别地,当 $g = i$ 时,$t_{e_1,e_g} = F(\sigma_i)$. $t_{e_g,e_{g+1}}$ 表示跨时段对应弧 $(e_g, e_{g+1})$ 的行驶时间,$t_{e_g,e_{g+1}} = B_2 - t_{e_1,e_g} + (1 - \frac{B_2 - t_{e_1,e_g}}{c_{e_g,e_{g+1},1}}) c_{e_g,e_{g+1},2}$. $t_{e_{g+1},e_{j+1}}$ 表示跨时段后对应子路径 $(e_{g+1}, \cdots, e_{j+1})$ 的行驶时间,$t_{e_{g+1},e_{j+1}} = \sum_{k=g+1}^{j} c_{e_k,e_{k+1},2}$.

第 $j+1$ 阶段的最优值为方式 ① 和 ② 中到达节点 $\sigma_{j+1}$ 的最早时刻,即 $F(\sigma_{j+1}) = \min\{F_1(\sigma_{j+1}), \min_{i=1,\cdots,j-2} F_{2,i}(\sigma_{j+1})\}$.

2) Ds-2.5-opt

Ds-2.5-opt 邻域将 2-opt 邻域和 or-opt 邻域相结合. 在 ds-2-opt 的基础上,采用前向插入和后向插入策略. 动态规划搜索过程和初始条件同 ds-2-opt.

对 $j = 3, 4, \cdots, N$,当前路径有 4 种方式加入节点 $\sigma_{j+1}$,其中方式 ① 和 ② 同 ds-2-opt 的方式 ① 和 ②.

方式 ③: 前向插入. 在节点 $\sigma_j$ 后加入节点 $\sigma_{j+1}$,将节点 $\sigma_j$ 插入到节点 $\sigma_i$ 和节点 $\sigma_{i+1}$ 之间 ($i = 1, \cdots, j-2$),如图 3 所示. $F_{3,i}(\sigma_{j+1})$ 表示第 $j+1$ 阶段对应方式 ③ 第 $i$ 种可能到达节点 $\sigma_{j+1}$ 的时刻.

方式 ④: 后向插入. 在节点 $\sigma_j$ 后加入节点 $\sigma_{j+1}$,将节点 $\sigma_{i+1}$ 插入到节点 $\sigma_j$ 和节点 $\sigma_{j+1}$ 之间 ($i = 1, \cdots, j-2$),如图 4 所示. $F_{4,i}(\sigma_{j+1})$ 表示第 $j+1$ 阶段对应方式 ④ 第 $i$ 种可能到达节点 $\sigma_{j+1}$ 的时刻.

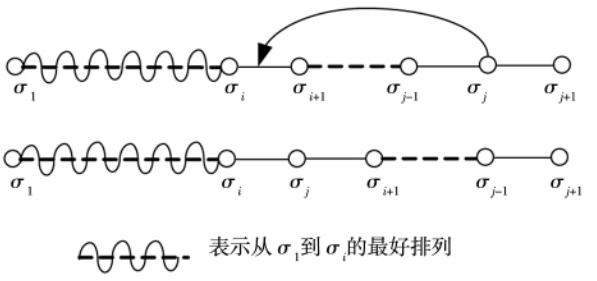


**图 3**　ds-2.5-opt 方式 ③ 邻域变换示意图

Fig. 3 The third form for ds-2.5-opt

© 1994-2012 China Academic Journal Electronic Publishing House. All rights reserved.　http://www.cnki.net

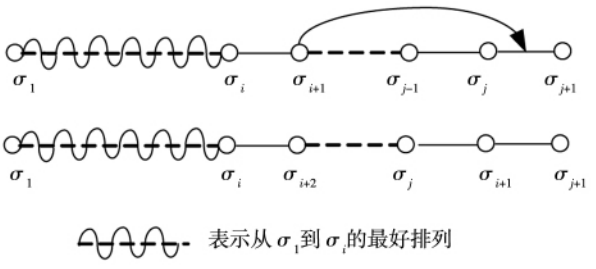


**图 4**　ds-2.5-opt 方式 ④ 邻域变换示意图

Fig. 4 The forrth form for ds-2.5-opt

针对方式 ③、④, 移动后的路径统一用( $e_1$, $e_2$, …, $e_{j+1}$) 表示, 满足 $\forall k \in [1,i]$, $e_k = \sigma_k$, $e_{j+1} = \sigma_{j+1}$. 针对方式③, 满足 $e_{i+1} = \sigma_j$, $\forall k \in [i+2,j]$, $e_k = \sigma_{k-1}$; 针对方式④, 满足 $e_j = \sigma_{i+1}$, $\forall k \in [i+1,j-1]$, $e_k = \sigma_{k+1}$. $F_{3,i}(\sigma_{j+1})$ 和 $F_{4,i}(\sigma_{j+1})$ 可通过式( 18) 计算得到.

