第 19 卷 第 1 期　　控 制 与 决 策　　2004 年 1 月
Vol 19 No. 1　　Control and Decision　　Jan. 2004

文章编号: 1001-0920(2004)01-0057-04

求解 PDPTW 问题的一种快速禁忌搜索算法

贾永基, 谷寒雨, 席裕庚
(上海交通大学 自动化研究所, 上海 200030)

摘　要: 提出一种解决实际规模和复杂度的 PDPTW 问题的快速禁忌搜索算法. 该算法分为构造初始解和改进解两个阶段: 在第 1 阶段, 使用插入算法来构造一个尽可能好的初始解; 在第 2 阶段, 使用禁忌搜索算法来改进得到的解. 最后构造了两个实际规模和复杂度的例子, 测试结果表明该算法对于求解此类 PDPTW 问题是有效的.

关键词: 装卸货问题; NP-难问题; 禁忌搜索; 时间窗口
中图分类号: TP301　　**文献标识码**: A

Quick taboo search algorithm for solving PDPTW problem

JIA Yong-ji, GU Han-yu, XI Yu-geng
(Institute of Automation, Shanghai Jiaotong University, Shanghai 200030, China. Correspondent: JIA Yong-ji, E-mail: ferrero@sjtu.edu.cn)

Abstract: A quick taboo search algorithm is proposed to solve the PDPTW with the reality scale and complexity. The approach is composed of two parts: Creating the initial solution and improving the solution. In the first phase, the insert algorithm is used to create the initial solution and in the second phase, the taboo search is applied to improve the solution. Two cases with the reality scale and complexity are created to test the algorithm. The results indicate that the proposed algorithm is effective and quick to solve such PDPTW problems.

Key words: PDPTW; NP-hard; taboo search; time windows

1 引　　言

带有时间窗口的装卸货问题(PDPTW)是为一个车队安排最优的路线来满足客户的运输需求, 是一类具有现实意义的组合优化问题[1]. 每个运输需求包括一个装货点、一个卸货点以及所运货物的重量和类型; 车队里每辆车从车库出发, 沿优化的路线为客户服务并最终返回车库; 每辆车有最大装货量和可装货物类型的限制; 所有车库、装货点和卸货点的地理位置可以不同, 但必须在规定的时间窗口范围内访问.

PDPTW 是VRPTW 的一般性推广, VRP 又是 TSP 的一般性推广, 而 TSP 已被证明是一个NP-hard 问题, 因此 PDPTW 也是NP-hard 问题. 通常情况下, 这类问题的规模较大, 复杂度很高, 有许多实际的约束条件, 解决这类问题的难度相当大.

PDPTW 问题在实际生活中应用很广, 从车辆路径规划、飞机航线规划, 到VLSI电路设计、柔性制造系统等. 合理地解决此类问题能节省资源和能源, 缩短工作时间, 提高工作效率, 从而大大降低成本. 但由于它的复杂性, 目前这方面的研究相对较少, 特别是国内几乎没有相关的文献. 因此, 研究 PDPTW 问题的特点以及算法具有重要的实际意义.

收稿日期: 2002-10-10; **修回日期**: 2002-12-11.
基金项目: 国家 973 重点基础研究发展基金资助项目(G19980304).
作者简介: 贾永基(1976—), 男, 山东烟台人, 博士生, 从事系统优化方法等研究; 席裕庚(1946—), 男, 上海人, 教授, 博士生导师, 从事复杂系统控制理论、智能机器人等研究.

2 PDPTW 问题的数学描述

假设车辆的数目不受限制, 且每辆车的最大载货量相同. 以下给出单车库 PDPTW 问题的数学描述[2]:

设有 n 个客户, 以 i 为索引. 称客户 i 的装货地点为点 i, 卸货地点为点 $n+i$, 车库为点 0 和点 $2n+1$. 这样就区分了客户、他们的装卸货点以及其他的点. 需要指出的是, 不同的点可能代表相同的物理位置. 因此, $N=\{0,1,\cdots,n,\cdots,2n,2n+1\}$ 是所有点的集合, $P^+=\{1,2,\cdots,n\}$ 指装货点集合, $P^-=\{n+1,n+2,\cdots,2n\}$ 指卸货点集合, $P=P^+\cup P^-$ 是除了车库点以外所有点的集合.

