

基于多元优化算法的三维装箱问题的研究

李孙寸¹ 施心陵¹ 张松海¹ 董易¹ 高莲¹

摘要 用多元优化算法 (Multi-variant optimization algorithm, MOA) 实现三维装箱问题的求解. 算法通过随机放置和局部调整从而逐步逼近最优解. 随机放置是将随机选择的几个箱子装入容器中; 局部调整是根据目标函数值对随机放置容器的箱子序列作局部调整优化; 通过递推的随机放置和局部调整优化, 目标函数值逐步逼近最优值, 从而获得一个较为理想的三维装箱方案. 算法通过对 BR1 ~ BR10 共 1 000 组三维装箱问题测试实例的测试仿真, 得到理想的装箱效果, 说明用多元优化算法实现三维装箱问题的有效性和可行性.

关键词 三维装箱问题, 多元优化算法, 随机放置, 局部调整, 逐步逼近

引用格式 李孙寸, 施心陵, 张松海, 董易, 高莲. 基于多元优化算法的三维装箱问题的研究. 自动化学报, 2018, 44(1): 106–115

DOI 10.16383/j.aas.2018.c160381

Multi-variant Optimization Algorithm for Three Dimensional Container Loading Problem

LI Sun-Cun¹ SHI Xin-Ling¹ ZHANG Song-Hai¹ DONG Yi¹ GAO Lian¹

Abstract This paper investigates that three-dimensional container loading problem is solved by the Multi-variant optimization algorithm. The Multi-variant optimization algorithm applied random placement and partial adjustment to gradually approximate the optimal solution. The random placement denotes that several boxes of randomly selected are put into the container; the partial adjustment indicates that the sequence of the boxes in the container of random placement are topically adjusted and optimized with objective function value. Then, the objective function value will gradually approximate the optimal value by recursively random replacement, partial adjustment and optimization, and we acquire a desirable three-dimensional container loading program. In order to verify the effectiveness and practicability of three-dimensional container problem with the multi-variant optimization algorithm, 1 000 groups of three-dimensional container loading problems that vary from BR1-BR10 are tested in this paper and acquire desirable results.

Key words Three-dimensional container loading problem, multi-variant optimization algorithm, random placement, partial adjustment, gradually approximation

Citation Li Sun-Cun, Shi Xin-Ling, Zhang Song-Hai, Dong Yi, Gao Lian. Multi-variant optimization algorithm for three dimensional container loading problem. *Acta Automatica Sinica*, 2018, 44(1): 106–115

装箱问题按照目标函数的不同类型分为容器装载问题、箱柜装载问题和背包装载问题^[1]. 容器装载问题是将所有待装箱子装入一个不限尺寸的容器中, 在诸多方案中找到一个使容器体积最小的装载方案, 该类型问题一般规定容器的长和宽为常数, 高为变量; 箱柜装载问题是给定一些规格统一的方型容器和一些不同类型的方型箱子, 找到一个能把所有箱子装入最少容器中的装载方案; 背包装载问题

是给定一个固定规格的容器和一些有一定价值的箱子, 然后将部分箱子装入容器中, 找到一个使得装入容器中箱子总价值最大的装载方案, 若将箱子的体积作为价值, 则背包装载问题就转换为找到一个如何使容器的填充率最大的装载方案. 其中容器装载问题和箱柜装载问题具有一定的局限性, 限制了其工业领域的应用范围. 因此本文从背包装载问题入手对三维装箱问题作初步研究.

装箱问题是传统的 NP 问题, 具有广泛的应用领域. 随着当今世界信息技术、大数据、数据流、数据关系的分析处理的不断发展, 为装箱问题的应用拓展了新的领域, 例如信息网络、交通网和云存储等有限资源的最大利用率问题, 有限资金投资的收益率最大化问题, 以及大数据相对关系匹配度最大化问题等. 如何提高容器的空间利用率, 是当代科学研究和实践中一个非常重要的课题. 找到高利用率的装

收稿日期 2016-05-09 录用日期 2017-02-03
Manuscript received May 9, 2016; accepted February 3, 2017
国家自然科学基金 (61261007, 61561049), 云南省自然科学基金 (2013FA008) 资助
Supported by National Natural Science Foundation of China (61261007, 61561049) and Natural Science Foundation of Yunnan Province (2013FA008)
本文责任编辑 王红卫
Recommended by Associate Editor WANG Hong-Wei
1. 云南大学信息学院 昆明 650500
1. School of Information Science and Engineering, Yunnan University, Kunming 650500

箱方案, 对节约天然资源, 降低企业成本, 提高企业利润, 有着十分积极的现实意义。

近年来, 国内外的学者们陆续提出了一些解决三维装箱问题的算法。Ngoi 等^[2] 采用独特的空间表征技术求解装箱问题。Bischoff 等提出了一种启发式算法, 同时给出了三维装箱问题的一些测试实例^[3-4], 其中文献 [3] 根据层和条的选择策略, 提出了按层布局的贪婪算法。Sixt 提出了基于元启发式的禁忌搜索和模拟退火算法^[5]。Gehring 等提出并行化的遗传算法, 该方法先对箱子进行一定的预处理, 再用遗传算法对箱子进行优化组合来获得装载方案^[6]。Bortfeldt 等提出了一种禁忌搜索算法, 但该方法只适用于箱子种类较少的情况^[7]。接着, Bortfeldt 等针对层的策略, 又提出一种混合遗传算法^[8]。同时期, 国内学者何大勇等提出了求解复杂集装箱装载问题的遗传算法^[9]。随后, Eley 提出了基于“块”的算法^[10], 在此基础上, Bortfeldt 等也基于“块”的概念, 给出了并行禁忌搜索算法^[11]。Moura 等基于“剩余空间的概念”, 提出了一个贪心随机自适应搜索算法 (Greedy randomized adaptive search procedures, GRASP)^[12]。张德富等针对圆形的装箱问题提出了拟人退火算法^[13], 随后, 又提出了三维装箱问题的组合启发式算法^[14]。Parrenóo 等在文献 [12] 的基础上往前发展了一步, 将 GRASP 做了改进, 取得了明显的装箱效果^[15]。Huang 等针对单一类型的装箱问题提出了一个有效的拟人型穴度算法^[16]。张德富等提出了混合模拟退火算法, 该方法首先将箱子生成复合块, 接着给出一个基于块装载的构造型启发式算法, 然后将该启发式算法和模拟退火算法相结合, 提出了一个有效求解三维装箱问题的混合模拟退火算法^[17]。Fanslau 等也基于复合块的概念, 设计了一个有效的启发式树状搜索算法^[18]。张德富等又在文献 [17-18] 的基础上设计了一个多层启发式搜索算法^[19], 该算法是采用深度和宽度同时搜索的思想, 提出的多层搜索算法, 该算法用多层搜索思想来选择一个近似最优块进行装载, 然后逐步构造直到获得一个装载序列, 较之前的算法取得了很好的装载效果, 但随着箱子规模的增大, 该算法需要较长的计算时间。刘胜等提出的求解三维装箱问题的启发式正交二叉树搜索算法^[20], 该装箱算法装箱效率较已有算法装箱效率有显著提高, 但是当箱子种类较少时, 该算法的优势不明显。当然, 还有一些学者也取得了很好的研究成果^[21-28]。