$$F(\sigma_4) = \begin{cases} F(\sigma_3) + c_{\sigma_3,\sigma_4,1} \\ \dfrac{c_{\sigma_3,\sigma_4,2}}{c_{\sigma_3,\sigma_4,1}} F(\sigma_3) + \left(1 - \dfrac{c_{\sigma_3,\sigma_4,2}}{c_{\sigma_3,\sigma_4,1}}\right) B_2 + c_{\sigma_3,\sigma_4,2} \\ F(\sigma_3) + c_{\sigma_3,\sigma_4,2} \end{cases} \begin{array}{l} B_1 \leqslant F(\sigma_3) < B_2 - c_{\sigma_3,\sigma_4,1} \\ B_2 - c_{\sigma_3,\sigma_4,1} \leqslant F(\sigma_3) < B_2 \\ F(\sigma_3) \geqslant B_2 \end{array} \quad (19)$$

对 $j = 4, \cdots, n$, 当前路径有 5 种方式加入节点 $\sigma_{j+1}$:

方式 ①: 同 ds-2-opt 和 ds-2.5-opt 的方式 ①.

方式 ② ~ ⑤: 节点 $\sigma_j$ 后加入节点 $\sigma_{j+1}$, 并删除三条弧( $\sigma_h$, $\sigma_{h+1}$) ,( $\sigma_i$, $\sigma_{i+1}$) 和( $\sigma_j$, $\sigma_{j+1}$) , 执行 3-opt 移动, 其中 $1 \leqslant h \leqslant i-2 \leqslant j-4$, 分别采用四种不同的方式连接. 其邻域变化如图 5 所示. 移动后的路径统一用( $e_1$, $e_2$, …, $e_{j+1}$) 表示. $F_{l,h,i}(\sigma_{j+1})$ ( $l = 2,3,4,5$, $1 \leqslant h \leqslant i-2 \leqslant j-4$) 表示第 $j+1$ 阶段对应方式 ② ~ ⑤ 各种可能到达节点 $\sigma_{j+1}$ 的时刻. 递归方程为

$$F_{l,h,i}(\sigma_{j+1}) = \begin{cases} F(\sigma_h) + \sum_{k=h}^{j} c_{e_k,e_{k+1},1} \\ t_{e_1,e_g} + t_{e_g,e_{g+1}} + t_{e_{g+1},e_{j+1}} \\ F(\sigma_h) + \sum_{k=h}^{j} c_{e_k,e_{k+1},2} \end{cases} \begin{array}{l} B_1 \leqslant F(\sigma_h) < B_2 - \sum_{k=h}^{j} c_{e_k,e_{k+1},1} \\ B_2 - \sum_{k=h}^{j} c_{e_k,e_{k+1},1} \leqslant F(\sigma_h) < B_2 \\ F(\sigma_h) \geqslant B_2 \end{array} \quad (20)$$

式中 $t_{e_1,e_g}$ 表示在跨时段前到达节点 $e_g$ 的时刻, 其中 $e_g$ 为跨时段对应弧的前节点( $g \in \{h, h+1, \cdots, j\}$) , $t_{e_1,e_g} = F(\sigma_h) + \sum_{k=h}^{g-1} c_{e_k,e_{k+1},1}$ ( $h = 1, \cdots, j-2$) . 特别地, 当 $g = h$ 时, $t_{e_1,e_g} = F(\sigma_h)$. $t_{e_g,e_{g+1}}$ 表示跨时段对应弧( $e_g$, $e_{g+1}$) 的行驶时间, $t_{e_g,e_{g+1}} = B_2 - t_{e_1,e_g} + \left(1 - \dfrac{B_2 - t_{e_1,e_g}}{c_{e_g,e_{g+1},1}}\right) c_{e_g,e_{g+1},2}$. $t_{e_{g+1},e_{j+1}}$ 表示跨时段后对应子路径( $e_{g+1}, \cdots, e_{j+1}$) 的行驶时间, $t_{e_{g+1},e_{j+1}} = \sum_{k=g+1}^{j} c_{e_k,e_{k+1},2}$.

