零散货物装载的一种快速算法

马士华, 刘小群
(华中科技大学 管理学院, 湖北 武汉 430074)

摘 要: 研究了零散货物的装载问题。根据待装货物体积、重量与货车容积、载重的相对关系, 将货物分为两类。对于不同货物类型的单车装载问题, 优先考虑其主要约束目标, 而对其他目标则适时调整。对于货物足够多的多车装载问题, 关键是按照货物的类型确定不同的装车顺序。对于货车足够多的多车装载问题, 则利用上次循环得到的解, 对下一次的装车次序进行调整, 使解不断趋于优化。

关键词: 零散货物; 装载; 启发式算法
中图分类号: F253 文献标识码: A 文章编号: 1003-5192(2005)02-0064-04

A Quick Heuristic Algorithm for the Bulk Cargo Loading Problem

MA Shi-hua, LIU Xiao-qun
(*School of Management, Huazhong University of Science and Technology, Wuhan* 430074, *China*)

Abstract: The bulk cargo loading problem is researched specially in this paper. In one truck-loading problem, the bulk cargo is classified as two types according to the relative relation of the volume and weight between the goods and the truck. To different types, its main constraint is taken into account firstly, and the other constraints are adjusted in time. To multi-truck loading problem of sufficient goods, identifying the truck loading sequence is the pivotal process. To multi-truck loading problem of sufficient trucks, the truck loading sequence is modified based on the last solution and the solution is tended to optimization.

Key words: bulk cargo; loading/packing; heuristic algorithm

1 引言

装载问题广泛存在于工业生产和交通运输中, 如集成电路的布置与设计、平板上物品的摆放、集装箱内货物的堆码等。所谓零散货物装载问题就是指: n 件物品, 每件具有相应的体积和重量, 需要装入 M 个容器(如货车、火车、集装箱等, 以下统称为货车)中去, 要求充分利用容器的容积和载重, 尽可能地多装载物品。

零散货物装载问题是具有复杂约束条件的组合优化问题, 是布局问题的一种[1], 有关此类问题的研究综述可见 Dyckhoff[2] 和 Morabito[3] 等学者的论文。零散货物装载的常用算法有数学规划法、图论法、各类启发式算法和人工智能方法等。归纳起来, 可以分为两大类: 一是基于组合优化理论的各种探索法。如 Bhattachaya 的深度优先搜索算法, 在求解的过程中, 优先考虑问题的一个维度[4]。该方法对求解小规模货物装载问题相当有效, 但是由于该算法的计算量随装入货物数量的增加呈指数式增长, 不太适合大规模货物的装载。

二是基于数学规划基础上的各种解析法。该类方法主要通过在算法设计过程中, 对目标和约束条件进行转换, 从而在计算精度和计算时间上达成平衡。如 Berghammer 提出的线性逼近算法, 对装载问题采用逐渐逼近的方法进行优化, 不过该算法对逼近的收敛条件较难确定[5]。Armstrong 等针对多车货物装载提出了多项式近似算法, 收敛性较好, 但是仅适合货物相对近似的零散货物[6]。最近, 有部分研究者对装载问题进行了基于各种智能算法上的探索, 如遗传算法等。

在实际应用中, 由于零散货物在体积、重量等方面差异很大, 且货车的型号、规格也大不一样, 很难找到一种适应面广、收敛速度快的求解方法, 而

收稿日期: 2004-05-09
基金项目: 国家自然科学基金重点资助项目(70332001)

经常求助于各种计算量小、求解迅速的启发式算法。本文针对零散物轻重不一、体积差异悬殊的特点，结合装载货车的属性，通过优先考虑主要约束目标，而对其他目标则适时调整，构建了零散货物装载的一种快速算法。

2 零散货物装载的数学模型

零散货物装载问题可描述如下：某物资中心（工厂、分销中心、配送中心等）拥有用于运输货物的车辆若干，现有一批货物需要进行运输或配送，要求合理安排装载计划，使得货车的利用率最高。

