计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.200200016

求解一刀切式二维矩形 Strip Packing 问题的混合搜索算法

郭 超[1,2] 王 磊[1,2] 尹爱华[3]
1 武汉科技大学计算机科学与技术学院 武汉 430065
2 智能信息处理与实时工业系统湖北省重点实验室 武汉 430065
3 江西财经大学软件与通信工程学院 南昌 330013
(2570492776@qq.com)

摘 要 一刀切式二维矩形 Strip Packing 问题是一种 NP 难度问题。问题的实用背景是诸如玻璃板材切割、集成电路布局等工业生产中,需要优化布局和切割方案以提高利用率。总体框架是首先针对二维矩形 Packing 问题提出混合搜索算法,然后采用跳跃式查找与折半查找相结合的方式,将混合搜索算法用于求解二维矩形 Strip Packing 问题。从拟人途径提出占角、动作空间、极高度、组合拼凑等基本定义以及基本算法。以基本算法为基础,混合搜索算法分为 3 个阶段:第一阶段生成初始解。第二阶段调用邻域搜索子程序对矩形块的优先级进行调整。当邻域搜索遇到局部最优解时,采用基于随机扰动的跳坑策略子程序跳出局部最优陷阱,并在新区域继续搜索。第三阶段调用优美度枚举子程序对占角动作的选择进行优化。混合搜索算法计算了 2 组共 91 个 benchmark 实例,并将其计算结果与 SPTRS 算法进行了比较。SPTRS 算法计算结果的平均相对误差是 4.26%,混合搜索算法计算结果的平均相对误差是 3.83%。因此,混合搜索算法是一种求解一刀切式二维矩形 Strip Packing 问题的高效启发式算法。

关键词:矩形条带装箱;拟人;启发式;全局优化;一刀切
中图法分类号 TP301

Hybrid Search Algorithm for Two Dimensional Guillotine Rectangular Strip Packing Problem

GUO Chao[1,2], WANG Lei[1,2] and YIN Ai-hua[3]
1 College of Computer Science and Technology, Wuhan University of Science and Technology, Wuhan 430065, China
2 Hubei Province Key Laboratory of Intelligent Information Processing and Real-time Industrial System, Wuhan University of Science and Technology, Wuhan 430065, China
3 School of Software and Communication Engineering, Jiangxi University of Finance and Economics, Nanchang 330013, China

Abstract The guillotine rectangular strip packing problem is NP-hard. This problem has industrial applications such as glass cutting and integrated circuit layout design. These real world applications can be formulated as packing problems with the objective of maximize the usage ratio of the materials. The whole sketch is as follows. Firstly, a hybrid search algorithm is presented for solving the two dimensional guillotine rectangular packing problem (2D-GRPP), then the hybrid search algorithm is adopted for solving the two dimensional guillotine rectangular strip packing problem (2D-GRSPP) in the manner of jump search and binary search. According to the quasi-human approach, the basic definitions such as corner-occupying, action space, maximal height and rectangle combination are presented so as to induce the basic algorithm. Based on the basic algorithm, the hybrid search algorithm involves three phases. In the first phase, the initial solution is generated. In the second phase, the local search procedure runs to adjust the priority numbers of the rectangles. When the local search procedure encounters local optimal solutions, the off-trap strategy procedure is used to jump out of the trap and guide the search into the new areas. In the third phase, the beauty degree enumeration procedure is adopted to improve the selection of the corner-occupying actions. The hybrid search algorithm (called HS) is tested on two sets of 91 benchmark instances. The computational results show that the proposed algorithm generally outperforms the best heuristics (called SPTRS) in the literature up to now. The mean relative errors of HS and SPTRS are 3.83% and 4.26%, respectively. The HS algorithm is efficient for solving the problem.

Keywords Rectangular strip packing, Quasi-human, Heuristic, Global optimization, Guillotine

1 引言

一刀切式二维矩形 Strip Packing 问题是指:给定一个矩形容器和 n 个小矩形块,容器的长度 W 和 n 个小矩形块的长度和高度均为已知的正实数。在满足约束条件的前提下,要求用高度 H 尽可能小的矩形容器将所有矩形块装入。约束条件为:1)小矩形块须完全在矩形容器内;2)矩形块的每条边均与矩形容器的某条边平行或重合;3)每两个小矩形块相互

基金项目:国家自然科学基金(61862027,71701156)
This work was supported by the National Natural Science Foudation of China(61862027,71701156).
通信作者:王磊(wanglei77@wust.edu.cn)

不重叠;4)矩形块的方向固定;5)“一刀切”约束。将容器看做二维板块,对其进行切割。切割线要从被切割板块的一端延续到另一端,将容器内的矩形块切割出来。图 1 展示了“一刀切”和“非一刀切”式布局。

