求解 2-D Strip Packing 问题的 u-分组优化算法

黄 海[1] 李松斌[2,3]

(莆田学院信息工程学院 莆田 351100)[1] (中国科学院声学研究所南海研究站 海口 570105)[2]

(中国科学院声学研究所国家网络新媒体工程技术研究中心 北京 100190)[3]

摘 要 2-D strip packing 问题指将带有价值的矩形物品装入长宽固定的箱子中,使其装入的物品价值最大。基于装箱的期望目标 ε,提出一种新的分组构造函数,结合装箱矩形特点计算出最优分组参数 u 并对矩形进行分类,同时对不同类别的矩形引入相应的数据结构,最后对不同类别矩形基于箱子 X 轴的 u 等分点进行填充,使其装入的物品价值最大。文中的主要贡献在于:提出了一种有效的分组构造函数;计算出了对应的最优分组参数 u;简化了不同类别箱体的数据结构以及相应的装箱算法;特别地,在期望目标 ε、多项式时间复杂度和至少装入(1−ε) OPT 价值物品的情况下,可将所需箱体宽度从 1+ε 减小到 1,而高度保持不变。

关键词 u-分组,二维装箱,近似算法,NP 难度,启发式

中图法分类号 TP308,TP301.6 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2017.05.053

u-Block Optimization Algorithm for 2-D Strip Packing Problem

HUANG Hai[1] LI Song-bin[2,3]

(Department of Information Engineering, Putian University, Putian 351100, China)[1]

(Haikou Laboratory, Institute of Acoustics, Chinese Academy of Sciences, Haikou 570105, China)[2]

(National Network New Media Engineering Research Center, Chinese Academy of Sciences, Beijing 100190, China)[3]

Abstract The 2-D strip packing problem is the problem of loading a subset of a given set of rectangular boxes with profits into a rectangular container so that the stowed profits is maximized. The paper presented u-Block optimization algorithm to find more efficient parameter of block for any value. A new data structure was introduced for different classes of rectangles. At last, u-points of the X axis of the different kinds of rectangular boxes were filled to the maximum profits of the items. Two algorithms were developed, one is an effective block parameter u, another is block selection algorithm based on a simplified data structure. In particular, we presented polynomial time approximation for any value. In the case of polynomial time complexity and ε, we can reduce the required box width from 1+ε to 1, while the beight remains the same.

Keywords u-Block, 2-D packing problem, Approximation scheme, NP-hard, Heuristics

2-D packing 问题是一类关于空间利用最优化的 NP 难题。它研究在给定的矩形区域内如何对一组矩形模块进行互不重叠的最优布局,它在工程领域和实际生活中的应用都非常广泛,比如:板材切割、报纸版面编排、集成电路元件排放、运输的装箱等。2-D packing 问题是一类将给定的 n 个矩形物品尽可能多地装入箱子的问题。有些问题对箱子数目限定,有些也对箱子大小限定,还有些对是否允许对待装的矩形进行旋转的限定。如果待装物品带有价值,则考虑的是如何装入箱子使其价值最大。国内外解决 2-D packing 问题的方法可分为随机型和确定型算法。最新的随机型算法主要有遗传算法和模拟退火算法[1-3]等;确定型算法主要有启发式算法[4-7]和分支限界法[8]等。

2-D strip packing 问题是 2-D packing 问题装箱物体带有价值的一类问题。我们考虑如何将 n 个大小属于(0,1]×(0,1]且带有正价值的矩形物品在矩形不旋转的情况下装入长宽固定为 1 的箱子,使其价值最大化。在预期目标 ε>0 下,目前国内外能达到多项式时间复杂度的近似算法很多,例如 Jansen 和 Zhang[9]能取得(2+ε)-近似最优,Harren[10]将最优近似比提高到(1.25+ε)。也有一些通过扩充箱体取得最优解,其中最有代表性的是 Fishkin[11],其通过扩充箱体到[0,1+ε]×[0,1+ε]来保证装箱价值不低于(1−ε)OPT,其中 OPT 是最优装箱价值。本文算法对期望值 ε 计算最有效分组参数 u,并对矩形物品的长宽大小进行分类,对不同类别的矩形采用简化的数据结构和相应的装箱算法,在多项时间内

到稿日期:2016-09-18 返修日期:2017-01-17 本文受国家自然科学基金(61303249),福建省教育厅 A 类科技项目(JA15443),福建省莆田市科技项目(2014G16)资助。

