ISSN 1000-9825, CODEN RUXUEW
Journal of Software, Vol.18, No.9, September 2007, pp.2083−2089
DOI: 10.1360/jos182083

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

三维装箱问题的组合启发式算法*

张德富[1,2+], 魏丽军[1], 陈青山[1], 陈火旺[2,3]

[1](厦门大学 计算机科学系,福建 厦门 361005)

[2](东南融通博士后工作站,福建 厦门 361005)

[3](国防科学技术大学 计算机学院,湖南 长沙 410073)

A Combinational Heuristic Algorithm for the Three-Dimensional Packing Problem

ZHANG De-Fu[1,2+], WEI Li-Jun[1], CHEN Qing-Shan[1], CHEN Huo-Wang[2,3]

[1](Department of Computer Science, Xiamen University, Xiamen 361005, China)

[2](Longtop Group Post-Doctoral Research Center, Xiamen 361005, China)

[3](School of Computer, National University of Defense Technology, Changsha 410073, China)

+ Corresponding author: Phn: +86-592-5918207, Fax: +86-592-2580035, E-mail: dfzhang@xmu.edu.cn, http://www.xmu.edu.cn

Zhang DF, Wei LJ, Chen QS, Chen HW. A combinational heuristic algorithm for the three-dimensional packing problem. *Journal of Software*, 2007,18(9):2083−2089. http://www.jos.org.cn/1000-9825/18/2083.htm

Abstract: By combining the personification heuristics and simulated annealing, a combinational heuristic algorithm for the three-dimensional packing problem is presented. This personification heuristic algorithm is inspired by the strategy of building wall in the daily life. The point-finding way and the rules of horizontal and vertical reference line are developed to control the packing process. Simulated annealing algorithm is further used to improve the personification heuristics. Computational results on benchmark instances show that this algorithm can compete with excellent heuristics from the literature.

Key words: three-dimensional packing; heuristic algorithm; personification; simulated annealing algorithm

摘 要: 通过组合拟人启发式和模拟退火算法,提出了三维装箱问题的组合启发式算法.拟人启发式算法的主要思想来源于日常砌墙中的策略.利用找点法以及水平和垂直参考线规则来控制装填过程.用模拟退火算法改进拟人启发式.经过一些数据的测试,实验结果表明,该算法能够同文献中的优秀算法竞争.

关键词: 三维装箱;启发式算法;拟人;模拟退火算法

中图法分类号: TP301 **文献标识码**: A

把一些箱子装入容器中是一个在工业生产中经常遇到的数学难题,比如集装箱的装箱问题.对于二维装箱问题的求解算法已经研究得比较多,例如文献[1−3],而对于三维装箱问题的求解算法,则研究得相对较少.本文研究三维装箱问题,并且假设箱子和容器都是方型的,有关不规则形状箱子的装填算法可参考文献[4],并且我们不考虑箱子的重量和装填稳定性等约束条件,只是把箱子作为一个几何体,考虑其体积.有约束的装载可参考文

* Supported by the Academician Start-Up Fund of China under Grant No.X01109 (厦门大学院士启动基金); the 985 Information Technology Fund of China under Grant No.0000-X07204 (985 信息科技平台资助)

Received 2006-10-09; Accepted 2007-01-04

献[5].根据目标的不同,三维装箱问题可分成以下几类:

箱柜装载问题(three-dimensional bin packing problem,简称 3D-BPP):给定一些不同类型的方型箱子和一些规格统一的方型容器,问题是要把所有箱子装入最少数量的容器中.文献[6]提出了该问题的一种启发式算法.

容器装载问题(three-dimensional container-packing problems,简称 3D-CPP):在该问题中,所有箱子要装入一个不限尺寸的容器中,目标是要找一个装填,使得容器体积最小,文献[7]给出了三维矩形块布局的序列三元组编码方法.该问题更一般的形式是容器的长和宽不变,找一个装填使容器高最小,文献[8]对该问题的不同算法进行了比较.