(a)一刀切式布局 (b)非一刀切式布局

图 1 一刀切和非一刀切

Fig. 1 Guillotine and non-guillotine cutting

当前文献中的二维矩形 Strip Packing 问题分为 4 个子类。1)RF 子类。矩形块可旋转 90°,不要求“一刀切”。2)OF 子类。矩形块方向固定,不要求“一刀切”。3)RG 子类。矩形块可旋转 90°,要求“一刀切”。4)OG 子类。矩形块方向固定,要求“一刀切”。本文研究的问题属于 OG 子类[1]。

二维矩形 Packing 问题与二维矩形 Strip Packing 问题略有不同。二维矩形 Strip Packing 问题是指:给定一个矩形容器和 n 个小矩形块,容器和 n 个小矩形块的长度和高度均为已知的正实数。在满足约束条件的前提下,将矩形块装入容器,使得容器内部的矩形块的面积之和尽可能大。

本文首先针对一刀切式二维矩形 Packing 问题提出了混合搜索算法 HS,然后采用跳跃式查找与折半查找相结合的方式,将 HS 算法用于求解一刀切式二维矩形 Strip Packing 问题。

本文结合了非随机算法和随机算法的优点,从“金角银边草肚皮”的围棋谚语得到启发,确定了优先占角、其次占边、最后选择中间区域的原则,并提出了动作空间、极高度等概念。本文还针对矩形块存在结构同构的特殊情况进行优化,提出了组合拼凑的策略,将同构矩形块作为整体考虑,以提高面积利用率。

本文第 2 节介绍了相关工作;第 3 节给出了基本定义;第 4 节介绍了算法的整体框架和各模块;第 5 节对实验结果进行了分析;最后总结全文。

2 相关工作

二维矩形 Strip Packing 是一种典型的 NP 难度问题。算法可分为完整算法和启发式算法两大类。完整算法(例如穷举型分支限界[2])枚举所有可行的切割方案,能找到最优布,局但是计算时间太长,因此只适用于问题规模较小的实例。

启发式算法可分为非随机型和随机型算法。非随机算法着力于动作的优先序,设计基本算法和树搜索算法。Jiang 等[3]提出底部左齐择优匹配算法,考虑将矩形块靠下靠左放置,按照完全匹配优先等启发式规则来选取动作。底部左齐择优匹配算法与遗传算法结合后,可进一步提高效率。Cui 等[4]提出递归型分支限界算法,采用启发式规则以缩短计算时间,提高面积利用率。Huang 等[5-6]沿拟人途径设计算法,从人类下围棋、装箱等行为得到启发,在占角动作、穴度概念基础上提出了一种拟人型基本算法,并基于基本算法,依照“下棋向前看几步”的谚语,提出向前看、然后回溯的增强算法。He 等[7]提出的动作空间的概念,可简化对合理占角动作的判定,明显改进了拟人型穴度算法。Wang 等[8]提出了占角动作的新定义,提出占角动作排序的 9 项指标,对 9 项指标进行顺序调整后可提高布局优度。Huang 等[9]和 He 等[10]将拟人算法用于求解 3 维长方体 Packing 问题。Zhang 等[11]提出一种递归算法,算法基于选择占角动作的优先规则。Bortfeldt 等[12]提出一种树搜索算法,用于求解二维矩形和三维长方体“一刀切”式 Strip Packing 问题。

随机算法以动作的优先序和基本算法来定义解空间,设计各种邻域结构,对优先序进行调整,从而发现较优布局。Alvarez[13]提出贪心随机适应搜索算法,算法分为构造阶段和改进阶段。在构造阶段,算法对选择动作的优先规则进行随机化,得到一组初始解;在改进阶段,从每一个初始解出发,在解空间中搜索以进一步提高布局优度。Leung 等[14]提出两阶段智能搜索算法,首先采用贪心局部搜索算法得到局部最优解,然后用模拟退火算法跳出局部最优陷阱。Wei 等[15]提出禁忌搜索算法以避免陷入局部最优陷阱。Yang 等[16]提出一种简单有效的随机局部搜索算法,其基本算法基于最大匹配度和最小浪费原则。当搜索遇到局部最优解时,进行若干次随机扰动以跳出局部最优陷阱。Wei 等[17]改进了对动作的评分规则,提出一种包括贪心选择、局部改进和随机改进的 3 阶段智能搜索算法。Zhang 等[18]提出变邻域搜索算法,除了常用的交换式邻域和插入式邻域,还增加了块交换和块插入式邻域。Bortfeldt[19]通过设计布局的层次结构,定义遗传操作算子,提出一种用于求解二维“一刀切”式 Strip Packing 问题的遗传算法。

3 基本定义

定义 1(格局) 如图 2 所示,已知一个长度和高度均固定的矩形容器,其中已经放入了一些矩形块,每个矩形块的位置、切割方式已知,容器外尚有若干矩形块等待放入,这称为一个格局[20]。