在很多实际工业应用中, 待装箱子具有规格多样性和数量不确定性等特点。因此本文结合最优化原理, 用多元优化算法 (Multi-variant optimization algorithm, MOA) 实现三维装箱问题的求解, 该算法的基本思想是随机放置、局部调整和逐步逼近。通

过对 BR1~BR10 共 1000 组实例的仿真测试, 取得明显的装箱效果, 证明了本文算法的有效性和可行性。

1 问题描述

给定一个长 L 、宽 W 、高 H 的矩形容器 C 和 n 个矩形箱子 $b_1, \dots, b_i, \dots, b_n$ 。每个箱子 b_i 的长为 l_i , 宽为 w_i , 高为 h_i , 体积为 $v_i = l_i \times w_i \times h_i$, 容器 C 的体积 $V = L \times W \times H$ 。该问题的目标为: 在箱子没有溢出容器, 且满足方向性约束和稳定性约束的条件下使容器的填充率最大, 即要尽可能多地将箱子装入容器中。问题的目标函数为

$$fitness = \max \frac{\sum v_i}{V} \quad (1)$$

在实际装箱问题中, 一般会遇到 3 个著名的约束条件^[18]:

约束 1 (C1). 方向性约束。在许多应用场合下, 箱子的装载具有方向性约束。方向性约束即为箱子的长度边、宽度边和高度边是否可以与容器的高度边平行放置。若没有这个约束条件的问题, 在装箱过程中, 箱子的任意一条边都可以竖直放置。

约束 2 (C2). 稳定性约束。在许多应用场合下, 像在物流和交通运输等领域中, 有时箱子在装载时必须满足稳定性约束。也就是说每个箱子在装载时要必须得到容器底部或是其他已经装载了的箱子的支撑。针对不同的应用情况, 稳定性约束有部分支撑约束和完全支撑约束。部分支撑约束是指被装载箱子的底部允许有部分悬空, 完全支撑约束是指被装载箱子的底部不允许有悬空的部分。

约束 3 (C3). 完全切割约束。在切割时, 每一刀必须将材料完全分割成两个部分。例如在木材和石材等的切割应用中, 一般的切割机只能做到完全切割。

本文算法在装箱过程中, 箱子的放置方式只选择其中的几种, 即并不是箱子的每一条边都可以与容器的高度边平行放置, 这就保证了本文算法满足 C1 方向性约束, 箱子的放置方式见第 2.3 节述; 同时, 剩余空间的分割与合并方式保证了本文算法满足 C2 稳定性约束, 剩余空间的分割与合并方式见第 2.2 节和第 2.4 节。综上所述, 本文算法是基于 C1 和 C2 两个约束进行的。

2 装箱准备

2.1 最优化原理

根据一类多阶段决策问题的特性, 由 Bellman 等提出了解决动态规划的“最优化原理”, 作为整个过程的最优策略具有这样的性质: 无论过去的状态

和决策如何, 对当前决策形成的状态而言, 余下的所有状态必须构成最优策略. 即若要解决某一个优化问题, 需要做出 n 个决策 $D_1, \dots, D_i, \dots, D_n$, 若这个决策序列是最优策略, 对于任何一个 $i, 1 < i < n$, 不论前面的 i 个决策是如何, i 决策以后的最优决策只取决于由前面决策所确定的当前的 i 状态, 即以后的决策 $D_{i+1}, D_{i+2}, \dots, D_n$ 也是最优的^[29].

2.2 空间的划分

箱子的装载是在容器的剩余空间中进行的, 剩余空间是指容器中未被填充的空间. 箱子理论上可以放在容器的任意位置, 但不能超出容器的容纳范围, 也不能与其他货物交迭放置. 为了满足装箱问题的稳定性约束和保证每一个箱子装进容器之前, 剩余空间都是立方体空间, 本文选择将箱子从容器的原点坐标处开始放置, 并且箱子摆放时必须与坐标轴平行或正交, 不能斜放, 也不能悬浮放置. 当第一个箱子装进容器后, 将剩余空间分割成相对于刚刚装入的箱子的前空间、右空间和上空间三个立方体空间, 如图 1 所示. 对每一个箱子装入容器后产生的剩余空间都严格做同样的分割处理. 为了装箱的有效进行, 每装进一个箱子, 更新记录一次容器的剩余空间.

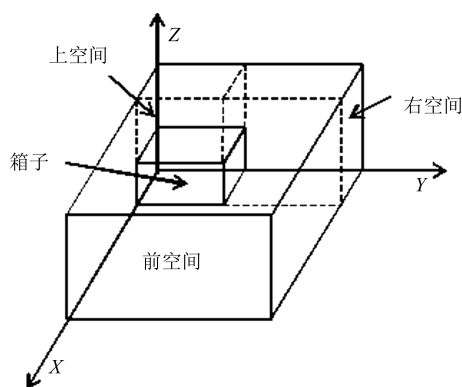


图 1 空间的分割
Fig. 1 Space division

2.3 箱子的放置

假设箱子的摆放没有方向性约束 C1, 则在装箱过程中, 箱子的任意一条边都可以竖直放置, 此时可以将箱子的摆放方式细分为 6 种: 1) l 平行于 L , w 平行于 W , h 平行于 H ; 2) w 平行于 L , l 平行于 W , h 平行于 H ; 3) l 平行于 L , h 平行于 W , w 平行于 H ; 4) h 平行于 L , l 平行于 W , w 平行于 H ; 5) h 平行于 L , w 平行于 W , l 平行于 H ; 6) w 平行于 L , h 平行于 W , l 平行于 H . 如图 2 (a)~2 (f) 所示. 如果箱子的摆放有方向性约束 C1, 则根据具体的方向性约束来选择箱子的摆放方式. 箱子放入

容器时, 本文算法先选择摆放方式后选择放置空间.

2.4 剩余子空间的合并

随着装箱的不断进行, 剩余空间中可能会产生一些废弃空间, 废弃空间指的是装不下任何一个待装箱子的剩余子空间. 要提高容器的利用率, 就要最大程度的使用废弃空间, 即要尽可能地将废弃空间变为可用空间, 唯一的办法就是将废弃空间与其他剩余子空间合并. 空间合并需同时满足 3 个条件: 1) 两个空间相邻; 2) 两个空间底面同高; 3) 两个空间合并后, 可用空间必须增大. 在同时满足上述 3 个条件时, 根据两个相邻空间的相对位置, 将空间的合并方式分为左右合并和前后合并两种方式, 如图 3 所示.