客户 i 的货物为 l_i, 即从点 i 运到点 $n+i$ 的货物量. 以 $[a_i,b_i]$ 代表客户 i 的装货时间窗口, $[a_{n+i},b_{n+i}]$ 代表客户 i 的卸货时间窗口; V 是车辆 v 的集合, Q 是每一辆车的容量. 以 $[a_0,b_0]$ 代表车从车库出发的时间窗口, $[a_{2n+1},b_{2n+1}]$ 代表车返回车库的时间窗口. 对于不同的两个点 i 和 j, $t_{i,j}$ 和 $c_{i,j}$ 分别代表从点 i 到点 j 的运输时间和运输代价.

数学表达式中用到 3 种类型的变量: 二进制变量 $X^v_{i,j}, v\in V, i,j\in N, i\neq j$; 时间变量 $T^v_i, i\in N$; 货物变量 $L^v_i, i\in P$. 如果车辆 v 从点 i 行驶到点 j, 则二进制变量 $X^v_{i,j}$ 为 1, 否则为 0. 时间变量 T^v_i 为车辆 v 在点 i 的服务开始时间, 货物变量 L^v_i 为车辆 v 在对点 i 服务后的总货物量.

单车库带时间窗口约束的装卸货问题, 其数学描述如下:

$$\min\sum_{v\in V}\sum_{j\in N}c_{i,j}X^v_{i,j}, \tag{1}$$

subject to

$$\sum_{v\in V}\sum_{j\in N}X^v_{i,j}=1,\quad i\in P; \tag{2}$$

$$\sum_{j\in N}X^v_{i,j}-\sum_{j\in N}X^v_{j,i}=0,\quad i\in P,\ v\in V; \tag{3}$$

$$\sum_{j\in P^+}X^v_{0,j}=1,\quad v\in V; \tag{4}$$

$$\sum_{j\in P^-}X^v_{i,2n+1}=1,\quad v\in V; \tag{5}$$

$$\sum_{j\in N}X^v_{i,j}-\sum_{j\in N}X^v_{j,n+i}=0,\quad i\in P^+,\ v\in V; \tag{6}$$

$$T^v_i+t^v_{i,n+i}\le T^v_{n+i},\quad i\in P^+,\ v\in V; \tag{7}$$

$$X^v_{i,j}=1\Rightarrow T^v_i+t^v_{i,j}\le T^v_j,\quad i,j\in P,\ v\in V; \tag{8}$$

$$X^v_{0,j}=1\Rightarrow T^v_0+t^v_{0,j}\le T^v_j,\quad j\in P^+,\ v\in V; \tag{9}$$

$$X^v_{i,2n+1}=1\Rightarrow T^v_i+t^v_{i,2n+1}\le T^v_{2n+1},\quad j\in P^-,\ v\in V; \tag{10}$$

$$a_i\le T^v_i\le b_i,\quad i\in P,\ v\in V; \tag{11}$$

$$a_0\le T^v_0\le b_0,\quad v\in V; \tag{12}$$

$$a_{2n+1}\le T^v_{2n+1}\le b_{2n+1},\quad v\in V; \tag{13}$$

$$X^v_{i,j}=1\Rightarrow L^v_i+l_j=L^v_j,\quad i\in P,\ j\in P^+,\ v\in V; \tag{14}$$

$$X^v_{i,j}=1\Rightarrow L^v_i-l_{j-n}=L^v_j,\quad i\in P,\ j\in P^-,\ v\in V; \tag{15}$$