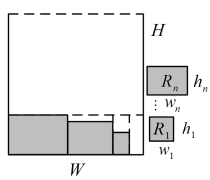


图 2 格局

Fig. 2 Configuration

定义 2(角区) 角区是矩形容器未被使用区域中的直角区域[21],可分为 90°角区和 270°区。图 3 中有 10 个角区。

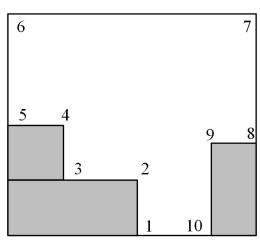


图 3 角区

Fig. 3 Corner area

定义 3(占角动作) 在满足约束条件的前提下,将矩形块放入容器,使得矩形块的一角与容器内某个 90°角区重合,

指定矩形的位置、方向和切割线，这称为占角动作。图 4 中 A，B，C，D，E 均为占角动作[21]，其中省略了切割线。

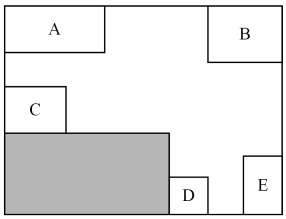


图 4　占角动作

Fig. 4　Corner-occupying action

定义 4（动作空间）　在当前布局下，在矩形容器中放入一个矩形块，并确定矩形块的切割方式。矩形容器的四条边与切割线组成的空间称为动作空间[7,21]。与文献[7]和文献[21]中动作空间的定义不同的是，一刀切式 Strip Packing 问题的动作空间之间不能相互重叠。

图 5 中存在 3 个动作空间。动作空间的定义有助于计算矩形块的合理动作。

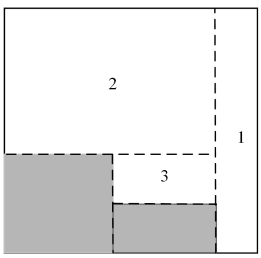
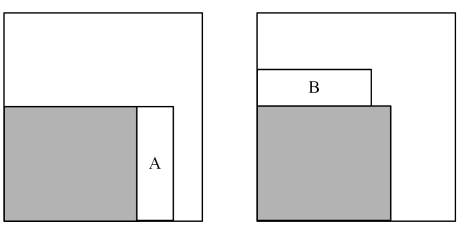


图 5　动作空间

Fig. 5　Action space

定义 5　剩余空间的平整度[21]。在矩形容器里，剩余空间的角区数越少，就越有利于后续矩形块的放置。使用平整度公式 e^{4-m} 量化角区对后续矩形块放置的影响程度，其中 m 为剩余空间的角区数，$e=2.718\cdots$。公式表明，剩余空间的角区数越少，其平整度越高。图 6(a)中剩余空间的角区数为 6，图 6(b)中剩余空间的角区数为 8。



(a)角区数为 6　　(b)角区数为 8

图 6　平整度

Fig. 6　Smooth degree

定义 6（占角动作的重叠度）　做完一个占角动作后，占角动作所对应的矩形块与动作空间的的贴边数称为重叠度[21]。

定义 7（占角动作的穴度）　如图 7 所示，将长度和高度分别为 w_i 和 h_i 的矩形块 R_i 放入动作空间 j 中，动作空间 j 的长度和高度分别为 W_j 和 H_j，此占角动作的穴度[21]用式(1)计算：

$$1-\min(d_x,d_y)\sqrt{W_jH_j} \tag{1}$$

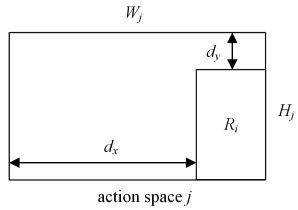


图 7　穴度

Fig. 7　Caving degree

定义 8（矩形块排序 3 项指标[21]）　1)矩形块的面积，大优先；2)矩形块的高度，大优先；3)矩形块的序号，小优先。

定义 9（占角动作的浪费度[21]）　当切割出一个矩形块之后，可能生成新的动作空间。形成浪费的原因是新的动作空间的面积过小，放不下任何一个小矩形块。新生成的被浪费的动作空间的数量称为浪费度。若图 8 中的剩余空间 A 和 B 不能被再次利用，则其浪费度为 2。

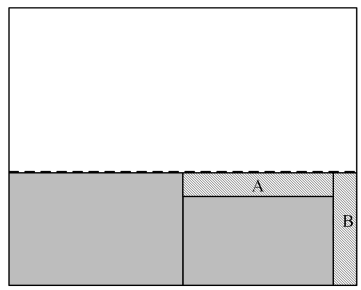


图 8　浪费度

Fig. 8　Waste degree

定义 10（极高度）　如图 9 所示，做完一个占角动作以后，产生两个动作空间，这两个动作空间的高度分别为 h_1 和 h_2，极高度定义为 h_1 和 h_2 中的较大值。极高度的定义用于选择切割方式。图 10(a)中剩余的动作空间 s_2 高度更大，能够容纳高度大的矩形块，从而减少了容器高度的开销。若极高度更大，则可能更有利于后续矩形块的放置。