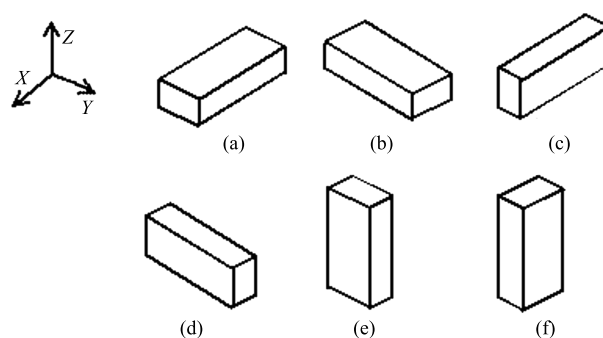


图 2 箱子的摆放方式
Fig. 2 The placement method of box

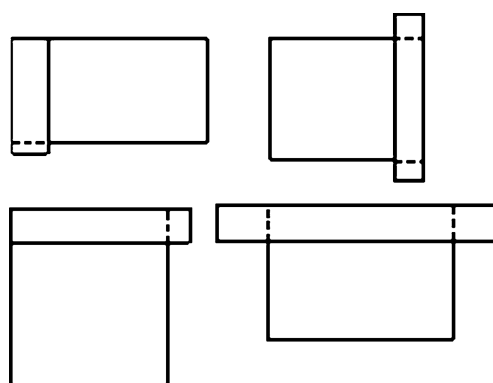


图 3 空间合并方式
Fig. 3 The merging method of residual space

3 多元优化算法

3.1 算法简述

多元优化算法 (Multi-variant optimization algorithm, MOA) 是基于全局和局部搜索交替进行的思想提出的一种全新的群智能算法, 该算法在问题的解空间中随机产生全局搜索元及相应的局部搜索元, 对解空间进行全面细致地搜索, 从而逐渐逼近全

局最优解. 算法基于计算机的数据结构, 构造多元化结构体, 结构体用于记忆并协调搜索信息、调动全局局部交替寻优、记忆寻优过程, 多次全局局部交叉搜索后, 获得全局最优解和多个次优解^[30]. 结构体是搜索元全局和局部交替寻优信息交流的平台, 是寻优过程记忆的载体. 搜索元和结构体构成了 MOA 的基本框架^[30].

结构体是一种能够按照一定规则, 记忆协调搜索信息, 有固定结构的数据表, 如图 4 所示.

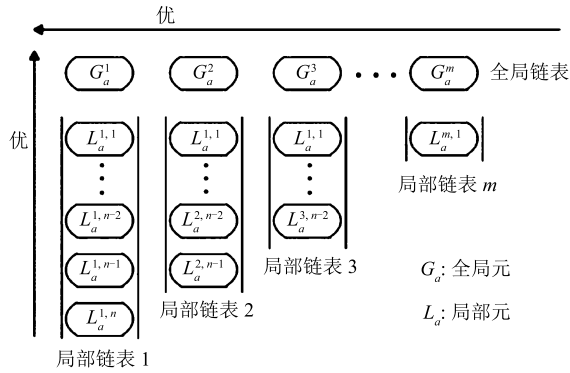


图 4 MOA 数据结构图

Fig. 4 The data structure diagram of MOA

结构体是用指针实现的二维有序链表, 由全局链表和局部链表组成. 结构体中, 第一行为全局链表, 用于保存全局搜索元并记忆和共享全局信息; 每一列叫做局部链表, 用于保存各个局部解空间内的局部搜索元和记忆局部信息.

全局链表和局部链表都是以搜索元的适应度值为关键字 (Key) 的有序链表. 全局链表中前端搜索元优于后端搜索元, 局部链表中上部搜索元优于下部搜索元. 搜索元是结构体的组织细胞, 用于存放和接收寻优信息. 搜索元具有进行全局探索和局部调整的功能.

全局搜索元 G_a (Global atom) 是全局解空间内随机生成的一个解. 一个 d 维全局元描述如下:

$$G_a = [\text{unifrnd}(l_1, u_1), \dots, \text{unifrnd}(l_d, u_d)] \quad (2)$$

式中, l_i 和 u_i 分别为第 i 维待优化参数 x_i 的下限和上限, $\text{unifrnd}(l_i, u_i)$ 为返回 l_i 和 u_i 之间的一个随机数.

局部搜索元 L_a (Local atom) 是以某个全局元为中心的局部邻域内随机生成的一个解. 以全局元 G_a 为中心的局部搜索元描述如下:

$$L_a = G_a + [\text{unifrnd}(-r, r), \dots, \text{unifrnd}(-r, r)] \quad (3)$$

式中 r 为局部邻域半径, 决定了局部搜索的范围.

将全局链表记忆较优的全局元作为具有搜索价值的局部解空间的中心, 根据其所在全局链表中的

位置共享该局部解空间在所有已经发现的具有搜索价值局部解空间中的优越程度: 位置越靠前, 其所在区域越优. 当局部搜索元发现优于全局元的解时, 局部链表记忆较优的局部元将取代全局元作为新的具有搜索价值的局部解空间的中心, 以分享局部搜索获得的信息.

为了更好地说明 MOA 的实现过程, 给出测试函数

$$F(x) = \exp \left(-2 \log(2) \times \left(\frac{x - 0.1}{0.8} \right)^2 \right) \times \sin^6(5\pi x), \quad 0 \leq x \leq 1 \quad (4)$$

在给定的区间 $0 \leq x \leq 1$ 内, 用 MOA 求 $F(x)$ 取最大值时对应的解, 寻优过程如图 5 所示, 多个全局搜索元和对应的局部搜索元对解空间进行全面细致地搜索, 逐渐逼近最优解 $x = 0.1$, 搜索结果如图 6 所示.