在该问题中，我们假设：(1)这批货物的流向是相同的；(2)没有特殊装载要求的货物，即所有的货物都可以混装；(3)在装载过程中，仅考虑货车的容积和载重限制。

为分析方便，将装载问题中涉及的变量定义如下：

$N(i)$为待装货物的集合，i 为第 i 个货物的编号，$i=1, 2, \cdots, n$，n 为待装货物的数目；$N(v_i)$为待装货物的体积集合，v_i 为货物 i 的体积，$i\in N(i)$；$N(g_i)$为待装货物的重量集合，g_i 为货物 i 的重量，$i\in N(i)$；$M(j)$为货车的集合，j 为第 j 辆货车的编号，$j=1, 2, \cdots, m$，m 为货车的数量；$M(V_j)$为货车的容积集合，V_j 为货车 j 的容积，$j\in M(j)$；$M(G_j)$为货车的载重集合，G_j 为货车 j 的载重，$j\in M(j)$；x_{ij}为 0～1 变量，如果货物 i 装入货车 j 中，记 $x_{ij}=1$，否则记 $x_{ij}=0$；y_j 为 0～1 变量，如果货车 j 被选中用于货物装载，记 $y_j=1$，否则记 $y_j=0$；$s_j^{(k)}(i)$为第 k 次迭代时货车 j 装载货物的集合，$j\in M(j)$，$i\in N(i)$；$s_j^*(i)$为货车装载货物的可接受集合，即满意解或最优解，$j\in M(j)$，$i\in N(i)$；$S^{(k)}$为第 k 次迭代时货车集 $M(j)$ 装载货物的集合，即 $S^{(k)}=(s_1^{(k)}, s_2^{(k)}, \cdots, s_m^{(k)})$；$S^*$ 为货车集 $M(j)$装载货物的可接受集合，即满意解或最优解。

建立零散货物装载的数学模型如下

$$\min Z=\sum_{j=1}^{m} y_j \qquad \max Z_g=\sum_{y_j=1}\sum_{i=1}^{n} g_i x_{ij}$$

$$\max Z_v=\sum_{y_j=1}\sum_{i=1}^{n} v_i x_{ij}$$

$$\text{s.t.}\quad \sum_{j=1}^{m} x_{ij}\leq 1 \quad i=1, 2, \cdots, n$$

$$\sum_{i=1}^{n} g_i x_{ij}\leq G_j y_j \quad j=1, 2, \cdots, m$$

$$\sum_{i=1}^{n} v_i x_{ij}\leq V_j y_j \quad j=1, 2, \cdots, m$$

$$x_{ij}=0 \text{ or } 1 \quad i=1, 2, \cdots, n;\ j=1, 2, \cdots, m$$

$$y_j=0 \text{ or } 1 \quad j=1, 2, \cdots, m$$

问题的解就是在货物集 $N(i)$中求一组互不相交的集合 $s_j^*(i)$，充分利用货车的容积和载重，使得参与装载的车辆最少，或充分利用有限的装载车辆，尽可能多装货物，提高车辆的容积和载重利用率。

3 零散货物装载的算法设计

3.1 单车装载

按照上述数学模型，单车装载问题可描述如下：设有一货车，其最大容积是 V，最大装载量是 G，用于运送 n 种货物，货物的体积集为 $N(v_i)=\{v_1, v_2, \cdots, v_n\}$，重量集为 $N(g_i)=\{g_1, g_2, \cdots, g_n\}$。要求合理选择待装货物以充分利用车辆的容积和载重。

如何从大量的货物中选择合适的货物进行装载并不是一件简单的问题。动态规划等精确式算法在求解中能有效收敛到最优解，但是，随着货物数量的增多，计算量越来越大，增长的很快。所以，当装载的货物很多时，一般要求构造有效的启发式算法，在使计算复杂度迅速降低的同时，仍然能够获得比较满意的装载水平。目前，针对具体的零散货物装载情况，一些学者提出了相应的装载算法。如孙焰等提出的基于组合和优化理论的多项式近似算法，能较有效解决铁路零担物资的装箱问题[7]；徐天亮等提出的容重比平衡算法，解决大小轻重不一的较小货物装车问题，尤其适合在配送中心等地使用[8]。

本文在前述研究的基础上，根据货物的特点，采取优先考虑某决策目标（如 $\max Z_g$），而对其他目标（如 $\max Z_v$）适时调整的方法，求解单车货物装载问题。算法的基本思路如下：