背包装载问题(three-dimensional knapsack loading problems,简称 3D-KLP):每个箱子有一定的价值,背包装载是选择箱子的一部分装入容器中,使得装入容器中的箱子总价值最大.如果把箱子的体积作为价值,则目标转化为使容器浪费的体积最小.在本文中,我们研究 3D-KLP 的求解算法.当箱子的长、宽分别与容器的长、宽相等时,3D-KLP 问题等价于经典的背包问题.因此,3D-KLP 是 NP-难问题,对于实际应用中规模比较大的数据,很难求得它的最优解.因此,提出了很多针对该问题的启发式算法.根据文献[9]的分类,常用的启发式算法有:砌墙算法、建堆算法、立方体排列算法和切割算法.文献[10]首次对该问题提出了砌墙算法,该算法把容器分成层,层再分成条,最终转化为一维背包问题;文献[11]提出了按层布局的贪婪算法,针对层和条的选择策略,文献[9]提出了基于砌墙策略的树搜索算法;文献[12]提出了建堆启发式算法,该方法先把箱子放入合适的堆中,再通过解决一个二维装箱问题把这些堆摆放入容器中;文献[13]提出了立方体排列算法,通过一些立方体的排列(相似箱子的排列),递归地填充容器;文献[14]提出了切割算法,该方法利用分治的思想,通过切割树来表示装填,每棵切割树代表把一个容器切割成小的容器,树叶代表箱子;特别地,文献[15]通过组织装填树把大空间分成小空间进行装填,最后再对剩余空间进行处理,取得了不错的效果.这些算法都假定装填结果满足一定的结构,虽然简化了装填过程,但势必造成搜索解空间变小,从而导致解的质量不是很好.为了改进解的质量,本文基于拟人启发式和模拟退火算法,为 3D-KLP 问题提出了一种组合启发式算法.

1 问题介绍

我们给出问题 3D-KLP 的形式化定义:给定一个长方型容器 C 和一个长方型箱子的集合 $B=\{b_1,\ldots,b_n\}$,容器 C 长 H、宽 W、高 D,每个箱子 b_i 有长 h_i、宽 w_i 和高 d_i,每个箱子的体积为 $v_i=h_iw_id_i$.令 0-1 标志 ow_i,oh_i,od_i 分别表示是否允许箱子尺寸 h_i,w_i,d_i 作为垂直方向尺寸.设 S 为 B 的一个子集,定义 S 中所有箱子体积之和为 V_S,即

$$V_S=\sum_{b_i\in S}v_i .$$

问题的目标是选择 B 的一个子集 S,使得 V_S 最大,并且满足以下条件:对任何箱子 $b_i\in S$,在容器 C 中对应有一个装填位置;所有 S 中的箱子必须全部包含在 C 中;S 中任何两个箱子都不重叠;S 中箱子 b_i 的方向必须与方向标志 ow_i,oh_i,od_i 相一致.定义子集 S 对应的填充率为 V_S/WHD.

2 拟人启发式算法

拟人的思想在解决实际问题时是很有效的[16].在日常砌墙时,人们一般会先放置一块参考砖,并以参考砖的高度作为基准,规定每个物体的高度都不能超过参考砖的高度,当物体不能放入时,则提高参考砖的高度.受此思想的启发,我们在三维装箱过程中,在水平和垂直方向上同时引入参考砖来引导装填过程.我们使用了记录可放置点的方法来查找装填位置.该方法与其他方法的不同之处在于并不需要装填结果满足一定的结构,这使得装填过程非常灵活,并且通过引入水平参考线和垂直参考线来引导装填过程.