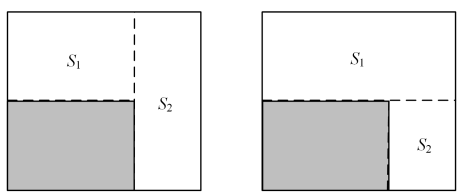
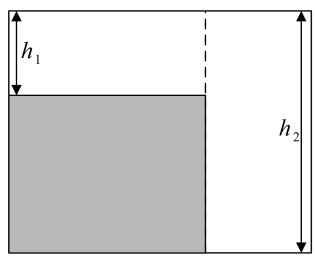


图 9　极高度

Fig. 9　Maximal height



(a)先竖切后横切　(b)先横切后竖切

图 10　切割方式

Fig. 10　Cutting pattern

定义 11（矩形块组合）　给定矩形块 R_i 和矩形块 R_j，将其组合在一起，使得 R_i 的左上顶点与 R_j 的左下顶点重合，或

者 R_i 的右下顶点与 R_j 的左下顶点重合。

定义 12(整体矩形块) 矩形块 R_i 和矩形块 R_j 组合以后,其外接矩形被称为整体矩形块,它是一个虚拟的矩形块,内部包含若干个矩形块,每个矩形块的位置、切割方式已知。

图 11 中,将矩形块 R_1 和矩形块 R_2 组合。虚线框是整体矩形块。本文将整体矩形块的优先级设定为与其包含的最高优先级的矩形块相同。

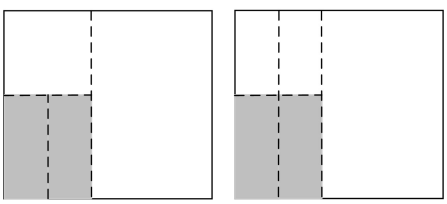
(a)不可行 (b)可行

图 11 整体矩形块

Fig. 11 Overall rectangular block

定义 13(不可行的矩形块组合) 给定矩形块 R_i 和矩形块 R_j,将其组合。有 3 种情况是不可行的:1)组合后的整体矩形块的高度大于其长度;2)组合后的整体矩形块的长度大于矩形容器的长度;3)整体矩形块的面积利用率(整体矩形块内部包含的矩形块面积之和除以整体矩形块的面积)小于阈值(本文为 80%)。

定义 14(可行的矩形块组合) 给定矩形块 R_i 和矩形块 R_j,将其组合。除去不可行的组合之外,均为可行的组合。图 12(a)和图 12(b)分别是不可行与可行的矩形块组合。



(a)不可行 (b)可行

图 12 组合切割

Fig. 12 Cutting pattern of rectangular block

定义 15(矩形块同构) 若矩形块 R_i 和矩形块 R_j 存在一种可行的组合,则称其为同构。

4 混合搜索算法

4.1 矩形块组合子程序

矩形块组合子程序用伪代码描述如下:

```
矩形块的初始队列 S={R1,…,Rn}。
round=1。n 是矩形块数。
while (1) {
  for (i=1;i<n;i++) {
    if (round==1) j=i+1;
    else j=last_round_n+1;
    for (;j<=n;j++){
      若矩形块 Ri 和矩形块 Rj 可以组合,则将其所有可行的组合形
      成的整体矩形块加入队列 S 末尾。
      }
    }
    last_round_n=n;
    round ++;
    刷新矩形块数 n。
    if (n 值未变化) break;
}
```

组合是将同构的矩形块作为整体进行切割。一方面,优先级小的矩形块与优先级大的矩形块组合后,使得优先级小的矩形块也能得到优先切割的机会,避免了矩形块排序指标造成的切割优先顺序的局限性。另一方面,组合切割策略能减少动作空间的碎片数。如图 12 所示,图 12(a)使用组合切割方式产生 2 个动作空间,图 12(b)使用单独切割方式产生 3 个动作空间。二者相比,图 12(a)中的切割方式更有利于后续的切割。

4.2 基本算法 A_0

定义 16(占角动作的 9 项指标[21-22]) (1)动作做完后,框内剩余空间的平整度,大优先;(2)动作的重叠度,大优先;(3)矩形块的优先级,大优先;(4)动作的穴度,大优先;(5)动作的浪费度,小优先;(6)动作的极高度,大优先;(7)矩形块的序号,小优先;(8)矩形块左下顶点的 y 坐标,小优先;(9)矩形块左下顶点的 x 坐标,小优先。

基本算法命名为 A_0,其步骤如下:

Step1 初始格局。矩形容器为空,所有矩形块均在矩形容器外。

Step2 在当前格局下,列举出每个矩形块的所有占角动作,依字典序按 9 项指标(见定义 16)选择最优先的占角动作来做,并确定切割方式。切割完成以后,演化到新格局。