(1)将待装货物体积$\sum_{i\in N(i)} v_i$ 和货车体积 V 的比值$\sum_{i\in N(i)} v_i/V$ 与待装货物重量$\sum_{i\in N(i)} g_i$ 和货车载重 G 的比值$\sum_{i\in N(i)} g_i/G$ 进行比较，如果

$$\left(\sum_{i\in N(i)} v_i/V\right)\geq\left(\sum_{i\in N(i)} g_i/G\right)$$

称待装货物为体积主导货物；如果

$$\left(\sum_{i\in N(i)} v_i/V\right)<\left(\sum_{i\in N(i)} g_i/G\right)$$

称待装货物为重量主导货物。

(2)对于体积主导货物,在装载过程中,货物主要是占据了货车的容积,而不是耗用了载重。在这种情况下,应该在满足容积限制下,尽可能地多装载一些相对较重的货物,以提高货车的载重利用率;而对于重量主导货物,则正好恰恰相反。

(3)注意到不是所有的货物都进行了装载,而在第(1)步的计算过程中使用的是所有待装货物的集合,这是不合适的,需要对解进行调整。

调整思路如下:设第 k 次($k\geq 2$)循环求解得到的货车装载货物集合是 $S^{(k)}$,则剩余货物集合为 $T^{(k)}=P\setminus S^{(k)}$($P$ 为按照一定规则排序后的货物集合)。考察假如删除 $S^{(k)}$ 中的元素(或元素组合),装入 $T^{(k)}$ 中货物(或货物组合),如果货物(或货物组合)可装入,且能使得货车的装载效率得以提高,则删除 $S^{(k)}$ 中货物(或货物组合),装入新的货物(或货物组合),直到 $S^{(k)}$ 中所有元素都被考察过。

(4)如果①连续两次循环求得的货车利用率非常接近;②货车利用率已经达到期望水平,计算结束。

具体的算法设计如下:

Step1　输入数据:$N(i)$, $N(v_i)=\{v_1, v_2, \cdots, v_n\}$, $N(g_i)=\{g_1, g_2, \cdots, g_n\}$, V, G。

Step2　判断货物属性:如果 $(\sum_{i\in N(i)} v_i/V)\geq(\sum_{i\in N(i)} g_i/G)$,为体积主导货物,否则为重量主导货物(以下以体积主导货物为例,重量主导货物类似)。

Step3　初始化数据,$k=1$, $V_{sum}^{(1)}=0$, $G_{sum}^{(1)}=0$, $S^{(1)}=\Phi$。

Step4　按照货物体积由小到大排序,记结果为 P。

Step5　设定考察顺序:依次考察 P 中首尾元素,即考察顺序为 $P[1]$, $P[n]$, $P[2]$, $P[n-1]$, …。

Step6　设某次考察时候,其对应的货物序号为 i,如果 $V_{sum}^{(1)}+v_i\leq V$, $G_{sum}^{(1)}+g_i\leq G$,将货物 i 放入车中,更新 $V_{sum}^{(1)}$, $G_{sum}^{(1)}$, $S^{(1)}$;否则按规则考察下一个元素,直到所有元素都考察过一次为止。

Step7　判断是否收敛,若收敛,结束计算,输出结果,否则 $k=k+1$,转入下一次计算。

Step8　设第 k 次货车装载货物的集合是 $S^{(k)}$,则剩余货物集为 $T^{(k)}=P\setminus S^{(k)}$。

设 $S^{(k)}[i]=p$ ($i=1, 2, \cdots$),依次考察集合 $T^{(k)}$ 中的首尾元素。如果有由 $T^{(k)}$ 中部分货物构成的子集 t,同时满足①子集 t 中的货物可代替 $S^{(k)}[i]$ 装入车中(即:$\sum_{i\in t} v_i\leq V+v_p-V_{sum}^{(k)}$, $\sum_{i\in t} g_i\leq G+g_p-G_{sum}^{(k)}$);②货车的利用率得到提高($\sum_{i\in t} v_i\geq v_p$, $\sum_{i\in t} g_i\geq g_p$),则删除货物 p,将子集 t 中的货物装入车中:$(S^{(k)}\setminus\{S^{(k)}[i]\})\cup t\rightarrow S^{(k+1)}$, $T^{(k)}\setminus t\rightarrow T^{(k)}$,继续考察 $S^{(k)}[i+1]$。