$$X^v_{0,j}=1\Rightarrow L^v_0+l_j=L^v_j,\quad j\in P^+,\ v\in V; \tag{16}$$

$$L^v_0=0,\quad v\in V; \tag{17}$$

$$l_i\le L^v_i\le Q^v,\quad i\in P^+,\ v\in V; \tag{18}$$

$$X^v_{i,j}\in\{0,1\},\quad i,j\in N,\ v\in N. \tag{19}$$

以上诸式中: (1) 为目标函数, 优化目标是最小化总运输代价; (2) 和 (3) 为每个客户点必须且只能被服务一次; (4) 和 (5) 为每辆车必须从车库出发到装货点, 最后从卸货点回到车库; (6) 为装货点 i 及其对应的卸货点 $n+i$ 必须被同一辆车 v 访问; (7) 为节点 i 必须在节点 $n+i$ 之前被访问; (8)～(10) 为路径与调度间的一致性; (11)～(13) 为时间窗口限制; (14)～(16) 为路径和车辆容量的一致性; (17) 和 (18) 为车辆容量限制.

3 现有求解 PDPTW 问题的方法

现有文献中提出的求解 PDPTW 问题的方法, 可分为最优化方法和启发式方法两大类.

最优化方法也称精确方法, 就是找到一组路径集合, 使得其目标函数值比其他任何一组可行路径集合的目标函数值更好. 目前, 解决 PDPTW 问题的最优化方法主要有两种: 动态规划算法和列生成算法.

PDPTW 问题是 NP-hard 问题, 并且现实中的 PDPTW 问题规模很大, 可能有成百上千个客户需求, 想以可接受的运算速度找到最优解几乎是不可能的. 而启发式算法可在相对短时间内找到优良解, 所谓优良解就是接近最优解但并不一定是最优解. 启发式算法分为经典启发式算法和现代启发式算法两类.

经典启发式算法, 如分解法、插入法、局部改进法等, 能给出大规模问题的可行解, 但解的质量依赖于问题的特征.

现代启发式算法, 如模拟退火、禁忌搜索、人工智能方法等, 具有一定的灵活性, 但研究成果很少.

目前提出的解决 PDPTW 问题的现代启发式算法主要有: 反应式禁忌搜索算法[3] 和嵌入禁忌搜索的模拟退火算法[4]. 反应式禁忌搜索算法可根据目前的状态和搜索效果自动调整禁忌长度等参数, 从而容易得到较好的解. 嵌入禁忌搜索的模拟退火算法是传统的模拟退火和禁忌搜索算法的改进, 当经过一定次数的没有改进的循环后, 会从当前最好解重新开始循环过程. 但这两种启发式算法的时间复杂度都很高, 很难适应现实问题对快速求解的要求.

4　求解 PDPTW 问题的一种快速禁忌算法

本文采用改进的两阶段法来处理实际的 PDPTW 问题. 在第 1 阶段, 使用插入算法来构造一个尽可能好的初始解; 在第 2 阶段, 使用禁忌搜索算法来改进得到的解. 该算法的主要步骤如下:

算法 1　求解 PDPTW 的改进二阶段算法

Step 1: 使用插入算法构造初始可能解;

Step 2: 使用禁忌搜索算法改进解.

4.1　初始可行解的构造

本文使用插入算法来构造初始可行解. 基本算法如下:

算法 2　插入算法[5]

Step 1: 设置初始车辆路径为空.

Step 2: 如果还有未服务的货物, 则选择其中的一个; 否则, 便找到初始可行解, 终止.

Step 3: 寻找它在当前构造路径中的最佳可行位置. 如果存在这样的插入位置, 则将其插入到最佳插入位置, 并跳到 Step 2; 否则, 新建一条路径, 并将其插入, 跳到 Step 2.

在大多数情况下, 插入算法得出的初始解不会偏离最优解太远.