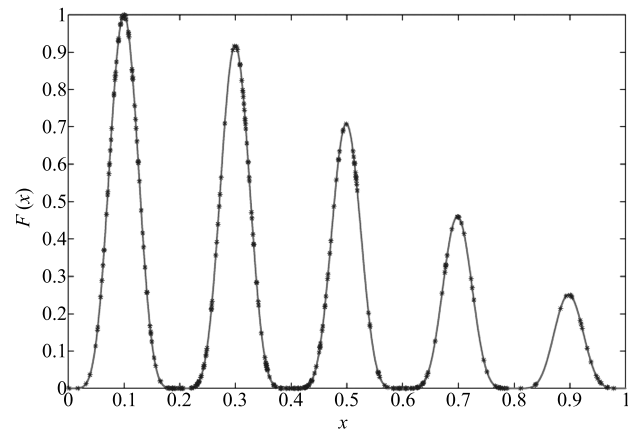


图 5 测试寻优过程图

Fig. 5 The test optimization graph

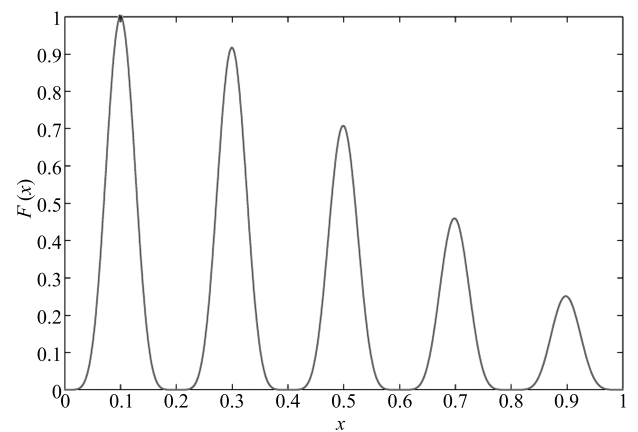


图 6 测试寻优结果图

Fig. 6 The result of test results

3.2 问题的解

针对待装箱子在实际应用中的复杂性和不确定

性, 本文采用多元优化算法实现三维装箱问题, 基本思想是:

1) 随机放置: 装入容器的箱子以及箱子的摆放方式均随机选择;

2) 局部调整: 对装入容器的局部箱子按目标函数值作局部调整;

3) 逐步逼近: 递推的局部随机放置与调整优化, 使目标函数值逐步逼近最优值.

在用 MOA 实现三维装箱问题的求解之前进行预处理:

1) 箱子编号: 按自然数序列对待装箱子进行编号, 若待装箱子总数为 d , 则所有待装箱子的编号为 $1, 2, \dots, i, \dots, d$. 若箱子数量很庞大或不确定, 则可以省略这一步, 直接分批次装即可.

2) 箱子摆放方式编号: 若没有方向性约束 $C1$, 则待装箱子的摆放方式编号为: $1, 2, 3, 4, 5, 6$, 这 6 种摆放方式分别代表第 2.3 节中描述的 6 种摆放方式; 若有方向性约束 $C1$, 则根据具体的方向约束, 待装箱子的摆放方式编号为第 2.3 节中的一种或几种.

多元优化算法实现背包装载问题的具体步骤如下:

步骤 1. 编码. 编码是应用多元优化算法首先要解决的问题. 根据不同研究对象的不同性质, 把一个问题的可行解从其解空间转换到多元优化算法能处理的搜索空间中. 假设总共有 d 个待装的箱子, 则 MOA 结构体里一个全局搜索元的编码方式为 $P = [a_1, \dots, a_i, \dots, a_d]$, a_i 是 $1 \sim d$ 之间的随机的且互不重复的整数, 表示箱子的序号, i 表示箱子 a_i 装入容器的次序. d 维的搜索元 P 由全局搜索元式 (2) 产生, 装箱问题中, 需要全局搜索, 同时对局部进行调整.

步骤 2. 分批随机放置箱子. 在实际应用中, 待装箱子的种类多样且数量庞大, 因此将待装箱子分几批次装入容器. 首先取出 P 中的前 n 个待装箱子, 即第一批次的装箱个数为 n , 然后将此批次的箱子依次装入容器中, n 的大小根据箱子的数量设置. 每一个箱子装入容器时, 采取一种拟人的方法, 先随机放置, 再做逐步调整. 具体如下:

步骤 2.1. 首先随机选取一个可用空间即剩余空间, 然后随机选取一种货物即箱子的摆放方式, 最后按该种摆放方式将箱子装入选取好的剩余空间.

步骤 2.2. 若该箱子无法以该种摆放方式装入剩余空间, 则从箱子剩下的几种摆放方式中再随机选取一种, 并判断是否可以装入剩余空间. 如果可以, 则将箱子放入所选的剩余空间并记录箱子的摆放方式; 反之, 继续选取摆放方式, 直到将箱子放入剩余空间或已遍历完该箱子的所有摆放方式为止.

步骤 2.3. 重复步骤 2.1 和步骤 2.2, 直到将箱

子放入剩余空间或遍历完所有的剩余空间为止. 每装一个箱子, 更新记录一次容器的剩余空间, 若有废弃空间且满足合并的条件, 则将废弃空间与其相邻的剩余空间合并.

步骤 2.4. 重复步骤 2.1 ~ 2.3, 直到无剩余空间可装载或这一批次的 n 个箱子全部装入为止.

步骤 3. 调整优化. 利用最优化原理调整优化容器的填充率. 已有的决策序 $D_1 = [a_1, a_2, \dots, a_n]$ 不一定是最优的决策序列, 要使该序列为最优决策序列, 利用最优化原理逐步调整该序列, 直到得到一个满意的结果, 具体步骤如下:

步骤 3.1. 在容器外未装的箱子集中随机抽取一个箱子及其对应的摆放方式.

步骤 3.2. 将抽取的箱子的体积与 a_1 的体积相比较, 若抽取的箱子体积大于 a_1 的体积, 同时抽取的箱子能够放入 a_1 放进容器之前对应的剩余空间 (指 a_1 被放入容器之前算法记录的剩余空间), 则用随机抽取的箱子替换 a_1 . 反之, 若抽取的箱子的体积小于或等于 a_1 的体积, 或者抽取的箱子不能放入 a_1 放进容器之前对应的剩余空间, 则 a_1 不用被随机抽取的箱子替换. 用同样的方法依次优化该批次装进容器的箱子, 得到新的装箱序列 $D_2 = [a_1, a_2, \dots, a_n]$, D_2 是 D_1 的优化序列.

步骤 3.3. 将下一批箱子装入容器后, 按步骤 3.1 和步骤 3.2 中的方法对新装进容器的几个箱子进行调整优化. 重复上述步骤, 直到待装箱子已装完或剩余空间全为废弃空间时, 即可获得一个潜在的装箱方案. 由于局部调整后的每一个局部序列都是最优序列, 因此整个全局序列同样也是最优序列, 符合最优化原理.

步骤 4. 逐次逼近, 渐近收敛装箱过程, 直到满意为止.

算法的伪代码如下:

Begin

设置全局元个数 m ; 每个全局元内部的局部元个数即分的批次 bn ; 每个局部元的维度即每批次装箱的个数 bn_i ($1 \leq i \leq bn$); 全局解空间的下限 $l = 1$ 和上限 $u = \text{sum}_{\text{box}}$, sum_{box} 为箱子总数; 最大循环次数 I_{max} ; 最大适应度值 (即最大填充率) $\text{fitness}_{\text{max}}$; 初始化适应度值 $\text{fitness}_{\text{best}} = 0$, $K = 0$.