Step9　若满足上述条件的 t 不存在,则在考虑删除 $S^{(k)}[i]$ 同时,拟删除 $S^{(k)}[i+1]$ ($i=1, 2, \cdots$)。同样依次考察集合 $T^{(k)}$ 中的首尾元素,判断有无满足上述条件的货物子集 t 存在。如果存在,删除 $S^{(k)}[i]$ 和 $S^{(k)}[i+1]$,将子集 t 中的货物装入车中。如果 t 不存在,则在考虑删除 $S^{(k)}[i]$ 和 $S^{(k)}[i+1]$ 的同时,拟删除 $S^{(k)}[i+2]$,继续依次考察集合 $T^{(k)}$ 中的首尾元素,判断有无满足上述条件的货物子集 t 存在。

依次类推,当 $S^{(k)}$ 中所有元素都曾被尝试删除过一次时,计算结束,记结果为 $S^{(k+1)}$,转 Step7。

该算法主要的运算集中在 Step4, Step8 和 Step9。

对于 Step4,对 n 个货物按照规则进行整序,需要执行 $O(n\cdot\log n)$ 次运算。对于依次从 P 中选取合适的货物放入车中,其循环中的全部计算量是 $O(n)$。对于 Step8 和 Step9,设解 $S^{(k)}$ 中的元素数目为 m,则 $T^{(k)}=P\setminus S^{(k)}$ 中的元素数目为 $n-m$。对于每一次考察 $S^{(k)}[i]$ ($i=1, 2, \cdots, m$),需要对 $T^{(k)}$ 进行 $O(n-m)$ 次运算。因此对解进行调整的计算复杂度是 $O(m\cdot(n-m))$。由于 $m<n$, $O(m\cdot(n-m))\leq O(n^2)$,且 $O(n\cdot\log n)\leq O(n^2)$,因此该算法的复杂度不超过 $O(n^2)$。

较之孙焰等的 A_k 算法[7](复杂度为 $O(kn^{k+1})(k\geq 1)$),复杂度显著降低。

根据设计的算法,对文献[7]中的问题进行求解。由于

$$(\sum_{i\in N(i)} v_i/V=2.41)>(\sum_{i\in N(i)} g_i/G=2.16)$$

该批货物为体积主导货物。按照货物体积由小到大排序,结果为 $P=\{8, 7, 6, 5, 4, 3, 2, 1\}$。按规则依次考察 P 中的元素,解得 $S^{(1)}=\{8, 1, 7, 6\}$,则

$T^{(1)}=P\setminus S^{(1)}=\{5,4,3,2\}$。拟删除 $S^{(1)}[1]=8$ 号货物，$T^{(1)}$ 中没有合适的货物可装入；拟删除 $S^{(1)}[1]=8$ 号和 $S^{(1)}[2]=1$ 号货物，$T^{(1)}$ 中有 $T^{(1)}[1]=5$ 号和 $T^{(1)}[4]=2$ 号货物可取代装入。继续拟删除 $S^{(1)}$ 中货物，解得不到改善，此次优化结束。继续进行第三次循环计算，发现结果同第二次完全一样。根据收敛条件，认为该问题的最优解已经求出：货车可装载的最优货物集 $S^{*}=\{5,2,7,6\}$，货车装载货物的最大体积为 $V^{*}_{sum}=108$，最大重量为 $G^{*}_{sum}=247$，货车的容积利用率为 98.18%，载重利用率为 98.8%。

3.2 多车装载

对于多车装载问题，根据货物和车辆的数量，可分为两大类：一是装载货车足够的多，而待装的货物相对有限，要求使用的车辆数目最少。二是装载车辆相对有限，要求充分利用现有车辆的容积和载重，尽可能多载货物，使车辆的利用率最高。不过对于求解过程而言，这两类装载问题的目标是基本一致的：尽可能地提高每一辆参与货物装载的货车的利用率。