2.1 可放置点

我们把容器嵌入到一个三维坐标系中(如图 1 所示),使其底部左上角在坐标原点,长、宽、高分别在 x,y,z 轴.按顺序考虑箱子,根据经验,当前放置的箱子必须靠住容器或者前面放置的箱子.因此,我们考虑如下记录可放置点的方法.我们用每个箱子底面左上角坐标作为其放置坐标.由于箱子可旋转,假设箱 b_i 放入容器后沿 x,y,z

轴上长度分别为 h_i', w_i', d_i'.显然, h_i', w_i', d_i' 为 h_i,w_i,d_i 的排列,我们按以下 6 种顺序考虑$\langle h_i', w_i', d_i'\rangle$的大小:$\langle h_i,w_i,d_i\rangle$,$\langle w_i,h_i,d_i\rangle$,$\langle h_i,d_i,w_i\rangle$,$\langle d_i,h_i,w_i\rangle$,$\langle w_i,d_i,h_i\rangle$,$\langle d_i,w_i,h_i\rangle$.

假设给定的箱子序列为$(b_1,b_2,\ldots,b_n)$,首先,第 1 个箱子可放置点为(0,0,0),若第 1 个箱子可以放入点(0,0,0)处,则第 2 个箱子的可放置点有 3 个,分别为$(h_1',0,0)$,$(0,w_1',0)$,$(0,0,d_1')$,假设第 2 个箱子选择了点$(h_1',0,0)$,则我们删除点$(h_1',0,0)$,同时增加点$(h_1'+h_2',0,0)$,$(h_1',w_2',0)$,$(h_1',0,d_2')$,即第 3 个箱子的可放置点有 5 个.考虑第 i 个箱子,若其选择了点(x,y,z),则我们先从可放置点中删除点(x,y,z),再增加点$(x+h_i',y,z)$,$(x,y+w_i',z)$,$(x,y,z+d_i')$(如图 2 所示),若所有可放置点都不能放入第 i 个箱子,则不对可放置点更新而直接考虑第 i+1 个箱子.从放置过程中可以看出,在放入一个箱子时,删除了 1 个可放置点,同时增加了 3 个可放置点,即最终增加了 2 个点,因此,考虑第 i 个箱子时,最多有 $2\times(i-1)+1$ 个可放置点.

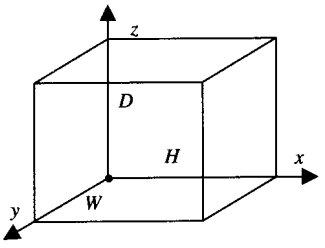

Fig.1 Coordinate system

图 1 坐标系

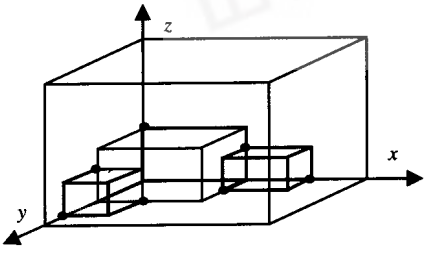

Fig.2 Available points

图 2 可放置点

2.2 参考线

我们考虑两条参考线,z 轴上的参考线 L_z 和 x 轴上的参考线 L_x.我们的放置策略是:按顺序考虑箱子,在考虑箱子 b_i 时,我们先把可放置点按 y 坐标从小到大排序,y 坐标相同的按 x 坐标从小到大排序,x,y 坐标都相同的按 z 坐标从小到大排序,我们按排好序的可放位置去检测箱子 b_i 能否放入该位置中.注意,在评判箱子 b_i 能否放入位置(x,y,z)中时,不仅要求其不能与容器和其他箱子相交,而且要求 $z+d_i'\le L_z$,$x+h_i'\le L_x$.在检测一个可放置位置时,我们根据方向标志 ow_i,oh_i,od_i 尝试所有的可放置方向,当找到第 1 个可放入点时,则把箱子 b_i 放入该位置,并更新可放置点.若所有可放置点都不能放入该箱子,则分以下两种情况考虑:(1) 若 $L_x<H$,则提高 x 轴上的参考线(reference line)(如图 3 所示),即把该箱子作为水平方向参考箱子(reference box);(2) 否则,提高 z 轴上的参考线(如图 4 所示).当然,如果提高参考线后,该箱子还是不能放下,则在此次装填中,该箱子不能放入容器中,在以后的装填中将不再考虑该箱子,也不考虑其作为参考箱子.