第 $j+1$ 阶段的最优值为方式 ① ~ ⑤ 各种可能性中到达节点 $\sigma_{j+1}$ 的最早时刻, 即 $F(\sigma_{j+1}) = \min\{F_1(\sigma_{j+1}), \min_{\substack{l=2,3,4,5 \\ h=1,\cdots,j-4 \\ i=h+2,\cdots,j-2}} F_{l,h,i}(\sigma_{j+1})\}$.

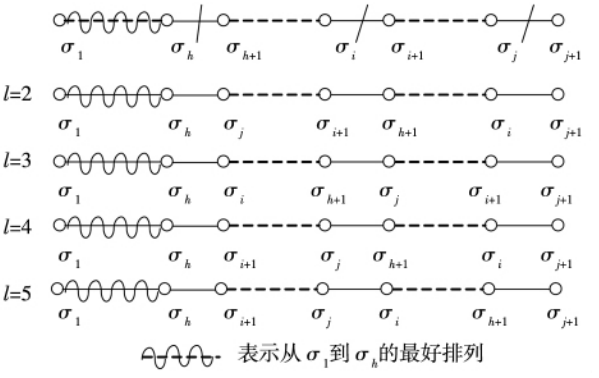


**图 5**　ds-3-opt 方式 ② ~ ⑤ 邻域变换

Fig. 5 The 2nd ~ 5th form for ds-3-opt

3) ds-3-opt

3-opt 移动从任意一条路径开始, 断开其任意 3 条弧, 重新连接断开部分, 构成一条新路径. 算法前 3 个节点初始化同 ds-2-opt 的初始化.

$j = 3$, 在节点 $\sigma_3$ 后加入节点 $\sigma_4$, 有

$$B_1 \leqslant F(\sigma_3) < B_2 - c_{\sigma_3,\sigma_4,1}$$
$$B_2 - c_{\sigma_3,\sigma_4,1} \leqslant F(\sigma_3) < B_2 \quad (19)$$
$$F(\sigma_3) \geqslant B_2$$

© 1994-2012 China Academic Journal Electronic Publishing House. All rights reserved.　http://www.cnki.net

在最后阶段 $j = n$ 时，得到最优值 $F(\sigma_{n+1})$，以上三种算法均可通过动态规划回溯找到最优路径. 对于 ds-2-opt 和 ds-2.5-opt，其计算复杂度为 $O(n^2)$；对于 ds-3-opt，其计算复杂度为 $O(n^3)$.

### 3.3　底层与顶层策略相结合

环状交换动态规划算法将顶层策略和底层策略相结合，该算法不仅能对某条路线的顾客序列搜索呈指数形式大小的邻域，还能在不同路线之间交换顾客，从而在很大程度上扩展算法的邻域搜索空间. 算法步骤如下:

**步骤 1**　初始化，采用最近邻算法[1] 构造车辆的初始顾客分配方案.

**步骤 2**　采用动态搜索算法安排每辆车的路线，计算对应的最早返回车场时刻.

**步骤 3**　构造 $G^p$. 定义节点和弧. 采用动态搜索算法计算每对弧的费用.

**步骤 4**　while $G^p$ 中包含负值环，采用动态规划回溯找到该环，执行环状交换，更新路线.

采用动态搜索算法计算每辆车的最早返回车场时刻.

更新 $G^p$: 重新定义节点和弧. 采用动态搜索算法计算每对弧的费用.

End

**步骤 5**　输出最终路线，停止.

## 4　改进环状交换策略

### 4.1　顶层加入虚拟顾客

环状交换算法要求每条路线中都必须移出一个顾客至下一条路线，且接收来自上一条路线的一个顾客，这样可能会破坏原有路线的最优性. 为了避免这种现象发生，本节在原有环状交换的基础上，每条路线里加入一个或者多个虚拟顾客(为了说明方便，这里只加入一个虚拟顾客). 该虚拟顾客实际并不存在，其目的是为了构造环状结构，实际上只交换了部分(或所有) 路线里的顾客. 加入虚拟顾客能够在保证不破坏原有路线顾客组合的最佳性前提下，保证环状交换的可行性，使得交换过程更加柔性化. 如图 6 所示.