多车装载的快速算法思路如下：

(1)对车辆进行选择。由于车辆的选择次序对于问题的解有直接影响，因此按照合理的规则选取恰当的车辆尤为重要。在车辆选择上，选取过程为：

① 计算待装货物体积 $\sum_{i\in N(i)} v_i$ 和所有货车容积 $\sum_{j\in M(j)} V_j$ 的比值 $\sum_{i\in N(i)} v_i/\sum_{j\in M(j)} V_j$；

② 计算待装货物重量 $\sum_{i\in N(i)} g_i$ 和所有货车载重 $\sum_{j\in M(j)} G_j$ 的比值 $\sum_{i\in N(i)} g_i/\sum_{j\in M(j)} G_j$；

③ 如果 $(\sum_{i\in N(i)} v_i/\sum_{j\in M(j)} V_j)\geq(\sum_{i\in N(i)} g_i/\sum_{j\in M(j)} G_j)$，说明待装货物为体积主导货物，按照货车容积的大小，由大到小依次选择；如果

$$(\sum_{i\in N(i)} v_i/\sum_{j\in M(j)} V_j)<(\sum_{i\in N(i)} g_i/\sum_{j\in M(j)} G_j)$$

说明待装货物为重量主导货物，按照货车载重的大小，由大到小依次选择。

(2)依次安排货车装载。对于按照上述过程选出的车 j，按照单车装载的方法进行装载，但是不需要对解进行调整。

(3)解的调整。在第二种情况下，由于车子不足，解的调整主要是每辆车上装载货物的调整，其调整方法同单车货物装载解的调整。

在第一种情况下，由于不是所有的车辆都参与了装载，但是在选车的时候，将与待装货物进行比较的是所有货车的容积、载重之和，这是不合适的。调整的方法如下：

① 第 k 次循环得到的用于货物装载的车集是 $M^{(k)}(j)$，计算 $M^{(k)}(j)$ 中所有货车的容积 $\sum_{j\in M^{(k)}(j)} V_j$ 和载重 $\sum_{j\in M^{(k)}(j)} G_j$；

② 将所有待装货物体积 $\sum_{i\in N(i)} v_i$ 和 $\sum_{j\in M^{(k)}(j)} V_j$ 的比值与所有待装货物重量 $\sum_{i\in N(i)} g_i$ 和 $\sum_{j\in M^{(k)}(j)} G_j$ 的比值进行比较，如果

$$(\sum_{i\in N(i)} v_i/\sum_{j\in M^{(k)}(j)} V_j)\geq(\sum_{i\in N(i)} g_i/\sum_{j\in M^{(k)}(j)} G_j)$$

按照 $M^{(k)}(j)$ 中货车容积的大小，由大到小依次选择货车；如果

$$(\sum_{i\in N(i)} v_i/\sum_{j\in M^{(k)}(j)} V_j)<(\sum_{i\in N(i)} g_i/\sum_{j\in M^{(k)}(j)} G_j)$$

按照 $M^{(k)}(j)$ 中货车载重的大小，由大到小依次选择货车；

③ 如果第 $k+1$ 循环计算中，$M^{(k)}(j)$ 的货车不足，需要从 $M(j)\setminus M^{(k)}(j)$ 中增加货车，说明解的质量已经下降，不必再计算；如 $k+1$ 次运算后需要的货车数目小于第 k 次，说明解的质量有改善，循环得到优化。

(4)解的收敛判断。关于解的收敛条件，可以根据实际情况的不同，采取不同的方式。当货车或者货物的数目都非常多时，一般可以预先设定一个期望装载水平 δ，如果求得的货车装载水平高于 δ，计算结束。当货车的数目较少时，而货物相对较多时，一般可以通过多次循环，当①连续两次循环求得的货车利用率非常接近

$$\left|\frac{\sum_{i\in S^{(k+1)}} v_i}{\sum_{j\in M^{(k+1)}(j)} V_j}-\frac{\sum_{i\in S^{(k)}} v_i}{\sum_{j\in M^{(k)}(j)} V_j}\right|\leq\varepsilon$$