While $K < I_{\text{max}}$ && $\text{fitness}_{\text{best}} < \text{fitness}_{\text{max}}$

For $i = 1 : m \% \text{生成 } m \text{ 个全局元}$

$G_a^i \leftarrow$ 根据式 (2) 生成 m 个全局搜索元

(令 G_a^i 内某一维度的元素为 $G_a^{i,j}$, 则 $G_a^{i,j}$ 满足

$1 \leq G_a^{i,j} \leq \text{sum}_{\text{box}}$ 且 $G_a^{i,j} \neq G_a^{i,j+k}$

($1 \leq k \leq \text{sum}_{\text{box}}$))

End For

For $i = 1 : m$

For $j = 1 : bn$

$L_a^{i,j} \leftarrow$ 由 G_a^i 和 bn_i 产生局部元

```

    对  $L_a^{i,j}$  做调整优化
  End For
End For
更新调整结构体信息, 适应度值越好的装箱方案
越往结构体左边移
更新记录  $fitness_{best}$ 
 $K = K + 1$ 
End While
End Begin

```

算法的时间复杂度分析. 如算法的伪代码所示, 算法中最外层的 While 大循环内有两层并列的循环, 其中第一个 For 循环内的每一个全局元是一个随机生成的数组, 只需一个执行周期, 即生成每个全局元的时间复杂度为 $O(1)$, 因此第一个 For 循环的时间复杂度为 $O(m)$; 第二个循环内包含两层嵌套的 For 循环, 其中内层的 For 循环内的局部元生成的时间复杂度为 $O(1)$, 由于对每一个局部元优化调整的个数即每批次装箱的个数 bn_i 与局部元的个数 bn 相互独立, 则内层 For 循环的时间复杂度为 $O(sum_{box} + bn)$, 因此第二个循环的时间复杂度为 $O(m(sum_{box} + bn))$; 再综合最外层 While 大循环的时间复杂度 $O(K)$, 因此算法的时间复杂度的通式为 $O(K(m + m(sum_{box} + bn)))$.

由于算法中 $fitness_{best}$ 的更新具有随机性, 在最好的情况下, 在 $K = 1$ 时, $fitness_{best} = fitness_{max}$, 此时算法的时间复杂度为 $O(m + m(sum_{box} + bn))$, 但发生这种情况的概率极小. 在一般情况下, $fitness_{best}$ 只会不断趋近于 $fitness_{max}$, 因此算法在一般情况下的时间复杂度为 $O(I_{max}(m + m(sum_{box} + bn)))$.

4 实验测试

4.1 测试准备

为了证明本文算法的有效性和可行性, 用具体的三维装箱测试数据对算法做了大量测试. 采用的测试数据来自文献 [9], 测试了其中的 BR1~BR10 共 1000 个实例, 这些数据可以从 OR-Library 网站下载. 在 BR1~BR10 中, 每种类型有 100 组测试实例, 同时每种类型中箱子的类型数分别为 3, 5, 8, 10, 12, 15, 20, 30, 40, 50, 箱子的多样性从弱到强, 从而能够更好地反映算法在不同多样性装箱问题中的应用情况. 本文算法用 Matlab 实现, 实验程序运行在 Intel Core (TM) i3-3240 CPU @ 3.40 GHz 处理器上, 运行环境为 Windows 7 专业版.

4.2 测试过程

以 BR1 中第 1 组实例为例具体描述测试过程, 将该实例取名为 BR1-1. 表 1 为 BR1-1 的待装箱子

的三维值及数量.

表 1 BR1-1 待装箱子的三维值及数量

Table 1 The specification and quantity of BR1-1

长 (cm)	宽 (cm)	高 (cm)	数量
108	76	30	40
110	43	25	33
92	81	55	39

1) 在测试之前的预处理

预处理 1. 读取表 1 数据. 通过程序将表 1 转换为具体的数据矩阵, 数据矩阵的每一行代表一种类型的箱子, 每一行的前三列分别代表箱子的长、宽、高的值, 最后一列代表该种类型箱子的数量.

预处理 2. 对所有箱子按顺序编号. 由表 1 可知, BR1-1 中共有 112 个箱子, 其中, 第 1 种类型有 40 个, 第 2 种类型有 33 个, 第 3 种类型有 39 个. 因此第 1 种类型的箱子的编号为 1~40, 第 2 种类型的箱子的编号为 41~73, 第 3 种类型的箱子的编号为 74~112.

2) 以 BR1-1 为例的测试过程

步骤 1. 编码. 随机产生 112 个 1~112 之间不重复的整数串, 如 [91, 63, 3, 36, 18, 39, ..., 52, 74, 10, 97, 2], 此整数串中的每一个整数与预处理 2 中的箱子的编号对应, 例如 91 代表 91 号箱子. 相当于把所有待装箱子打乱.

步骤 2. 分批次装载箱子. 根据待装箱子的总数设置装箱过程分几批次装载, 同时设定每一批次装多少个箱子. 例如 BR1-1 有 112 个待装箱子, 装箱过程即可设置为分 5 批次装载, 第 1 批次装 32 个, 第 2 批次装 27 个, 第 3 批次装 22 个, 第 4 批次装 17 个, 第 5 批次装 14 个, 这些具体的数据只是某一次实验设置的结果, 并不是固定不变的. 接着进行装箱过程, 先将在步骤 1 产生的箱子序列串中从 91 号开始的前 32 个箱子依次装入容器中, 每一个箱子装入容器之前更新记录一次当前的剩余空间, 若中间有装不下的箱子, 则取该箱子后面的箱子继续装, 直到已装入容器的个数达到设定的 32 个或者是剩余空间已装不下任何一个未被装的箱子为止, 此时将箱子分为已装箱子和未装箱子两部分.

步骤 3. 调整优化. 从未被装入容器的箱子中随机抽取一个如编号为 10 的箱子, 比较 10 号箱子与 91 号箱子的体积大小, 若 10 号箱子的体积小于 91 号箱子的体积, 则装入容器的箱子保持不动, 同时将 10 号箱子放回未被装入容器的箱子集中, 此时默认 91 号箱子作为第一个装入容器的最优态; 接着从剩下的未被装入容器的箱子中随机抽取一个如编号为 97 的箱子, 比较 63 号箱子与 97 号箱子的体积

大小,若 97 号箱子的体积大于 63 号箱子,则此时判断若 97 号箱子是否能被 63 号箱子装入容器之前的剩余空间装下,若能装下,则将 63 号箱子替换成 97 号箱子作为第二装入容器的最优态,63 号箱子则放到未装的箱子集中. 替换后及时更新剩余空间,以便于下一个箱子正确地装入容器中. 如上所述,依次将第 1 批的 32 个箱子优化完后,继续将第 2 批次的箱子装入容器中,对第 2 批次装入的箱子也做同样的优化处理,第 3 批、第 4 批和第 5 批亦然.