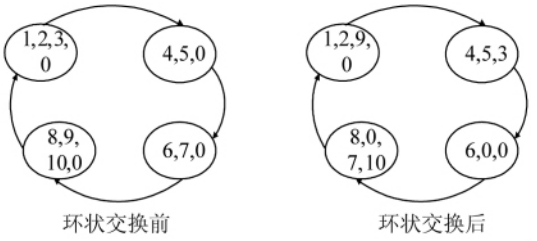


**图 6　带有虚拟顾客环状交换示意图**

Fig. 6 The illustration of cycle transfer for adding dummy customer strategy

该算法需要在各条路线里加入一虚拟顾客，设定该顾客标号为 0. 该虚拟顾客只是加入到弧中，增加原有的状态，但在实际计算弧费用的时候并不参与运算. 阶段 $p$ ($p = 1, \cdots, M$) 存在四种可能性，分别为 $(x, y)$，$(0, y)$，$(x, 0)$，$(0, 0)$. 其中 $x$ 为从第 $p$ 条路线移出并加入到第 $p+1$ 条路线的实顾客(即 $x \neq 0$)，$y$ 为第 $p+1$ 条路线移出的实顾客(即 $y \neq 0$).

(1) $(x, y)$ 表示实进实出，该情形与不带虚拟顾客的环状交换相同.

(2) $(0, y)$ 表示虚进实出，第 $p$ 条路线没有顾客加入到第 $p+1$ 条路线中，但后者有顾客 $y$ 移出.

(3) $(x, 0)$ 表示实进虚出，第 $p$ 条路线有顾客 $x$ 加入到第 $p+1$ 条路线中，但后者没有顾客移出.

(4) $(0, 0)$ 表示虚进虚出，第 $p+1$ 条路线保持原有状态.

### 4.2　底层嵌入 insert

由于在 $G^p$ 中每对弧的费用计算对应求解一个 TDTSP 问题，且 $G^p$ 里弧的数目很大，导致算法运行时间很长. 在后面实验 2 中发现环状交换动态规划算法运行时间非常长，尤其底层算法为 ds-3-opt 时最长，这是因为算法运行时间主要取决于底层算法的运行时间. 本节中提出一种底层嵌入 insert 的环状交换动态规划算法，在计算弧费用时，环状交换后的车辆返回车场时刻由动态搜索算法改为 insert 计算，这样将在很大程度上节省运行时间，同时，由于每更新一次路线时，其初始车辆最早返回车场时刻采用动态搜索算法求解，最终解的质量可以得到保证.

假设在第 $p$ 阶段，路线 $I(j^{p+1})$ 中顾客初始顺序为 $\{X_1, X_2, X_3, \cdots, X_{i-1}, X_i, X_{i+1}, \cdots, X_n\}$，共有 $n$ 个顾客. 在该阶段，$A^p$ 里有一对弧 $(Y_j, X_i)$，其中 $Y_j$ 为路线 $I(j^p)$ 中第 $j$ 个位置的顾客，在执行

© 1994-2012 China Academic Journal Electronic Publishing House. All rights reserved.　http://www.cnki.net

环状交换时从路线 $I(j^p)$ 中移出,并加入到路线 $I(j^{p+1})$ 中. 在应用 insert 时,将 $X_i$ 顾客从路线 $I(j^{p+1})$ 中删除,并将 $Y_j$ 加入到该排列的最后一个位置,依次往前插入,找到一个最小值. 该最小值作为环状交换后的车辆返回车场时刻. 图 7 演示了这一过程.

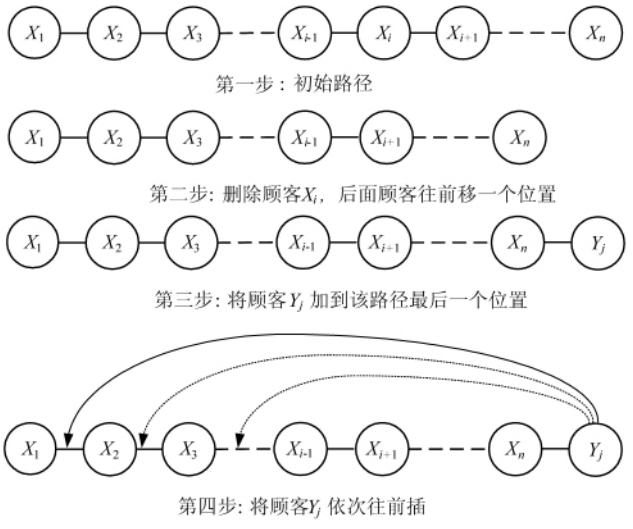


**图 7　insert 计算示意图**

Fig. 7 The illustration of calculation for embedding insert strategy

算法步骤如下:

**步骤 1**　初始化,采用最近邻算法[1] 构造车辆的初始顾客分配方案.

**步骤 2**　采用动态搜索算法安排每辆车的路线,并计算对应的最早返回车场时刻.

**步骤 3**　构造 $G^p$. 定义节点和弧. 用 insert 计算每对弧的费用.

**步骤 4**　while $G^p$ 中包含负值环,采用动态规划回溯找到该环,执行环状交换,更新路线.

采用动态搜索算法计算每辆车的最早返回车场时刻.