黄 海(1977−),男,硕士,讲师,主要研究方向为智能计算、大数据的处理和分析;李松斌(1981−),男,博士,副研究员,CCF 会员,主要研究方向为多媒体信息处理、多媒体内容安全及取证等。

将价值至少为$(1-\varepsilon)OPT$的物品装入大小为$[0,1]\times[0,1+\varepsilon]$的箱体中。

1　问题描述

给出形式化定义：待装矩形物品$R=\{r_1,r_2,\cdots,r_n\}$，每个物品的价值为$p(r_i)$，长度表示为$l(r_i)$，宽度表示为$w(r_i)$，其中$p(r_i)\geqslant 0$，$l(r_i)\in(0,1]$，$w(r_i)\in(0,1]$。

问题的目标：选择R'中的一个子集R'装入长宽皆为 1 的箱体A中，使其价值$p(R')$最大；且装入的矩形不重叠也不交叉，矩形也不能旋转，只能保持原来的方向。其中$p(R')=\sum_{r_i\in R'}p(r_i)$。

2　u-分组优化算法的数据结构

2.1　u-分组策略

该策略的主要思想是：在期望目标$\varepsilon>0$下构造区间函数λ_i，并计算最有效分割参数u，基于参数u对装箱物品进行分成类。

基于期望目标ε，为实现构造区间函数λ_i，给出以下几个定义。

定义 1　$\varepsilon=\min(1/2,\varepsilon/4)$，其中$\varepsilon>0$为期望目标值。为方便处理，对$1/\varepsilon'$取偶数。基于$\varepsilon$构造$\varepsilon'$，使得本文算法可以得到所需要的近似比。

定义 2　$OPT=\sum_{r_i\in R^*}p(r_i)$，其中$S^*$为 2-D strip packing 问题的一个最优算法，R^*是S^*算法选择出的矩形的集合。OPT表示最优解R^*的总价值。

定义 3　区间函数$\lambda_j=(\lambda_{j-1})^{9/\lambda_{j-1}^2}$，$j\geqslant 2$，其中$\lambda_1=\varepsilon'$。构造区间函数$\lambda_j$用于装箱物品的分类。

定义 4　$R_j^*=\{r\mid r\in R^*,w(r)\in(\lambda_j,\lambda_{j-1}],l(r)\in(\lambda_j,\lambda_{j-1}]\}$，$j\geqslant 2$。利用区间函数$\lambda_j$，定义长宽介于$(\lambda_j,\lambda_{j-1}]$的矩形。

引理 1　在整数序列$2,3,4,5,\cdots,4/\varepsilon'+1$中存在一个整数$\eta$，使得$p(R_\eta^*)\leqslant(\varepsilon'/2)OPT$，其中$p(R_\eta^*)=\sum_{r_i\in R_\eta^*}p(r_i)$。

根据R_j^*的定义，$\forall r\in R^*$至多属于两个这样的集合，故$\sum_{j\geqslant 2}\sum_{r_i\in R_\eta^*}p(r_i)\leqslant 2OPT$，且由$\varepsilon'$的定义可得到引理 1。

由于η是一个常量，虽然最优解集合R^*是未知的，但可以在$O(n^{4/\varepsilon'})$复杂度下穷举所有可能的η，计算出η使得选出的箱体具有最大价值。

定义 5　$\varphi=9/u^2+1$，其中$u=\lambda_\eta$。

基于u的值对矩形物品分类，根据矩形的长度可分为长型(L-型)、短型(B-型)和中间型(M_L-型)3 类，它们的形式化定义为：$L=\{r\mid r\in R,l(r)>u\}$，$B=\{r\mid r\in R,l(r)\leqslant u^\varphi\}$，$M_L=\{r\mid r\in R,u^\varphi<l(r)\leqslant u\}$。同样，根据它们的宽可分为宽型(W-型)、窄型(N-型)和中间型(M_W-型)，相应的形式化定义为：$W=\{r\mid r\in R,w(r)>u\}$，$N=\{r\mid r\in R,w(r)\leqslant u^\varphi\}$，$M_W=\{r\mid r\in R,u^\varphi<w(r)\leqslant u\}$。以上对于类型矩形在最优解中对应的集合，我们也给出形式化定义：$L^*=L\cap R^*$，$B^*=B\cap R^*$，$M_L^*=M_L\cap R^*$，$W^*=W\cap R^*$，$N^*=N\cap R^*$，$M_W^*=M_W\cap R^*$。