步骤 4. 逐次逼近. 经过 5 次的装载调整优化之后,容器的填充率逐渐逼近上限 100%. 表 2 是某一次实验时容器填充率逐渐变化逼近的过程. 整个装箱过程满足最不利原则. 最不利原则是指:若最不利的情况也能满足问题的要求,则其他情况必然满足问题的要求.

4.3 实验结果与分析

许多国内外的研究者对文献 [3] 中的测试数据做了一些测试,本文对文献 [18] 提出的启发式树状搜索算法、文献 [3] 提出的按层布局的贪婪算法、文

献 [6] 提出的并行遗传算法、文献 [7] 提出的禁忌搜索算法、文献 [12] 提出的贪心随机自适应搜索算法、文献 [15] 提出的改进的 GRASP、文献 [17] 提出的混合模拟退火算法、文献 [19] 提出的多层启发式搜索算法、文献 [24] 提出的 VNS 算法和文献 [25] 提出的 FDA 算法等比较典型的算法得到的装箱效果进行比较分析,这些算法的计算结果与用多元优化算法实现装箱问题的计算结果如表 3 和表 4 所示. 表 3 是前 500 组实例的测试结果,表 4 是后 500 组实例的测试结果. 表 4 中最后一列是每个算法针对装箱实例 BR1~BR10 取得的平均填充率.

从表 3 和表 4 中可以看出: 1) 从横向数据来看,针对每一种算法,表 3 的装箱效果明显优于表 4 中的装箱效果,且从相邻算例之间填充率的变化量来看,表 3 要小于表 4. 不难看出是箱子种类数的不同造成这一现象. 表 3 中箱子的种类数、相邻算例之间箱子种类数的变化量都小于表 4. 箱子种类数很多即同类箱子的数量很少时,装箱过程中立方体对齐的难度大大增加,箱子与箱子中间产生的废弃空间就随之增多,从而问题求解结果的优度呈单调递减的

表 2 某次实验得到的填充率的变化关系表
Table 2 The change table of filling rate obtained from an experiment

批次	1/(调整优化后)	2/(调整优化后)	3/(调整优化后)	4/(调整优化后)	5/(调整优化后)
容器填充率	0.2749/(0.3591)	0.5495/(0.5968)	0.7324/(0.7936)	0.8926/(0.9065)	0.9625/(0.9625)

表 3 各种算法的装箱效果比较 (1~500 组)
Table 3 Comparison of packing effects of various (groups 1~500) algorithms

测试实例	约束	填充率 (%)				
		BR1	BR2	BR3	BR4	BR5
箱子的种类数		3	5	8	10	12
H-BR ^[3]	C1 and C2	83.79	84.44	83.94	83.71	83.80
GA-GB ^[6]	C1 and C2	85.80	87.26	88.10	88.04	87.86
TS-BG ^[7]	C1 and C2	87.81	89.40	90.48	90.63	90.73
GRASP ^[12]	C1	93.52	93.77	93.58	93.05	92.34
Maximal-space ^[15]	C1	93.85	94.22	94.25	94.09	93.87
HSA ^[17]	C1 and C2	93.81	93.94	93.86	93.57	93.22
CLTRS ^[18]	C1	95.05	95.43	95.47	95.18	95.00
	C1 and C2	94.51	94.73	94.73	94.41	94.13
MLHS ^[19]	C1	94.92	95.48	95.69	95.53	95.44
	C1 and C2	94.49	94.89	95.20	94.94	94.78
VNS ^[24]	C1	94.93	95.19	94.99	94.71	94.33
FDA ^[25]	C1	92.92	93.93	93.71	93.68	93.73
MOA	C1 and C2	95.62	94.68	94.41	94.23	94.03

表 4 各种算法的装箱效果比较 (501 ~ 1 000)

Table 4 Comparison of packing effects of various (groups 501 ~ 1 000) algorithms

测试实例	约束	填充率 (%)					平均
		BR6	BR7	BR8	BR9	BR10	
箱子的种类数		15	20	30	40	50	
H_BR ^[3]	C1 and C2	82.44	82.01	80.10	78.03	76.53	81.88
GA_GB ^[6]	C1 and C2	87.85	87.68	87.52	86.46	85.53	87.21
TS_BG ^[7]	C1 and C2	92.72	90.65	87.11	85.76	84.73	89.00
GRASP ^[12]	C1	91.72	90.55	86.13	85.08	84.21	90.40
Maximal-space ^[15]	C1	93.52	92.94	91.02	90.46	89.87	92.81
HSA ^[17]	C1 and C2	92.72	91.99	90.56	89.70	89.06	92.24
CLTRS ^[18]	C1	94.79	94.24	93.70	93.44	93.09	94.54
	C1 and C2	93.85	93.20	92.26	91.48	90.86	93.42
MLHS ^[19]	C1	95.38	94.95	94.54	94.14	93.95	95.00
	C1 and C2	94.55	93.95	93.12	92.48	91.83	94.02
VNS ^[24]	C1	94.04	93.53	92.78	92.19	91.92	93.86
FDA ^[25]	C1	93.63	93.14	92.92	92.49	92.24	93.23
MOA	C1 and C2	94.56	93.27	93.09	91.52	91.00	93.64

趋势,这在一定程度上体现了三维装箱问题存在的一个难题:箱子的种类数越多,即箱子的多样性越强,问题的求解就越困难.2)从具体的平均填充率数据来看,在同时满足方向性约束 C1 和稳定性约束 C2 的情况下,填充率最高的是由张德富等提出的多层启发式搜索算法 MLHS 的 94.02%,第 2 是本文算法 MOA 的 93.64%,第 3 是 Fanslau 等提出的启发式搜索算法 CLTRS 的 93.42%,第 4 是由张德富等提出的混合模拟退火算法 HAS 的 92.24%,第 5 为 Bortfeldt 等提出的禁忌搜索算法 TS.BG 的 89.00%,第 6 为 Gehring 等提出的并行化遗传算法 GA_GB 的 87.20%,第 7 是 Bischoff 等提出的按层布局的贪婪算法 H_BR 的 81.88%.因此从数据分析结果来看,本文算法具有有效性和可行性.

表 5 给出了本文算法对 BR1 ~ BR10 测试时的具体细节.表 5 中从左起的第三列 Minimum 到最后一列 Average 分别表示本文算法在同一类型的 100 个测试实例的运行中记录下的最短运行时间、最长运行时间、平均运行时间、最小填充率、最大填充率和平均填充率,这里的运行时间指程序的实际运行时间.