更新 $G^p$: 重新定义节点和弧. 采用 insert 策略计算每对弧的费用.

End

**步骤 5**　输出最终路线,停止.

## 5　实验及算法性能比较

本文对几类算法分别进行了测试比较,所采用的 10 组不同规模数据随机产生,其中包括顾客数、车辆数和在不同时段节点(包括顾客和车场) 间的车辆行驶时间. 以下实验均采用这些数据以便于比较. 所有程序均用C++ 编写,并在 PentiumⅣ 主频 1.6GHz 的计算机上进行实验仿真.

实验 1: 对环状交换动态规划算法进行测试,采用求解 TDVRP 问题的最近邻算法[1] 构造车辆的初始顾客分配方案,底层算法为动态搜索算法. 表 1 记录了环状交换动态规划算法与最近邻算法在不同实验规模下的实验结果,其中包括了算法的目标函数值(单位: h) 和算法运行时间(单位: s) .

由表 1,不难看出除 10 × 3 这组数据外,在同一数据规模下环状交换动态规划算法与最近邻算法目标函数值完全相同,环状交换并没有达到真正交换的目的. 但就 10 × 3 这组数据,环状交换动态规划算法改进了最近邻算法. 该组数据通过最近邻算法得到的车辆安排路线为 0 – 4 – 1 – 8 – 0 – 10 – 3 – 7 – 0 – 2 – 5 – 9 – 6 – 0. 环状交换动态规划算法共进行了 2 次循环,如表 2 所示.

实验 2: 对带虚拟顾客的环状交换动态规划算法进行测试,底层算法为动态搜索算法. 表 3 记录了该算法与最近邻算法在不同实验规模下的实验结果,其中包括了算法的目标函数值(单位: h) 和算法运行时间(单位: s) .

通过表 3,可以得出如下结论:

(1) 当环状交换加入虚拟顾客后,目标函数值大大改进. 虚拟顾客不仅没有破坏原有路线的最佳顾客组合,还能保证环状交换可行. 底层为 ds-2-opt、ds-2.5-opt、ds-3-opt 对应的平均目标函数值分别改进最近邻算法 28.30%、32.15% 和 35.20%.

(2) 最近邻算法与带虚拟顾客的环状交换动态规划算法相比,明显后者运行时间远远大于前者,且底层为 ds-3-opt 的运行时间最长. 当数据规模为 200 × 35 时,底层为 ds-3-opt 的环状交换动态规划算法运行时间约 17min.

© 1994-2012 China Academic Journal Electronic Publishing House. All rights reserved.　http://www.cnki.net

**表 1　环状交换动态规划算法性能**

Table 1 The performance of cycle transfer dynamic programming algorithms

| 实验规模（顾客数 × 车辆数） | 环状交换动态规划算法 | | | | | | 最近邻算法 | |
|---|---|---|---|---|---|---|---|---|
| | ds-2-opt | | ds-2.5-opt | | ds-3-opt | | | |
| | 目标函数值 | 运行时间 | 目标函数值 | 运行时间 | 目标函数值 | 运行时间 | 目标函数值 | 运行时间 |
| 5 × 2 | 23.48 | 0.01 | 23.48 | 0.01 | 23.48 | 0.01 | 23.48 | 0.00 |
| 10 × 3 | 33.90 | 0.05 | 33.90 | 0.06 | 33.90 | 1.00 | 36.59 | 0.01 |
| 20 × 5 | 60.54 | 0.02 | 60.54 | 0.04 | 60.54 | 1.00 | 60.54 | 0.02 |
| 30 × 6 | 75.21 | 0.20 | 75.21 | 0.41 | 75.21 | 0.87 | 75.21 | 0.14 |
| 50 × 8 | 104.11 | 0.51 | 104.11 | 0.98 | 104.11 | 1.38 | 104.11 | 0.36 |
| 60 × 10 | 169.28 | 1.30 | 169.28 | 1.85 | 169.28 | 2.77 | 169.28 | 0.77 |
| 100 × 15 | 189.89 | 2.88 | 189.89 | 3.50 | 189.89 | 3.90 | 189.89 | 1.20 |
| 150 × 22 | 278.34 | 4.64 | 278.34 | 6.91 | 278.34 | 8.42 | 278.34 | 2.09 |
| 200 × 35 | 367.30 | 6.79 | 367.30 | 9.43 | 367.30 | 12.75 | 367.30 | 3.18 |
| 250 × 40 | NA | | NA | | NA | | NA | |