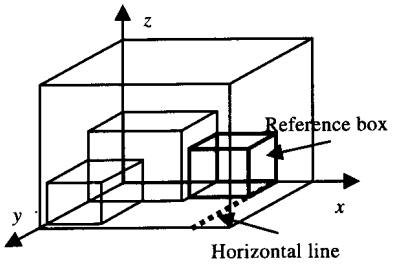

Fig.3 Horizontal line

图 3 水平线

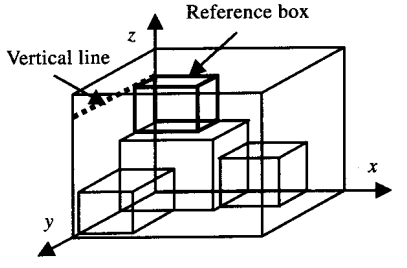

Fig.4 Vertical line

图 4 垂直线

2.3 平移箱子

根据经验,放置的箱子靠边时,才能放入更多的箱子.因此,在选择了可放置点,并且箱子放入该位置后,我们先把箱子尽量往 x 减小的方向移,再尽量往 y 减小的方向移,最后往 z 减小的方向移,所谓尽量是指容器或者其

他箱子挡住了箱子移动的方向.

2.4 装箱算法

综上,我们得到以下 3D-packing 算法,算法以有序箱子集合 $B=\{b_1,b_2,\ldots,b_n\}$为输入,返回此次装填得到的填充率,用 I 表示有序可放置点集合,在对 I 更新时应保持 I 元素的有序性,并用标志 *flag* 表示当前箱子是否能放入容器中.

算法. 3D-*packing*(B).

$I=\{(0,0,0)\}$,$L_z=L_x=0$;
for i=1 to n
  flag=false;
  for $(x,y,z)\in I$
    若 b_i 可以放入位置(x,y,z)并且满足 $x+h_i'\le L_x, z+d_i'\le L_z$,则
      flag=true,退出循环;
  若 *flag*=false,则
    若 $L_x=0$ 或 $L_x=H$,则
      若 b_i 可以放入位置$(0,0,L_z)$,则
        $x=0,y=0,z=L_z$,*flag*=true,$L_z=L_z+d_i'$,$L_x=h_i'$;
      否则,若 $L_z<D$,则
        $L_z=D,L_x=H,i=i-1$;
    否则
      for $(x,y,z)\in I$ 并且 $x=L_x,y=0$
        若 b_i 可以放入位置(x,y,z)并且满足 $z+d_i'\le L_z$,则
          flag=true,$L_x=L_x+h_i'$,退出循环;
      若 *flag*=false,则
        $L_x=H,i=i-1$;
  若 *flag*=true,则
    把 b_i 放入位置(x,y,z),$I=I/\{(x,y,z)\}$,把 b_i 先后沿 x,y,z 轴坐标减小方向平移,记平移后坐标为(x',y',z'),$I=I\cup\{(x'+h_i',y',z'),(x',y'+w_i',z'),(x',y',z'+d_i')\}$;
返回装填对应的装填率.

在该算法中,先置初始可放置点为坐标原点,并置两条参考线为 0,按顺序装填每个箱子.对于每个箱子,首先检测可放置点能否放入该箱子并且该箱子不超过参考线:若不能,则通过判断 L_x 的值来执行不同的策略,若 L_x 为 0 或为 H,表明应该增加 L_z 的值,即增加该箱子作为垂直方向参考箱子.注意,在增加 L_z 值的时候,为了充分利用空间,在箱子不能放入时,增加 L_z 到 D 并重新装填该箱子;否则,增加 L_x 的值,即增加该箱子作为水平方向参考箱子,在增加 L_x 的值时,考虑所有 $x=L_x,y=0$ 的点.同样,在箱子不能放入时,增加 L_x 到 H 并重新装填该箱子.最后,若箱子找到可放置位置(包括作为参考箱子),则把箱子放入该位置并删除该放置点,再对箱子先后进行 x,y,z 轴上往坐标减小方向的平移,平移后增加新的放置点.算法结束时,返回装填对应的填充率,即装入容器中的箱子总体积和容器体积的比率.