因$u=\lambda_\eta$，$u^\varphi=\lambda_{\eta+1}$，故$M_L^*\cup M_W^*$，根据通过引理 1 计算出的$\eta$值，得到的价值$p(R_\eta^*)\leqslant(\varepsilon'/2)OPT$，故在后面处理矩形装箱时忽略$M_L^*\cup M_W^*$类型，即只需考虑 L-型、B-型、W-型和 N-型，它们最多能组成 LW-型、LN-型、BW-型和 BN-型。

2.2　LW-型和 LN-型的数据结构

其主要思想是：放大 LW-型和 LN-型长度到最接近u^2的某个倍数，装箱的位置对齐至分割线。

以箱体A的长作为笛卡尔坐标的 X 轴，宽作为 Y 轴，对 X 轴按间隔u^2画分割线。将R^*中 LW-型和 LN-型的矩形的长度放大到最接近某个u^2的倍数，放置位置首先从箱体A的左下角开始，沿着 X 轴水平右移直到它们放置的 X 轴坐标刚好是u^2的倍数。

引理 2　将R^*中 LW-型和 LN-型矩形的长度放大至最接近某个u^2的倍数，装入箱体A的长度至多增加$2u$。

由于 LW-型和 LN-型的长度至少为u，因此装入箱体A的长度至多增加$2u^2(1/u)=2u$。在后续装箱中将箱体A的长度增加到$1+2u$，记为A'。

2.3　BW-型和 BN-型的数据结构

主要思想是：引入碎片容器F来处理 BW-型和 BN-型，缩小F的宽度到F'，进而保证其价值损失不超过$(\varepsilon'/4)OPT$。

(1)生成容器F，并移走放置在F分割线上的箱体。

对箱体A'的 X 轴按间隔u^2等分，可分成$(1+2u)/u^2$个区间，每个区间简称长条。在 3.2 节放置 LW-型和 LN-型后将长条分割成更小的空白碎片，记为F(见图 1)。S^*中的装箱方案可能会使得 BW-型和 BN-型位置处于相邻的两个容器F的分割线上。通过移走右边容器部分 BW-型和 BN-型矩形腾出空间来装原来处于分割线上的矩形。首先对容器F沿 X 轴按间隔$3u^\varphi$进行分割，每个细块记为E。由于每个F可分为$u^2(3/u^\varphi)$个长度为$3u^\varphi$的细块E，其中$u^2(3/u^\varphi)>1/4\varepsilon$，因此$F$中至少存在一个细块$E'$($S^*$算法置于$E'$中的 BW-型和 BN-型物品)的价值至多为$(\varepsilon'/4)OPT(F)$，其中$OPT(F)$为$S^*$放置在$F$中的矩形总价值，$E'$的长度大小为$u^\varphi\leqslant l(E)\leqslant 3u^\varphi$。接着将$S^*$中置于$E'$块的矩形移除，得到新的空白区间$E'$。最后将$S^*$中位于两个相邻 F 容器分割线上的 BW-型和 BN-型矩形移到E'中，使得没有矩形放置在F的分割线上，这样处理后损失的价值至多为$(\varepsilon'/4)OPT$。

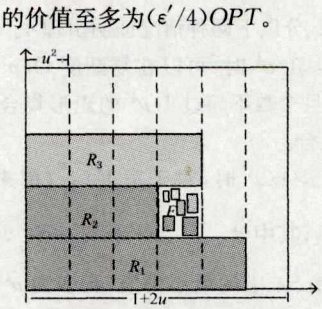

图 1　箱子分割图

(2)对F缩小宽度、放大长度，使其能装入同样的矩形。

对于S^*算法中至少有一个 BW-型装入容器F的问题，做如下处理：首先移除F中的所有 BN-型矩形，再将 BW-型沿着 Y 轴垂直往下移动至F的底部或位于它下面矩形的顶

部，在 F 内生成新装箱方案，它的宽度记为 $w(F)$，其中 $w(F)$ 至多为 $(1/u)w(BW)$，$w(BW)$ 是装入 F 中 BW-型宽度总和。将 F 的宽度缩小为 $w(F)+iu^{\varphi}$，i 取最接近这种结构的值。缩小 F 的宽度可能导致部分物品无法装入，我们通过放大 F 的长度来解决这个问题。由于 F 至多有 $1/u$ 个 BW-型矩形，因此通过计算并构造 F 新的长度和宽度得到引理 3。