4.2　禁忌搜索算法[6]

禁忌搜索可以跳出局部最优解, 如果参数调整合适, 则可找到全局最优解. 禁忌搜索成功的关键是邻域结构定义. 如果一个解仅通过一次操作便可变为另一个解, 则称这两个解为邻居.

在禁忌算法中, 执行 SPI 邻域操作[7]. SPI 试图将一个客户从当前车辆路径中移动到另一个可行的车辆路径, 而使代价减少最大. 一旦确定了一个客户, SPI 便试图把它移动到所有其他路径的所有可行位置中去. 所谓可行位置是指所有的约束都必须满足的位置, 其中最重要的是时间窗口约束和车辆容量约束. 为尽量减少使用的车辆数, 应尽可能将客户从较短的车辆路径移动到较长的车辆路径.

4.2.1　代价函数定义

假设从路径 r_1 中选择一个客户, 并把它插入到路径 r_2 中, 经过 SPI 操作后的路径记为 r_1' 和 r_2', 记路径 r 的代价为 $f(r)$. 则经过一次 SPI 操作的代价为

$$\mathrm{SPIdeltaCost} = f(r_1') + f(r_2') - f(r_1) - f(r_2). \tag{20}$$

4.2.2　禁忌搜索算法

本文的快速禁忌算法如下:

算法 3　快速禁忌搜索算法

Step 1: 寻找最好的允许 SPI 操作和最好的禁忌 SPI 操作, 也就是使 SPIdeltaCost 最小的 SPI 操作.

Step 2: 如果最好的允许 SPI 操作的 SPIdeltaCost 值不大于 0, 则执行最好的允许 SPI 操作, 跳到 Step 1.

Step 3: 如果满足禁忌特赦条件, 即最好的禁忌 SPI 操作可得到更好的解, 则解禁最好的禁忌 SPI 操作并执行, 跳到 Step 1.

Step 4: 如果最好的禁忌 SPI 操作的 SPIdeltaCost 值不小于 0, 则优化每一条路径.

Step 5: 如果解得到改进, 则跳到 Step 1; 否则, 如果存在最好的允许 SPI 操作, 则执行并跳到 Step 1; 如果存在最好的禁忌 SPI 操作, 则执行并跳到 Step 1; 否则, 结束.

Step 6: Step 1～ Step 5 迭代 h 次.

5　算法复杂度分析

现在分析算法的复杂度. 得到初始解的复杂度很低, 可以忽略不计. 下面分析禁忌搜索算法的复杂度.

首先分析 Step 1～ Step 5 的时间复杂度. 假设算法 2 得到的初始解是使用 m 辆车服务 n 个客户, 则平均每辆车将服务 $[n/m]$ 个客户, 也就是服务 $[2n/m]$ 个点. 从 m 条路径中选取一条要交换的路径有 m 种选法, 从该路径中选取的客户将插入到另外的 $m-1$ 条路径中. 在选定要交换的路径中, 确定用来交换的装货点最多需要 $[2n/m]$ 次. 一旦确定了装货点, 最多需要 $[2n/m]$ 次来定位其对应的卸货点. 类似地, 确定装货点在另一条路径中的插入位置最多需要 $[2n/m]$ 次, 而确定其对应的卸货点的插入位置也最多需要 $[2n/m]$ 次. 因此, 禁忌搜索算法的时间复杂度为

$$m(m-1)\frac{2n}{m}\frac{2n}{m}\frac{2n}{m}\frac{2n}{m} = O\left(\frac{n^4}{m^2}\right). \tag{21}$$

Step 4 虽然会增加算法的时间复杂度, 但实际上只在极少情况下才执行, 其执行的次数远远小于

算法的迭代次数 h. 因此在一般情况下, 该算法的时间复杂度为 $O\left(h\frac{n^4}{m^2}\right)$. 算法的运行时间是可以接受的.

6 测试及分析

6.1 测试实例

本文构造了两个实例(Case1 和 Case2), 用于测试所提出的快速禁忌搜索算法. 实例 Case1 和 Case2 的数据特点如表 1 所示.