从算法中可以看出,算法主要运行步骤为对可放置点的检测和更新上.在对第 i 个箱子进行填充时,需检测最多$|I|$个点,同时要删除 1 个和增加 3 个点.根据第 2.1 节的结论,可放点集合 I 的大小最多为 $2\times(i-1)+1$.在检测一个可放置点时,必须判断如果第 i 个箱子放入该位置是否会跟之前放置的箱子相交.因此,检测一个点需要时间为 $O(n)$,检测$|I|$个点需要时间为 $O(n^2)$.为了保持 I 的有序性,删除和插入一个可放置点,需要时间为 $O(I)$,因此,填充一个箱子需要的时间为 $O(n^2)$.算法需填充 n 个箱子,因此算法总的时间复杂度为 $O(n^3)$.

3 组合启发式算法

模拟退火算法[17]是一种随机搜索算法,具有跳离局部最优的能力.将拟人的思想与模拟退火算法相结合,事实证明是有效的[18,19].我们考虑拟人启发式算法和模拟退火算法的结合.在模拟退火算法中,关键是邻域的定义,由于实际应用中箱子都是分类的,因此以类来考虑箱子,而不是单个的箱子.从该算法的计算过程中可以发现,箱子的装填顺序对效果影响很大,因此,我们考虑的一个邻域就是交换随机选取的两类箱子的装填顺序.在计算中我们还发现,箱子的放置方向对最终结果的影响也很大,因此,另一个邻域就是交换某类箱子的任意两个尺寸,因为这可以改变箱子的装填方向.在计算时,两种邻域被选取的概率是相等的.由于我们的目标是使装入容器中的箱子的总体积最大,因此在生成初始解时,考虑贪心策略,先把箱子按体积从大到小的顺序排序,并且为了使放入的箱子方向统一,我们先交换箱子的长、宽、高(同时交换方向标志 ow_i,oh_i,od_i),使得对任意箱子 b_i 满足

$$d_i \geq w_i \geq h_i.$$

根据模拟退火的特性,我们动态地增加邻域的大小.记初始温度为 S_t,结束温度为 E_t,温度下降率为 d_t,邻域初始大小为 L,温度下降一次时增加邻域大小为 d_L.设初始输入箱子集合为 B,用 f 记录当前的填充率,用 f_{best} 记录当前最高的填充率,B_{best} 记录最高填充率时对应的顺序,用 t 表示当前温度,L_t 表示当前邻域长度.在实验时我们发现,一次退火时,解有些波动.为了保证解的稳定性,我们进行二次退火.综上,得到组合启发式算法如下:

算法. SA-3D-*packing*(B).

把 B 按元素体积从大到小顺序排序,交换元素的尺寸,使得对任意箱子 b_i 满足 $d_i \geq w_i \geq h_i$;

$f=f_{best}$=3D-*packing*(B),$B_{best}=B$;

for i=1 to 2

  $t=S_t,L_t=L$;

  while ($t>=E_t$)

    for j=1 to L_t

      从 B 的邻域 $N(B)$中随机选取一个 B';

      f'=3D-*packing*(B'),$df=f'-f$;

      若 $df>0$,则

        $f=f',B=B'$;

        若 $f>f_{best}$,则

          $f_{best}=f,B_{best}=B$;

      否则

        随机生成一个(0,1)之间的数 x;

        若 $x<\exp(10\times df/t)$

          $f=f',B=B'$;

    $L_t+=d_L,t^*=d_t$;

4 实验结果

我们用 C++实现了该算法,并且在 Pentium 4 2.60GHZ,512MB 内存的 PC 机上运行该程序.我们选取了文献[20]中的测试数据对本文的算法进行测试,该测试数据是随机生成的[11],满足所有箱子总体积小于容器体积,但最优解并不知道,即不确定是否能把所有箱子装入容器中.该测试数据总共有 7 个文件,每个文件有 100 个例子,共 700 个例子,分别对应不同的箱子种类.