Step3 更新占角动作所在的动作空间。如果动作空间不可再次利用,将其标记为浪费。

循环 Step2 和 Step3 直至终止格局,此时所有矩形块均已放入容器,或者虽仍有矩形块在容器外,但无法放入容器。

定义 17(整体矩形块拆卸) 在基本算法计算过程中,一旦发现某些整体矩形块所含矩形块已被放入容器中,则将这些整体矩形块拆卸,从矩形块队列中删去,以减少计算时间。

例 1 $(R_1,R_2,R_3,R_4,R_5,(R_1,R_2),(R_4,R_5))$是给定的矩形块队列,其中$(R_1,R_2)$由同构矩形块 R_1,R_2 组成,(R_4,R_5)由 R_4,R_5 组成。根据定义 16 中的指标选取最优占角动作后(假定对应矩形块是 R_5),则更新矩形块队列为$(R_1,R_2,R_3,R_4,(R_1,R_2))$。此过程称为整体矩形块拆卸。

4.3 混合搜索算法整体框架

混合搜索算法基于基本算法,分为 3 个优化阶段。本文提出的混合搜索算法的框架与文献[21]中提出的拟人型全局优化算法思路基本相同,针对"一刀切"约束对其各个子模块做了修改。算法针对矩形块优先级的设定和占角动作的选取进行优化。优化目的在于将造成的浪费减至最少。

在第一优化阶段,按照定义 16 中排序指标为所有矩形块排序,然后指定优先级,生成初始点 P_0。给定 P_0 后,从初始格局出发,运用基本算法 A_0 可唯一确定一个终止格局。这个终止格局记为 $L(A_0,P_0)$。

第二优化阶段中,调用邻域搜索子程序,对所有矩形块的优先级进行优化。若搜索到局部最优点,则调用跳坑策略子程序跳出局部最优,从而得到新的起点。对新起点再次调用邻域搜索子程序,如此反复,直至时间上限。

第二优化阶段结束后,可以得到一个新的矩形块优先级 P^+。对于 P^+,从初始布局开始,调用基本算法 A_0 可以得到

一个终止布局，此终止布局记为 $L(A_0,P^+)$。

在第三优化阶段中，调用优美度枚举子程序（见 4.7 节）。优美度枚举子程序 B_0 是一种树搜索程序，针对占角动作的选择进行优化。从初始格局出发，按照优先级 P^+，优美度枚举子程序可唯一确定一个终止格局。这个布局记为 $L(B_0,P^+)$。

4.4　初始点的生成

任意给定所有矩形块的优先级，调用基本算法 A_0 可生成一个布局。本文将所有矩形块的优先级看做 n 维空间中的点。

生成初始点的含义是为所有的矩形块指定优先级。根据大矩形块优先原则，为矩形块指定优先级的具体做法有以下 3 种：第一种，按照字典序用定义 8 中的 3 项指标依次对矩形块排序，这是“面积大优先”；第二种，将定义 8 中的第 1 项指标改为“周长大优先”；第三种，去掉定义 8 中的第 1 项指标，这是“高度大优先”。本文随机采取其中一种生成初始点。

4.5　邻域搜索子程序

$P=(p(R_1),\cdots,p(R_n))$ 指定了所有矩形块的优先级。P 表示 n 维空间中的一个点。只要指定 P 值，从初始格局出发，运用基本算法可唯一确定一个终止格局。这个终止格局记为 $L(A_0,P)$。

以 A_0 算法为基础进行适当的邻域搜索[18]（属于局部搜索），期望寻找更合理的矩形块的优先级。

定义 18（交换式邻域）　任意给定所有矩形块的优先级 P，矩形块的优先级最低为 1，最高为 k。将矩形块按照优先级从高到低的顺序排列。考虑优先级为 k_1 和 k_2 的矩形块，将它们的优先级互换，保持其余矩形块的优先级不变。从而得到一个新优先级 P^+，P^+ 的集合构成 P 的交换式邻域。其中 $1\leqslant k_1<k_2\leqslant k$。

定义 19（插入式邻域）　考虑优先级为 k_1 的矩形块，插入式邻域有两种方式：向前插入和向后插入。向前插入时，将优先级为 k_1 的矩形块提升为 k_2，$k_2>k_1$，原优先级小于等于 k_2 并且大于 k_1 的所有矩形块的优先级减 1。其余矩形块的优先级不变。向后插入时，将优先级为 k_1 的矩形块降为 k_3，$k_3<k_1$，原优先级大于等于 k_3 并且小于 k_1 的所有矩形块的优先级增 1。其余矩形块的优先级不变。

例 2　已知 8 个矩形块的优先级从高到低依次为：$P(R_7)=6$，$P(R_1)=P(R_8)=5$，$P(R_6)=4$，$P(R_3)=P(R_2)=3$，$P(R_4)=2$，$P(R_5)=1$。

使用交换式邻域。例如选取优先级为 5 和 3 的矩形块，对其进行优先级交换，结果为：$P(R_7)=6$，$P(R_3)=P(R_2)=5$，$P(R_6)=4$，$P(R_1)=P(R_8)=3$，$P(R_4)=2$，$P(R_5)=1$。