表 1 Case1 和 Case2 的数据特点

实 例	空间点数	客户数	地理分布	时间窗口约束	容量约束
Case1	101	51	分散	无规律	较松
Case2	11	70	密集	多数很宽 少数很窄	较紧

从表 1 可以看出, Case1 和 Case2 的特点差异很大, 都具有一定的代表性. Case1 的客户地理分布分散, 时间窗口约束强一些; Case2 的客户地理分布集中, 容量约束强一些.

6.2 测试结果

本文提出的算法对于这两种情况都给出了很好的解. 程序是用 C++ 语言编写的, 在 P4 1.6 G, 128M 内存的机器上测试. 其中总代价是使用的车辆数和每辆车行驶距离的加权和. 使用一辆车的代价为 5 000, 车辆行驶每公里的代价为 1. 测试结果如表 2 所示, 已知的最好解如表 3 所示.

表 2 快速禁忌搜索算法的测试结果

实 例	车辆数	运行时间/s	总代价
Case1	5	47	26 442
Case2	4	712	20 695

表 3 已知的最好解

实 例	车辆数	总代价
Case1	6	31 324
Case2	4	21 850

6.3 测试结果分析

比较表 2 和表 3 可知, 本文算法在相当短的时间内便得到了优于已知最好解的结果, 算法运行时间都在 12 min 以内(其中 Case1 的运行时间不到 1 min), 对于此类规模的 NP-hard 问题是可以接受的. Case2 所用时间大大多于 Case1, 这是因为 Case2 是 4 辆车服务 70 个客户, 而 Case1 是 5 辆车服务 51 个客户. 前已得到算法的时间复杂度为 $O\left(h\frac{n^4}{m^2}\right)$, 算法实际运行的时间和复杂度分析的结果是一致的. 测试结果充分表明, 本文提出的算法对于解决现实规模和复杂度的 PDPTW 问题是有效的.

7 结 语

本文提出一种解决实际规模和复杂度的 PDPTW 问题的快速禁忌搜索算法. 该算法分为构造初始解和改进解两个阶段: 构造初始解阶段使用插入算法, 为改进解阶段提供所用的初始解; 改进解阶段使用禁忌搜索算法, 并通过 SPI 邻域操作来改进所得到的解. 最后构造了两个实际规模和复杂度的测试实例, 测试结果表明, 该算法对于求解实际规模和复杂度的 PDPTW 问题是有效的. 本文算法易于改进, 可通过简单的变化来求解多约束的 PDPTW 问题, 例如多车库、多货物类型且有最大工作时间约束的复杂 PDPTW 问题.

参考文献(References):

[1] 郭伟. 通信网和运输网资源优化问题研究[D]. 上海: 上海交通大学, 2001.

[2] Yvan Dumas. The pickup and delivery problem with time windows[J]. *European J Operational Research*, 1991, 54(1): 7-22.

[3] William P Nanry, Wesley J Barnes. Solving the pickup and delivery problem with time windows using reactive tabu search[J]. *Transportation Research: Part B*, 2000, 34: 107-121.

[4] Li H, Lim A. A metaheuristic for the pickup and delivery problem with time windows[A]. *Proc of the 13th IEEE Int Conf on Tools with Artificial Intelligence*[C]. Dallas: IEEE Computer Society, 2001. 160-167.

[5] Solomon M. Algorithms for the vehicle routing and scheduling problems with time window constraints[J]. *Operations Research*, 1987, 35(2): 254-265.

[6] 邢文训, 谢金星. 现代优化计算方法[M]. 北京: 清华大学出版社, 1999.

[7] Lau H C, Liang Z. Pickup and delivery with time windows: Algorithms and test case generation[A]. *Proc of the 13th IEEE Int Conf on Tools with Artificial Intelligence*[C]. Dallas: IEEE Computer Society, 2001. 333-340.