**注**: NA 表示不能运行计算

**表 2　10 × 3 数据的环状交换过程**

Table 2 The process of cycle transfer for data 10 × 3

| 环状交换次数 | 路线安排 | 环状交换序列 | 目标函数值 |
|---|---|---|---|
| 0 | 0 – 4 – 1 – 8 – 0 – 10 – 3 – 7 – 0 – 2 – 5 – 9 – 6 – 0 | | 36.59 |
| 1 | 0 – 4 – 5 – 8 – 0 – 10 – 1 – 7 – 0 – 2 – 3 – 9 – 6 – 0 | 1 – 3 – 5 – 1 | 34.93 |
| 2 | 0 – 4 – 5 – 3 – 0 – 10 – 1 – 8 – 0 – 2 – 7 – 9 – 6 – 0 | 8 – 7 – 3 – 8 | 33.90 |

**表 3　带虚拟顾客的环状交换动态规划算法性能**

Table 3 The performance of cycle transfer dynamic programming algorithms with dummy customer

| 实验规模（顾客数 × 车辆数） | 最近邻算法 | | 带虚拟顾客的环状交换动态规划算法 | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | | ds-2-opt | | ds-2.5-opt | | ds-3-opt | |
| | 目标函数值 | 运行时间 | 目标函数值 | 运行时间 | 目标函数值 | 运行时间 | 目标函数值 | 运行时间 |
| 5 × 2 | 23.48 | 0.00 | 22.53 | 1.99 | 22.53 | 2.06 | 22.53 | 2.80 |
| 10 × 3 | 36.59 | 0.01 | 30.41 | 5.64 | 30.41 | 5.60 | 30.41 | 7.92 |
| 20 × 5 | 60.54 | 0.02 | 43.70 | 13.00 | 41.96 | 25.71 | 41.96 | 30.64 |
| 30 × 6 | 75.21 | 0.14 | 45.59 | 35.63 | 44.31 | 50.01 | 44.31 | 104.92 |
| 50 × 8 | 104.11 | 0.36 | 69.82 | 72.91 | 65.74 | 96.82 | 65.74 | 253.40 |
| 60 × 10 | 169.28 | 0.77 | 105.33 | 100.54 | 100.92 | 148.29 | 96.18 | 305.00 |
| 100 × 15 | 189.89 | 1.20 | 123.39 | 193.77 | 114.05 | 262.32 | 102.99 | 544.37 |
| 150 × 22 | 278.34 | 2.09 | 221.24 | 261.18 | 209.52 | 334.81 | 202.36 | 715.59 |
| 200 × 35 | 367.30 | 3.18 | 273.45 | 478.59 | 255.86 | 571.27 | 239.00 | 1 042.18 |
| 250 × 40 | NA | | NA | | NA | | NA | |
| 平均值 | 144.97 | 0.86 | 103.94 | 129.25 | 98.36 | 166.32 | 93.94 | 334.09 |

**注**: NA 表示不能运行计算

© 1994-2012 China Academic Journal Electronic Publishing House. All rights reserved.　http://www.cnki.net

实验 3: 对带虚拟顾客的环状交换动态规划算法进行测试,底层嵌入 insert 策略. 表 4 记录了不同实验规模下的实验结果,其中包括了算法的目标函数值( 单位: h) 和算法运行时间( 单位: s) .