从表 5 可以看出,1) 算法最短运行时间与最长运行时间相差很大,原因是本文算法具有较强的随机性;2) 随着箱子种类数的增加,算法的运行时间近似呈单调递增的趋势,这是由于箱子种类数越多,在求解问题时,箱子的组合情况越复杂,要取得好的装箱效果就需付出较大的时间代价.

用多元优化算法实现三维装箱问题时,经多批次随机放置和局部调整优化,装箱结果逐步逼近全局最优解的过程如图 7 所示.在实验过程中,装箱的批次数和每批次的装箱个数等这些参数的设置与每一个算例的箱子总数紧密相关.针对 1 000 个算例,经过测试,得出装箱的批次数的取值范围为 [5, 12] 比较适中,每批次的装箱个数呈缓慢递减的趋势.

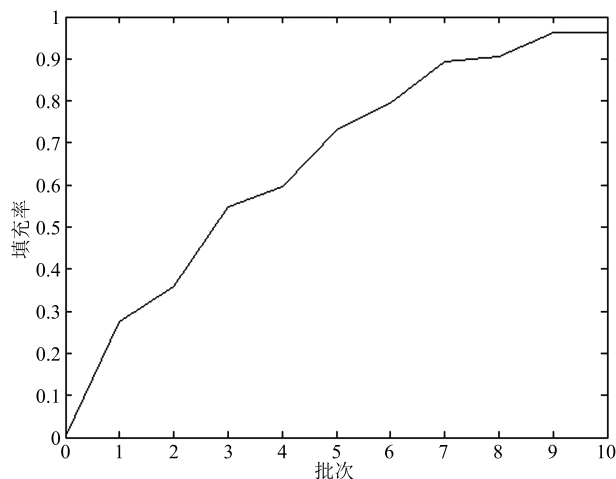


图 7 MOA 实现装箱问题过程中填充率变化图

Fig. 7 The filling rate variation diagram of packing problem with MOA

本文主要验证 MOA 算法实现三维装箱问题的有效性和可行性,进一步工作将对算法参数的设置做自适应讨论研究.

表 5 多元优化算法实现三维装箱问题时的一些测试细节
Table 5 Some test details of MOA for 3D packing problem

测试实例	约束	运行时间 (s)			填充率 (%)		
		Minimum	Maximum	Average	Minimum	Maximum	Average
BR1	3	1.83	114.66	28.84	91.04	98.31	95.62
BR2	5	2.41	57.89	26.78	89.18	97.23	94.68
BR3	8	3.42	191.23	86.35	90.60	96.97	94.41
BR4	10	1.26	274.91	105.67	88.82	96.04	94.23
BR5	12	7.50	219.01	110.63	89.17	97.45	94.03
BR6	15	5.83	495.02	265.74	87.36	96.56	94.56
BR7	20	14.61	811.50	384.38	5.82	95.44	93.27
BR8	30	30.82	1 312.80	560.21	84.61	95.17	93.09
BR9	40	33.40	1 798.75	866.08	84.69	94.07	91.52
BR10	50	54.52	2 401.94	1 380.62	82.27	93.84	91.00
Mean	19.30	15.56	767.77	381.53	87.36	96.11	93.64

5 结束语

三维装箱问题属于典型的 NP 难题, 本文采用多元优化算法求解, 具有随机放置、局部调整和逐步逼近三大特性. 经过多次实验测试, 取得理想的装箱效果, 证明了多元优化算法实现三维装箱问题的有效性和可行性. 由于本文采用的是分批次随机放置, 因此针对实际装箱问题中的复杂性和不确定性具有很好的优势. 但是算法若要得到更高的装箱效果, 就要增大试验次数, 随之计算时间也会增大. 同时, 分多少批次装载和每批次装多少个等这些参数的设置与对象紧紧相关, 不具有一般性.

References

1 Dyckhoff H, Finke U. *Cutting and Packing in Production and Distribution*. Heidelberg: Physica-Verlag, 1992.

2 Ngoi B K A, Tay M L, Chua E S. Applying spatial representation techniques to the container packing problem. *International Journal of Production Research*, 1994, **32**(1): 111–123

3 Bischoff E E, Ratcliff B S W. Issues in the development of approaches to container loading. *Omega*, 1995, **23**(4): 377–390

4 Davies A P, Bischoff E E. Weight distribution considerations in container loading. *European Journal of Operational Research*, 1999, **114**(3): 509–527

5 Sixt M. *Dreidimensionale Packprobleme. Lösungsverfahren Basierend auf den Meta-Heuristiken Simulated Annealing und Tabu-Suche*. Frankfurt am Main: Europaischer Verlag der Wissenschaften, 1996.

6 Gehring H, Bortfeldt A. A genetic algorithm for solving the container loading problem. *International Transactions in Operational Research*, 1997, **4**(5–6): 401–418

7 Bortfeldt A, Gehring H. A tabu search algorithm for weakly heterogeneous container loading problems. *OR Spectrum*, 1998, **20**: 237–250

8 Bortfeldt A, Gehring H. A hybrid genetic algorithm for the container loading problem. *European Journal of Operational Research*, 2001, **131**(1): 143–161

9 He Da-Yong, Zha Jian-Zhong, Jiang Yi-Dong. Research on solution to complex container-loading problem based on genetic algorithm. *Journal of Software*, 2001, **12**(9): 1380–1385
(何大勇, 查建中, 姜义东. 遗传算法求解复杂集装箱装载问题方法研究. 软件学报, 2001, **12**(9): 1380–1385)

10 Eley M. Solving container loading problems by block arrangement. *European Journal of Operational Research*, 2002, **141**(2): 393–409

11 Bortfeldt A, Gehring H, Mack D. A parallel tabu search algorithm for solving the container loading problem. *Parallel Computing*, 2003, **29**(5): 641–662

12 Moura A, Oliveira J F. A GRASP approach to the container-loading problem. *IEEE Intelligent Systems*, 2005, **20**(4): 50–57

13 Zhang D F, Li X. A personified annealing algorithm for circles packing problem. *Acta Automatica Sinica*, 2005, **31**(4): 590–595

14 Zhang De-Fu, Wei Li-Jun, Chen Qing-Shan, Chen Huo-Wang. A combinational heuristic algorithm for the three-dimensional packing problem. *Journal of Software*, 2007, **18**(9): 2083–2089
(张德富, 魏丽军, 陈青山, 陈火旺. 三维装箱问题的组合启发式算法. 软件学报, 2007, **18**(9): 2083–2089)

15 Parrenóo F, Alvarez-Valdes R, Oliveira J F, Tamarit J M. A maximal space-algorithm for the container loading problem. *INFORMS Journal on Computing*, 2008, **20**(3): 412–422