使用插入式邻域。例如考虑优先级为 4 的矩形块 R_6。

向前插入时将其优先级提高为 6 级，结果为：$P(R_6)=6$，$P(R_7)=5$，$P(R_1)=P(R_8)=4$，$P(R_3)=P(R_2)=3$，$P(R_4)=2$，$P(R_5)=1$。

向后插入时将其优先级降为 2 级，结果为：$P(R_7)=6$，$P(R_1)=P(R_8)=5$，$P(R_3)=P(R_2)=4$，$P(R_4)=3$，$P(R_6)=2$，$P(R_5)=1$。

为了节约计算时间，邻域搜索采用“见好就收”的拟人策略。具体做法是：按照基本算法依次计算当前点的全部邻点，若某个邻点对应的布局产生的浪费少于当前点，则用此点代替当前点。完成一次迭代，继续循环。

在计算的过程中，如果当前点对应布局产生的浪费比其所有邻点所对应的布局都要少，则当前点为局部最优点，停止本次邻域搜索子程序的运行。

4.6　跳坑策略子程序

当邻域搜索子程序遇到局部最优点时，混合搜索算法从新的起点继续进行邻域搜索，这称为跳坑策略子程序。与模拟退火和禁忌搜索不同的是，本文使用拟人途径跳出局部最优点。

起跳点的选择有两种方式，本文均匀地随机选择一种。第一种方式是在当前点起跳；第二种方式使用之前已搜索到的历史最优点。在起跳点进行随机扰动，具体做法为：在当前起跳点的所有邻点中均匀地随机选取 1 个点，这是迭代的 1 步。跳坑策略子程序共迭代 r 步。r 是[8,16]内均匀分布的随机整数。

4.7　优美度枚举子程序

定义 20（占角动作的优美度）　在当前格局，占角动作做完后，按照基本算法 A_0 计算到终止格局，终止格局中矩形块面积之和称为此占角动作的优美度。

优美度枚举子程序的具体步骤如下：

Step1（初始格局）　所有的矩形块均在矩形容器外部。

Step2（在当前格局）　按照定义 16 对所有占角动作进行排序，选取其中前 N 名动作作为候选，然后调用基本算法依次计算这 N 个动作的优美度。最后选取其中优美度最大的动作。如果存在多个动作优美度并列最大，则按照定义 16 对其排序，选取其中排序第一的动作。

Step3　更新动作空间。

Step4　循环执行 Step2 和 Step3，直到终止格局，停机。

本文中 N 取值为 32。优美度枚举子程序命名为 B_0。

4.8　拟人型全局优化算法族

定义 16 中的占角动作的指标顺序借鉴了人类在材料切割过程中积累的经验，具有一定的参考意义。但“尺有所短，寸有所长”，对定义 16 中指标排序进行调整，可能得到更优的布局。本文提出一个基本算法族如下。

算法 A_0 按照定义 16 的默认指标排序。

算法 A_1 由 A_0 修改而来，将定义 16 中的第(1)项指标和第(2)项指标互换。

算法 A_2 由 A_0 修改而来，将定义 16 中的第(4)项指标和第(5)项指标互换。

算法 A_3 由 A_0 修改而来，将定义 16 中的第(5)项指标和第(6)项指标互换。

算法 A_4 由 A_0 修改而来，将定义 16 中的第(4)项指标和第(6)项指标互换。

算法 A_5 由 A_0 修改而来，将定义 16 中的第(8)项指标和第(9)项指标互换。

算法 A_6 由 A_1 修改而来，将定义 16 中的第(4)项指标和第(5)项指标互换。

算法 A_7 由 A_2 修改而来，将定义 16 中的指标极高度放在浪费度前面。

算法 A_8 由 A_2 修改而来，将定义 16 中的第(8)项指标和第(9)项指标互换。

优美度枚举子程序 B_0 调用基本算法 A_0，进行树搜索。将优美度枚举子程序所调用的基本算法替换为 A_i，可以得到一个优美度枚举子程序族 B_i，其中 $0 \leqslant i \leqslant 8$。

对于 4.3 节中的混合搜索算法来说，其基于基本算法 A_0 和优美度枚举子程序 B_0。将 A_0 和 B_0 分别替换为 A_i 和 B_i，可得到一个混合搜索算法族。

5 实验结果

第 4 节中的算法用于计算矩形容器长度、高度均固定的情形。当矩形容器的高度不固定时，本文采用跳跃式查找和折半查找相结合的方式，给出一个矩形容器的高度 H 值。其流程用伪代码描述如下：

Step1 首先计算矩形容器高度的下界 LB。$LB=\lceil sum/W \rceil$，其中 sum 表示所有矩形块的面积和，$\lceil\ \rceil$ 表示向上取整，W 表示容器长度。在 0～8 中均匀地随机选取整数 i。转 Step2。

Step2 调用 A_iB_i 算法进行跳跃式查找，步长 $d=\max(1, LB/2)$。

```
for(l=LB;;l+=d){
  将矩形容器的高度设定为 l;
  调用 AiBi 算法计算;
  if (矩形块已全部放入矩形容器){
    H=l;
    break;
  }
}
if (H==LB) 转 Step4;
else 转 Step3;
```