通过比较,我们最终选取模拟退火参数为 S_t=1.0,E_t=0.01,d_t=0.92,L=0,d_L=箱子种类数.图 5 为其中一个数据的装箱效果图,表 1 是与文献[11]和文献[15]的对比结果.表 2 是算法运行的详细结果,分别列出了每个文件对应

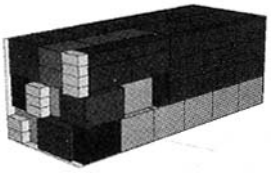

Fig.5 The packing result
图 5 装箱结果

100 个例子中的最大、最小、平均运行时间和填充率.从表 1 中可以看出,我们的算法比之前的实验结果都要好,与文献[11]相比,填充率提高了 6.378%,与文献[15]相比,填充率提高了 2.22%.从表 2 可以看出,700 个例子的平均运行时间为 85.1 秒,其中,最短的运行时间为 5.625 秒,最长的为 286.188 秒.由于箱子种类数直接影响模拟退火中需计算邻域的大小,因此,运行时间随着箱子种类数的增多而增长.从表 2 中还可以看出,当箱子种类数为 8 时,邻域空间的大小是合适的,因而平均填充率最高.这是因为箱子种类数少时,邻域空间比较小,而种类数比较多时,邻域空间又过大,从而均会影响填充率.

Table 1 Computational results comparisons of three algorithms
表 1 3 种算法的实验结果比较

Test file	Box type	Filling rate (%)		
		Bischoff	A.LIM	Combinational heuristic
Thepack1.txt	3	85.4	87.4	89.94
Thepack2.txt	5	86.25	88.7	91.13
Thepack3.txt	8	85.86	89.3	92.09
Thepack4.txt	10	85.08	89.7	91.94
Thepack5.txt	12	85.21	89.7	91.72
Thepack6.txt	15	83.874	89.7	91.45
Thepack7.txt	20	82.92	89.4	90.94
Average	10	84.942	89.1	91.32

Table 2 Computational results of combinational heuristic algorithm
表 2 组合启发式算法的实验结果

Test file	Box type	Computational time (s)			Filling rate (%)		
		Minimum	Maximum	Average	Minimum	Maximum	Average
Thepack1.txt	3	5.63	150.66	22.62	76.40	95.84	89.94
Thepack2.txt	5	12.20	108.89	31.24	84.90	95.36	91.13
Thepack3.txt	8	22.63	140.83	55.83	88.85	95.12	92.09
Thepack4.txt	10	29.39	185.91	74.55	88.63	95.18	91.94
Thepack5.txt	12	46.11	195.19	94.69	87.89	95.16	91.72
Thepack6.txt	15	58.00	252.98	126.38	88.46	94.46	91.45
Thepack7.txt	20	110.48	286.19	190.25	88.38	93.84	90.94
Total	10	5.22	286.19	85.08	76.40	95.84	91.32

5 结 论

本文提出了三维装箱问题的组合启发式算法,在装填过程记录可放置位置,并引入水平和垂直方向参考线来引导装填过程,最终通过与模拟退火的结合改变箱子装填顺序和装填方向.实验结果表明,该方法能够获得比较高的填充率.但是,当箱子种类数比较少时,由于解空间有限,改进并不明显;而当箱子种类数比较多时,算法运行需要比较长的时间.因此,如何改进装填算法和提高模拟退火搜索的效率,使得算法在箱子数及箱子种类数变化的情况下具有鲁棒性,是算法需要改进的地方.

References:

[1] Hopper E, Turton BCH. An empirical investigation of meta-heuristic and heuristic algorithms for a 2D packing problem. European Journal of Operational Research, 2001,128(1):34−57.

[2] Zhang DF, Kang Y, Deng AS. A new heuristic recursive algorithm for the strip rectangular packing problem. Computers & Operations Research, 2006,33(8):2209−2217.

[3] Cui YD, He DL, Song XX. Generating optimal two-section cutting patterns for rectangular blanks. Computers & Operations Research, 2006,33(6):1505−1520.