**表 4　带虚拟顾客的环状交换动态规划算法性能( 底层嵌入 insert 策略)**

Table 4 The performance of cycle transfer dynamic programming algorithms with dummy customer ( embedding insert strategy in lower level)

| 实验规模 ( 顾客数 × 车辆数) | 无 insert 策略 | | | | | | 底层嵌入 insert 策略 | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | ds-2-opt | | ds-2.5-opt | | ds-3-opt | | ds-2-opt | | ds-2.5-opt | | ds-3-opt | |
| | 目标函数值 | 运行时间 | 目标函数值 | 运行时间 | 目标函数值 | 运行时间 | 目标函数值 | 运行时间 | 目标函数值 | 运行时间 | 目标函数值 | 运行时间 |
| 5 × 2 | 22.53 | 1.99 | 22.53 | 2.06 | 22.53 | 2.80 | 21.93 | 1.50 | 21.93 | 1.50 | 21.93 | 1.50 |
| 10 × 3 | 30.41 | 5.64 | 30.41 | 5.60 | 30.41 | 7.92 | 28.55 | 2.72 | 28.55 | 2.80 | 28.55 | 2.75 |
| 20 × 5 | 43.70 | 13.00 | 41.96 | 25.71 | 41.96 | 30.64 | 43.70 | 6.39 | 41.96 | 10.43 | 41.96 | 15.78 |
| 30 × 6 | 45.59 | 35.63 | 44.31 | 50.01 | 44.31 | 104.92 | 46.74 | 16.08 | 44.31 | 22.58 | 44.31 | 29.66 |
| 50 × 8 | 69.82 | 72.91 | 65.74 | 96.82 | 65.74 | 253.40 | 72.51 | 25.99 | 67.02 | 31.06 | 67.02 | 41.84 |
| 60 × 10 | 105.33 | 100.54 | 100.92 | 148.29 | 96.18 | 305.00 | 110.92 | 32.33 | 104.46 | 42.77 | 96.18 | 67.32 |
| 100 × 15 | 123.39 | 193.77 | 114.05 | 262.32 | 102.99 | 544.37 | 128.26 | 41.58 | 116.63 | 56.12 | 105.71 | 80.55 |
| 150 × 22 | 221.24 | 261.18 | 209.52 | 334.81 | 202.36 | 715.59 | 228.33 | 60.23 | 213.52 | 82.49 | 203.89 | 105.47 |
| 200 × 35 | 273.45 | 478.59 | 255.86 | 571.27 | 239.00 | 1 042.18 | 273.45 | 91.66 | 260.41 | 124.80 | 245.31 | 189.61 |
| 250 × 40 | NA | | NA | | NA | | NA | | NA | | NA | |
| 平均值 | 103.94 | 129.25 | 98.36 | 166.32 | 93.94 | 334.09 | 106.04 | 30.92 | 99.86 | 41.61 | 94.98 | 59.38 |

**注**: NA 表示不能运行计算

由表 4,可以得到如下结论:

( 1) 在 5 × 2,10 × 3,20 × 5 三组小规模数据下,底层带有 insert 策略得到的目标函数值比不带 insert 策略要好. 原因在于小规模数据下,分配给每辆车的顾客数只满足动态搜索算法的初始条件,并不能起到真正 2-opt、or-opt、3-opt 移动的目的. 但 insert 策略可以对其邻域进行搜索,因此得到的解更好. 随着数据规模的扩大,动态搜索算法进行交换,此时 insert 策略得到的目标函数值不如动态搜索算法,但最大相差不超过 2%.

( 2) 就运行时间而言,明显底层嵌入 insert 策略的运行时间比无 insert 策略的运行时间要少很多. 在既考虑解质量又考虑运行时间的情况下,底层嵌入 insert 策略、顶层加入虚拟顾客的环状交换动态规划算法是个非常不错的选择.

## 6　结束语

文中对 TDVRP 问题建立了数学模型,提出满足 FIFO 准则的时变处理方法,并采用基于 VLSN 搜索技术的环状交换动态规划算法求解该问题. 基于 VLSN 搜索技术的环状交换动态规划算法能够在多条路线之间交换顾客,改变了传统只针对某条路线进行改进的邻域搜索思想. 在此基础上提出两类改进策略: 加入虚拟顾客的改进策略在保持原有顾客组合最佳性的同时,还能保持环状交换的可行性,使得环状交换更加柔性化; 底层嵌入 insert 的改进策略在保证最终解的质量前提下大幅度降低了算法运行时间. 实验结果验证了所提算法的有效性.

**参 考 文 献:**


[1] Malandraki C, Daskin M S. Time dependent vehicle routing problems: Formulations, properties and heuristic algorithms [J]. Transportation Science, 1992, 26( 3) : 185 – 200.

[2] Malandraki C, Dial R B. A restricted dynamic programming heuristic algorithm for the time dependent traveling salesman problem [J]. European Journal of Operational Research, 1996, 90( 1) : 45 – 55.