16 Huang W Q, He K. A caving degree approach for the single container loading problem. *European Journal of Operational Research*, 2009, **196**(1): 93–101

17 Zhang De-Fu, Peng Yu, Zhu Wen-Xing, Chen Huo-Wang. A hybrid simulated annealing algorithm for the three-dimensional packing problem. *Chinese Journal of Computers*, 2009, **32**(11): 2147–2156
(张德富, 彭煜, 朱文兴, 陈火旺. 求解三维装箱问题的混合模拟退火算法. 计算机学报, 2009, **32**(11): 2147–2156)

- 18 Fanslau T, Bortfeldt A. A tree search algorithm for solving the container loading problem. *INFORMS Journal on Computing*, 2010, **22**(2): 222–235
- 19 Zhang De-Fu, Peng Yu, Zhang Li-Li. A multi-layer heuristic search algorithm for three dimensional container loading problem. *Chinese Journal of Computers*, 2012, **35**(12): 2553–2561
(张德富, 彭煜, 张丽丽. 求解三维装箱问题的多层启发式搜索算法. 计算机学报, 2012, **35**(12): 2553–2561)
- 20 Liu Sheng, Zhu Feng-Hua, Lv Yi-Sheng, Li Yuan-Tao. A heuristic orthogonal binary tree search algorithm for three dimensional container loading problem. *Chinese Journal of Computers*, 2015, **38**(8): 1530–1543
(刘胜, 朱风华, 吕宜生, 李元涛. 求解三维装箱问题的启发式正交二叉树搜索算法. 计算机学报, 2015, **38**(8): 1530–1543)
- 21 Zhang Ya-Jian, Liu Yong, Xie Song-Jiang. An improved genetic algorithm for bin-packing problem. *Control Engineering of China*, 2016, **23**(3): 327–331
(张雅舰, 刘勇, 谢松江. 一种求解装箱问题的改进遗传算法. 控制工程, 2016, **23**(3): 327–331)
- 22 Wang Yan, Pan Wei-Ping, Zhang Jun-Hui. An algorithm for solving the problem of three-dimensional packing of single-sized cuboids. *Packaging Engineering*, 2015, **36**(11): 96–99
(王岩, 潘卫平, 张俊晖. 单一尺寸长方体三维装箱问题的一种求解算法. 包装工程, 2015, **36**(11): 96–99)
- 23 Ren Yue-Miao, Chen Xian-Fu, Liu Bin. Study on algorithm of three-dimensional bin packing problem for trapezoidal bin. *Microcomputer and Its Applications*, 2015, **34**(9): 18–21, 25
(任岳淼, 陈贤富, 刘斌. 面向梯形箱子的三维装箱问题算法研究. 微型机与应用, 2015, **34**(9): 18–21, 25)
- 24 Parreño F, Alvarez-Valdes R, Oliveira J F, Tamarit J M. Neighborhood structures for the container loading problem: a VNS implementation. *Journal of Heuristics*, 2010, **16**(1): 1–22
- 25 He K, Huang W Q. An efficient placement heuristic for three-dimensional rectangular packing. *Computers and Operations Research*, 2011, **38**(1): 227–233
- 26 You Wei, Lei Ding-You, Zhu Xiang. Biased random-key hybrid genetic algorithm for three-dimensional loading problem. *Computer Engineering and Applications*, 2014, **50**(22): 265–270
(游伟, 雷定猷, 朱向. 三维装箱问题的偏随机密钥混合遗传算法. 计算机工程与应用, 2014, **50**(22): 265–270)
- 27 Zhang Ying, Liu Er-Chao, Qi Ming-Yao. Quick algorithm for the three-dimensional bin packing problem with support surface constraints. *Journal of Transportation Systems Engineering and Information Technology*, 2014, **14**(2): 192–198
(张莹, 刘二超, 戚铭尧. 考虑支撑面约束的三维装箱问题快速求解方法. 交通运输系统工程与信息, 2014, **14**(2): 192–198)
- 28 He Kun, Huang Wen-Qi. An action space based deterministic efficient algorithm for solving the three-dimensional container loading. *Chinese Journal of Computers*, 2014, **37**(8): 1786–1793
(何琨, 黄文奇. 基于动作空间的三维装箱问题的确定性高效率求解算法. 计算机学报, 2014, **37**(8): 1786–1793)
- 29 Xia Cheng-Ren. Methodology of teaching the principal of optimality. *Journal of Anqing Teachers College (Natural Science)*, 2003, **9**(2): 93–95
(夏成仁. 关于最优化原理的教学. 安庆师范学院学报(自然科学版), 2003, **9**(2): 93–95)

- 30 Li Bao-Lei, Shi Xin-Ling, Gou Chang-Xing, Lv Dan-Ju, An Zhen-Zhou, Zhang Yu-Feng. Multivariate optimization algorithm and its convergence analysis. *Acta Automatica Sinica*, 2015, **41**(5): 949–959

(李宝磊, 施心陵, 苟常兴, 吕丹桔, 安镇宙, 张榆锋. 多元优化算法及其收敛性分析. 自动化学报, 2015, **41**(5): 949–959)



李孙寸 云南大学信息学院硕士研究生. 主要研究方向为智能算法及其应用. 本文通信作者.

E-mail: lisuncun@outlook.com

(**LI Sun-Cun** Master student at the School of Information Science and Engineering, Yunnan University. Her research interest covers intelligent algorithms and its application. Corresponding author of this paper.)



施心陵 云南大学信息学院教授. 主要研究方向为智能优化算法, 自适应信号处理与信息系统, 医学电子学.

E-mail: xlshi@ynu.edu.cn

(**SHI Xin-Ling** Professor at the School of Information Science and Engineering, Yunnan University. His research interest covers intelligent algorithms and the application of intelligent algorithms, adaptive signal processing and information systems, medical electronics.)



张松海 云南大学信息学院硕士研究生. 主要研究方向为智能优化算法, 电网可靠性分析. E-mail: hai_zs@sina.com

(**ZHANG Song-Hai** Master student at the School of Information Science and Engineering, Yunnan University. His research interest covers intelligent algorithms and power system reliability analysis.)



董易 云南大学信息学博士研究生. 主要研究方向为智能优化算法及其应用.

E-mail: yeo003@163.com

(**DONG Yi** Ph.D. candidate at the School of Information Science and Engineering, Yunnan University. His research interest covers intelligent algorithms and its application.)



高莲 博士, 云南大学信息学院讲师. 主要研究方向为生物医学信号处理.

E-mail: yanglianbao@yunneidongli.com

(**GAO Lian** Ph.D., lecturer at the School of Information Science and Engineering, Yunnan University. Her main research interest is biomedical signal processing.)