[4] Dai Z, Yuan JL, Cha JZ, Guo W. An octree-based heuristic algorithm for three dimensional packing. Journal of Software, 1995, 6(10):629−636 (in Chinese with English abstract).

[5] He DY, Zha JZ, Jiang YD. Research on solution to complex container-loading problem based on genetic algorithm. Journal of Software, 2001,12(9):1380−1385 (in Chinese with English abstract).

[6] Faroe O, Pisinger D, Zachariasen M. Guided local search for three-dimensional bin-packing problem. INFORMS Journal on Computing, 2003,15(3):267−283.

[7] Lu YP, Zha JZ. Sequence triplet method for 3D rectangle packing problem. Journal of Software, 2002,13(11):2183−2187 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/13/2183.pdf

[8] Bischoff EE, Marriott MD. A comparative evaluation of heuristics for container loading. European Journal of Operational Research, 1990,44(2):267−276.

[9] Psinger D. Heuristics for the container loading problem. European Journal of Operational Research, 2002,141(2):382−392.

[10] George JA, Robinson DF. A heuristic for packing boxes into a container. Computers and Operations Research, 1980,7(3):147−156.

[11] Bischoff EE, Ratcliff MS. Issues in the development of approaches to container loading. OMEGA, 1995,23(4):337−390.

[12] Gilmore PC, Gomory RE. Multistage cutting stock problems of two and more dimensions. Operations Research, 1965,13 (1):94−120.

[13] Bortfeldt A, Gehring H. Applying tabu search to container loading problems. In: Proc. of the Operations Research. Berlin: Springer-Verlag, 1997. 533−538.

[14] Morabito R, Arenales M. An AND/OR graph approach to the container loading problem. Int'l Trans. in Operational Research, 1994,1(1):59−73.

[15] Lim A, Rodrigues B, Yang Y. 3-D container packing heuristics. Applied Intelligence, 2005,22(2):125−134.

[16] Huang WQ, Xu RC. Two personification strategy for the circle packing problem. Science in China, Series E, 1999,29(4):347−353 (in Chinese with English abstract).

[17] Kang LS, Xie Y, You SY, Luo ZH. Non-Numerical Parallel Algorithms (vol.1): Simulated Annealing. Beijing: Science Press, 1998 (in Chinese).

[18] Zhang DF, HuangWQ, Wang HX. Personification annealing algorithm for solving SAT problem. Chinese Journal of Computers, 2002,25(2):148−152 (in Chinese with English abstract).

[19] Zhang DF, Li X. A personified annealing algorithm for circles packing problem. ACTA Automatica Sinica, 2005,31(4):590−595.

[20] OR-Library. http://mscmga.ms.ic.ac.uk/info.html

附中文参考文献:

[4] 戴佐,袁俊良,查建中,郭伟.一种基于八叉树结构表达的三维实体布局启发式算法.软件学报,1995,6(10):629−636.

[5] 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.

[7] 陆一平,查建中.三维矩形块布局的序列三元组编码方法.软件学报,2002,13(11):2183−2187. http://www.jos.org.cn/1000-9825/13/2183.pdf

[16] 黄文奇,许如初.支持求解圆形 packing 问题的两个拟人策略.中国科学(E 辑),1999,29(4):347−353.

[17] 康立山,谢云,尤矢勇,罗祖华.非数值并行算法(第 1 册) ——模拟退火算法.北京:科学出版社,1998.

[18] 张德富,黄文奇,汪厚祥.求解 SAT 问题的拟人退火算法.计算机学报,2002,25(2):148−152.

张德富(1971−),男,湖南宁远人,博士,副教授,主要研究领域为计算智能,金融数据挖掘.

魏丽军(1982−),男,硕士生,主要研究领域为计算智能.

陈青山(1985−),男,硕士生,主要研究领域为计算智能.

陈火旺(1936−),男,教授,博士生导师,中国工程院院士,主要研究领域为计算机软件,人工智能.