[3] Ahn B H, Shin J Y. Vehicle-routing with time windows and time-varying congestion [J]. Journal of the Operational Research Society, 1991, 42( 5) : 393 – 400.

[4] Hill A V, Benton W C. Modeling intra-city time-dependent travel speeds for vehicle scheduling problems [J]. Journal of the Operational Research Society, 1992, 43( 4) : 343 – 351.

[5] Horn M E T. Efficient modeling of travel in networks with time-varying link speeds [J]. Networks, 2000, 36( 2) : 80 – 90.


© 1994-2012 China Academic Journal Electronic Publishing House. All rights reserved.　http://www.cnki.net

[6] Ichoua S, Gendreau M, Potvin J Y. Vehicle dispatching with time-dependent travel times [J]. European Journal of Operational Research, 2003, 144(2): 379 – 396.

[7] Fleischmann B, Gietz M, Gnutzmann S. Time-varying travel times in vehicle routing [J]. Transportation Science, 2004, 38(2): 160 – 173.

[8] Haghani A, Jung S. A dynamic vehicle routing problem with time-dependent travel times [J]. Computer & Operations Research, 2005, 32(11): 2959 – 2986.

[9] Donati A V, Montemanni R, Casagrande N, et al. Time dependent vehicle routing problem with a multi ant colony system [J]. European Journal of Operational Research, 2008, 185(3): 1174 – 1191.

[10] 李妍峰, 李 军, 赵 达. 用动态搜索算法求解时间依赖型旅行商问题 [J]. 西南交通大学学报(自然科学版), 2008,43(2): 187 – 193.
Li Yanfeng, Li Jun, Zhao Da. Dynasearch algorithms for solving time dependent traveling salesman problem [J]. Journal of Southwest Jiaotong University, 2008,43(2): 187 – 193. (in Chinese)

[11] 李妍峰, 李 军, 高自友. 时变网络环境下旅行商问题研究 [J]. 系统工程学报, 2010, 25(5): 585 – 591.
Li Yanfeng, Li Jun, Gao Ziyou. Traveling salesman problem in time varying network [J]. Journal of Systems Engineering, 2010, 25(5): 585 – 591. (in Chinese)

[12] 魏 航. 一种求解时变条件下有宵禁限制最短路的算法 [J]. 管理科学学报, 2009, 12(1): 9 – 17.
Wei Hang. An approach for time-varying shortest path problem with curfews [J]. Journal of Management Sciences in China, 2009, 12(1): 9 – 17. (in Chinese)

[13] 魏 航, 李 军, 蒲 云. 时变条件下有害物品运输的路径问题研究 [J]. 系统工程理论与实践, 2006, 26(10): 107 – 112.
Wei Hang, Li Jun, PU Yun. Route planning for hazardous materials transportation in time-varying network [J]. Systems Engineering: Theory & Practice, 2006, 26(10): 107 – 112. (in Chinese)

[14] Congram R K. Polynominally searchable exponential neighborhoods for sequencing problems in combinatorial optimization [D]. Southampton: University of Southampton, 2000.

[15] Ahuja R K, Ergum O, Orlin J B, et al. A survey of very large-scale neighborhood search techniques [J]. Discrete Applied Mathematics, 2002, 123(1 – 3): 75 – 102.

[16] Thompson P M, Psaraftis H N. Cyclic transfer algorithms for multivehicle routing and scheduling problems [J]. Operations Research, 1993, 41: 935 – 946.

# Very large scale neighborhood search algorithm for solving time dependent vehicle routing problem

*LI Yan-feng*[1], *LI Jun*[1], *GAO Zi-you*[2]

1. School of Economics and Management, Southwest Jiaotong University, Chengdu 610031, China;
2. Institute of System Science, School of Traffic and Transportation, Beijing Jiaotong University, Beijing 100044, China

**Abstract**: Time dependent vehicle routing problem was formulated by dealing with time periods crossing with first-in first-out property, and a novel intelligent optimization algorithm based on very large scale neighborhood search technology was developed to solve it. In the upper level dynamic programming algorithm was adopted to search cycle transfer neighborhood so as to obtain the optimal customer set for each vehicle, while in the lower level dynasearch heuristics was developed to rank the customers in each vehicle. Based on this, two improvements were given: adding dummy customer in the upper level and embedding insert in the lower level. The efficiency of the method are testified with the results of extensive computational tests.

**Key words**: time dependent vehicle routing problem; first-in first-out; very large scale neighborhood search; dynasearch algorithm

© 1994-2012 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net