ε 为解的差异水平，或②货车利用率已经达到期望水平时，计算结束。

4 结束语

本文对考虑体积和重量约束下零散货物的二维装载问题进行了研究。根据待装货物体积、重量与货车容积、载重的相对关系，将待装货物分为体

（下转第 63 页）

表 1 参数表

参 数	合同额	直接成本	间接成本	正常工期	最短工期	参数 ω	参数 β
合同 1	1600	1000	10	26	16	3000	0.1
合同 2	1800	1200	15	28	18	2800	0.1

计算得到均衡状态, 最优激励强度分别为: $\sigma_1^*=23.24$/月, $\sigma_2^*=16.02$/月; 最优工期 $T_1^*=T_2^*=22$ 个月。项目业主的均衡效用为 $U_e^*=6712.59$, 承包方的均衡效用分别为: $U_{c1}^*=363.38$ 和 $U_{c2}^*=216.17$。

6 结论

当一个工程项目分成多个施工标段, 有多个施工承包合同时, 应综合考虑合同中激励机制的设置, 以保证项目业主的收益目标最优。本文建立了多合同的 Stackelberg 决策模型, 通过求解该模型, 进行多合同的激励优化, 并选择最优工期, 从而使项目业主和各个承包方均达到收益最大, 达到博弈均衡状态。

参 考 文 献:

[1] Erenguc S S, Tufekci S, Zappe C J. Solving time/cost trade-off problems with discounted cash flows using generalized benders decomposition[J]. Naval Research Logistics, 1993, 40: 25-50.

[2] Patterson J H, Taibot F B, Slowinski R, et al.. Computational experience with a backtracking algorithm fro solving a general class of precedence and resource-constrained scheduling problems[J]. European Journal of Operational Research, 1990, 49: 68-79.

[3] Sung C S, Lim S K. A project activity scheduling problem with net present value measure[J]. International Journal of Production Economics, 1994, 94: 177-187.

[4] Sunde L, Lichtenberg S. Net-present－value cost/time tradeoff[J]. International Journal of Project Management, 1995, 13: 45-49.

[5] 汪嘉檠, 孙永广, 吴宗鑫. 收益激励的优化与最优工期的选择[J]. 系统工程, 2000, 18(3): 5-10.

[6] 盛昭翰. 主从递阶决策论-Stackelberg 问题[M]. 北京: 科学出版社, 1998. 1-40.

[7] Vallee T, Basar T. Off-line compution of stackelberg solutions with the genetic algorithm[J]. Computational Economics, 1999, 13(3): 201-209.

(上接第 67 页)

积主导货物和重量主导货物两类。对于体积主导货物, 在装载过程中, 货物主要是占据了货车的容积, 而不是耗用了载重。因此在算法设计中, 应在满足货车容积限制下, 尽可能地多装载一些相对较重的货物, 以提高货车的载重利用率; 而对于重量主导货物, 则正好恰恰相反。该算法计算思路简单, 收敛速度快, 尤为适合零散货物的装载。

参 考 文 献:

[1] Dowsland K A, Dowsland W B. Packing problems[J]. European Journal of Operational Research, 1992, 56(1): 2-14.

[2] Dyckhoff H, Finke U. Cutting and packing in production and distribution[M]. Physica Verlag: Heidelberg, 1992.

[3] Morabito R, Morales S. A simple and effective heuristic to solve the manufacture's pallet loading problem[J]. Journal of Operational Research Society, 1998, 49(8): 819-828.

[4] Bhattacharya S, Roy R. An exact depth-first algorithm for the pallet loading problem[J]. European Journal of Operational Research, 1998, 110(3): 610-625.

[5] Berghammer R, Reuter F. A linear approximation algorithm for bin packing with absolute approximation factor [J]. Science of Computer Programming, 2003, 48(1): 67-80.

[6] Armstrong R D, Jin Zhiying. Strongly polynomial simplex algorithm for bipartite vertex packing[J]. Discrete Applied Mathematics, 1996, 64(2): 97-103.

[7] 孙焰, 李致中. 求双目标配装的多项式近似算法[J]. 长沙铁道学院学报, 1997, 15(2): 33-39.

[8] 徐天亮, 刘小群. 多品种货物配装的优化方法[J]. 华中科技大学学报, 2003, 31(9): 15-17.