Step3 调用 A_iB_i 算法进行折半查找：

```
head=H-d+1;tail=H-1;
while(head<=tail){
  mid=(head+tail)/2;
  将矩形容器的高度设定为 mid;
  调用 AiBi 算法计算;
  if (矩形块已全部放入矩形容器){
    H=mid;
    tail=mid-1;
  }
  else
    head=mid+1;
}
```

Step4 输出 H 值以及对应的布局。

本文的算法命名为 HS，用于计算 OG 子类。算法计算了 2 组由 Hopper 和 Turton[23] 提出的 benchmark 问题实例：1)C 组(21 个实例)，分为 C1 至 C7 共 7 个小组，每个小组 3 个实例；2)NT 组(70 个实例)，分为 N1 至 N7、T1 至 T7 共 14 个小组，每个小组 5 个实例。

混合搜索算法 HS 用 C 语言实现。为节约测试时间，设定了 HS 计算的上限 time_ub。对于小规模实例(矩形块数小于等于 100)、中规模实例(矩形块数大于 100 且小于等于 500)、大规模实例(矩形块数大于 500)，HS 算法的计算时间上限 time_ub 分别为 360 s，600 s，3600 s。

对于一个问题实例，HS 算法需要考虑不同的容器高度(采用跳跃式查找和折半查找)来计算。具体来说，对于给定的容器高度，HS 算法的计算时间上限是 time_ub 的 1/6。对于混合搜索算法 HS 的 3 个优化阶段来说，第一优化阶段(生成初始点)的时间可忽略不计，第二优化阶段(邻域搜索＋跳坑策略)与第三优化阶段(优美度枚举)的计算时间上限是 time_ub 的 1/12。

容器高度的下界 $LB=\lceil sum/W \rceil$，其中 sum 表示所有矩形块的面积和，$\lceil\ \rceil$ 表示向上取整，W 表示容器长度。每个算法所生成布局所对应的容器高度记为 H，H 的相对误差 $Gap=100\%\times(H-LB)/LB$。AGap 表示平均相对误差。

将 HS 算法与 Bortfeldt 提出的先进算法 SPGAL[19] 和 SPTRS[12] 的计算结果进行了比较。运行环境简要说明如下：SPGAL 算法和 HS 算法在主频为 2.0 GHz 的个人电脑上运行；SPTRS 算法在主频为 3.16 GHz 的个人电脑上运行。

SPGAL 算法和 HS 算法是随机型的算法。对每个问题实例计算 10 次，表 1 和表 2 中所列为每个算法生成布局所对应的容器高度的相对误差的算术平均值。SPTRS 算法是非随机型算法，对每个问题实例只计算 1 次。ATime 表示平均计算时间，单位为秒(s)。计算的最好结果在表中用粗体表示。

表 1 对 C 组问题实例的计算结果(平均相对误差)

Table 1 Results (mean gaps in %) for instance set C

	SPGAL	SPTRS	HS
C1	3.2	3.3	**3.0**
C2	3.3	2.2	**1.8**
C3	3.9	**1.1**	**1.1**
C4	3.8	1.7	**1.6**
C5	2.4	**1.5**	**1.5**
C6	1.9	1.4	**1.3**
C7	1.7	1.4	**1.2**
ATime	143	58	394
AGap	2.9	1.8	**1.7**

表 2 对 NT 组问题实例的计算结果(平均相对误差)

Table 2 Results (mean gaps in %) for instance set NT

	SPTRS	HS		SPTRS	HS
N1	10.0	**8.3**	T1	14.9	**12.6**
N2	6.2	**5.8**	T2	6.2	**5.6**
N3	**5.6**	**5.6**	T3	5.3	**5.2**
N4	3.8	**2.9**	T4	3.8	**3.5**
N5	3.4	**3.2**	T5	3.2	**2.9**
N6	2.3	**2.0**	T6	2.2	**2.0**
N7	**1.3**	1.5	T7	1.6	**1.5**
ATime	66	394	ATime	73	394
AGap	4.7	**4.18**	AGaP	5.3	**4.76**

计算结果表明，对 C 组实例的计算，HS 算法所生成布局的质量高于 SPGAL 算法和 SPTRS 算法。对 NT 组实例的计算，HS 算法所生成布局的质量高于 SPTRS 算法。文献[12]未报道 SPGAL 算法对 NT 组实例的计算结果。

对两组共 91 个 benchmark 问题实例，SPTRS 算法计算结果的平均相对误差是 4.26%，本文提出的 HS 算法计算结果的平均相对误差是 3.83%。HS 算法生成布局的质量相对较高。

结束语 以占角式基本算法为基础，本文提出了一种三

阶段拟人型混合搜索算法，由邻域搜索、跳坑策略和优美度枚举组成。对 benchmark 问题实例的测试结果表明，算法生成布局的优度较高，但算法的速度有待于进一步提高。下一步的研究方向是将算法推广到三维立方体 Packing 以及其他相关问题。

参 考 文 献

[1] LODI A, MARTELLO S, VIGO D. Heuristic and metaheuristic approaches for a class of two-dimensional bin packing problems [J]. INFORMS Journal on Computing, 1999, 11(4): 345-357.

[2] MARTELLO S, MONACI M, VIGO D. An exact approach to the strip packing problem [J]. INFORMS Journal on Computing, 2003, 15(3): 310-319.

[3] JIANG X B, L X Q, LIU C C. Lowest-level left align best-fit algorithm for the 2D rectangular strip packing problem [J]. Journal of Software, 2009, 20(6): 1528-1538.

[4] CUI Y D, YANG Y L, CHENG X, et al. A recursive branch-and-bound algorithm for the rectangular guillotine strip packing problem [J]. Computers & Operations Research, 2008, 35(4): 1281-1291.

[5] HUANG W Q, CHEN D B, XU R C. A new heuristic algorithm for rectangle packing [J]. Computers & Operations Research, 2007, 34(11): 3270-3280.

[6] HUANG W Q, CHEN D B. An efficient heuristic algorithm for rectangle-packing problem [J]. Simulation Modelling and Practice Theory, 2007, 15(10): 1356-1365.

[7] HE K, HUANG W Q, JIN Y. Efficient algorithm based on action space for solving the 2D rectangular packing problem [J]. Journal of Software, 2012, 23(5): 1037-1044.

[8] WANG L, YIN A H. A beauty degree enumeration algorithm for the 2D rectangular packing problem [J]. Scientia Sinica Informationis, 2015, 45(9): 1127-1140.

[9] HUANG W Q, HE K. A pure quasi-human algorithm for solving the cuboid packing problem [J]. Science China Series F: Information Sciences, 2009, 52(1): 52-58.

[10] HE K, HUANG W Q. An efficient place heuristic for three-dimensional rectangular packing [J]. Computers & Operations Research, 2011, 38(1): 227-233.

[11] ZHANG D F, SHI L Y, LEUNG S C H, et al. A priority heuristic for the guillotine rectangular packing problem [J]. Information Processing Letters, 2016, 116(1): 15-21.

[12] BORTFELDT A, JUNGMANN S. A tree search algorithm for solving the multi-dimensional strip packing problem with guillotine cutting constraint [J]. Annals of Operations Research, 2012, 196(1): 53-71.

[13] ALVAREZ-VALDES R, PARRENO F, TAMMRIT J M. Reactive GRASP for the strip-packing problem [J]. Computers & Operations Research, 2008, 35(4): 1065-1083.

[14] LEUNG S C H, ZHANG D F. A two-stage intelligent search algorithm for the two-dimensional strip packing problem [J]. European Journal of Operational Research, 2011, 215(1): 57-69.

[15] WEI L J, OON W C, ZHU W B, et al. A skyline heuristic for the 2D rectangular packing and strip packing problems [J]. European Journal of Operational Research, 2011, 215(2): 337-346.

[16] YANG S Y, HAN S H, YE W G. A simple randomized algorithm for two-dimensional strip packing [J]. Computers & Operations Research, 2013, 40(1): 1-8.

[17] WEI L J, QIN H, CHEANG B, et al. An efficient intelligent search algorithm for the two-dimensional rectangular strip packing problem [J]. International Transactions in Operational Research, 2016, 232(1/2): 65-92.

[18] ZHANG D F, CHE Y X, YE F R, et al. A hybrid algorithm based on variable neighborhood for the strip packing problem [J]. Journal of Combinatorial Optimization, 2016, 32(2): 513-530.

[19] BORTFELDT A. A genetic algorithm for the two-dimensional strip packing problem with rectangular pieces [J]. European Journal of Operational Research, 2006, 172(3): 814-837.

[20] HE K, HUANG W Q, JIN Y. An efficient deterministic heuristic for two-dimensional rectangular packing [J]. Computers & Operations Research, 2012, 39(7): 1355-1363.

[21] DENG J K, WANG L, YIN A H. A quasi-human global optimization algorithm for solving the two dimensional rectangular packing problem [J]. Computer Engineering & Science, 2018, 40(2): 331-340.

[22] WANG L, YIN A H. A quasi-human algorithm for the two dimensional rectangular strip packing problem: in memory of Prof. Wenqi Huang [J]. Journal of Combinatorial Optimization, 2016, 32(2): 416-444.

[23] HOPPER E, TURTON B C H. An empirical investigation of meta-heuristic and heuristic algorithm for a 2D packing problem [J]. European Journal of Operational Research, 2001, 128(1): 34-57.

GUO Chao, born in 1994, postgraduate. His main research interests include heuristic algorithm for packing problems.

WANG Lei, born in 1977, Ph. D, lecturer. His main research interests include heuristic algorithm for rectangular packing problems and job shop scheduling problem.