引理 3 S^* 置于 F 中的矩形能被装在长度为 u^2+2u^4、宽度为 $w(F)+iu^{\varphi}$（其中 $i\leqslant n$）的容器 F'' 中。

用同样的方法将 F'' 的长度减少到 u^2，记为 F'，且损失的价值不超过 $4u^2OPT(C)$。由于装箱方案至多只有 $O(1/u^2)$ 个 F，因此处理所有的 F 损失的价值不超过 $4u^2OPT\leqslant(\varepsilon/4)OPT$。我们在 3.1 节中未考虑 $M_L\cup M_W$ 损失的价值为 $(\varepsilon'/2)OPT$，故目前为止损失的价值为 $(3\varepsilon'/4)OPT\leqslant\varepsilon'OPT$，再由引理 2 和引理 3 给出推论 1。

推论 1 存在算法 S'，使得装入箱体 A' 的矩形 R' 的总价值至少为 $(1-\varepsilon')OPT$，且其具有以下性质：1）R' 中的 LW-型和 LN-型长度已放大至最接近 u^2 的某个倍数，且它的放置起始位置对齐至箱体的分割线；2）对于至少含有一个 BW-型矩形的 F，它长度为 u^2，宽度为 $w'(F)+iu^{\varphi}$，其中 $i\leqslant n$。

3 u-分组优化算法的矩形选择策略

3.1 LW-型和 LN-型的选择

主要思想是：引入 LW-型和 LN-型的补集 $\overline{L_K^*}$，选择 K 个 LW-型和 LN-型价值最大的矩形，使其宽度总和不大于 $\overline{L_K^*}$ 且价值不小于 $(1-u)(1-u^4)P(\overline{L_K^*})$。

定义 6 $\overline{L_K^*}=(R^*\cap L)-L_K^*$，其中，$L=LW\cup LN$，$\overline{L_K^*}$ 为 R^* 中 k 个具有最大价值的 L-型矩形，k 是 $(0,n]$ 之间的整数。

定义 7 $\overline{L_{K_i}^*}=\{r\mid r\in\overline{L_K^*},l(r)=iu^2\}$，$i=1/u,1+1/u,\cdots,1/u^2$。

引理 4 对每个 $i=1/u,1+1/u,\cdots,1/u^2$，在多项式时间可以得到取值大小为 $O(n^{1+1/u^4})$ 的集合 $\Phi_i=\{y\mid y=\sigma+xw_l,x\in(0,1,\cdots,n-1/u^4)\}$，其中 σ 是长为 iu^2 的个数不超过 $1/u^4$ 个矩形的宽度总和，w_l 是算法 S 中最小价值矩形的宽度。更进一步，存在一个值 $\sigma'+x'w_l'\in\Phi_i$，使得 $\sigma'+x'w_l'\leqslant w(\overline{L_{K_i}^*})\leqslant\sigma'+(x'+1)w_l'$，$w_l'$ 是价值至多为 $u^4p(\overline{L_{K_i}^*})$ 的宽度。

对于整数 K，分以下两种情况证明引理 4。

（1）当 $|\overline{L_K^*}|\leqslant1/u^4$ 时，可以在复杂度 $O(n^{1/u^4})$ 内计算出所有长度为 iu^2 且个数不超过 $1/u^4$ 的矩形集合，从中找到满足条件的矩形集合。

（2）当 $|\overline{L_K^*}|>1/u^4$ 时，对于宽度 $\overline{L_{K_i}^*}$，用来近似表达式 $w(\overline{L_{K_i}^*})=\sigma+yw_l$，其中 $\sigma=\sum_{j=1}^{1/u^4}w(r_j)$，$r_j\in S_{K_i}^*$，$y\in\{0,1,\cdots,n-1/u^4\}$ 且 w_l 是 $S_{K_i}^*$ 中的最小价值（至多为 $u^4p(\overline{L_{K_i}^*})$）矩形 (r_m) 的宽度。计算 R 中长度为 iu^2 且个数不超过 $1/u^4$ 的所有可能集合，显然对于某个 $y\leqslant n-1/u^4$，有 $w(\overline{L_{K_i}^*})\in[\sigma+xw_l,\sigma+(x+1)w_l]$，从 $\overline{L_{K_i}^*}$ 中移除 r_m，则 $\overline{L_{K_i}^*}$ 的总宽度至多为 $\sigma+xw_l$ 且总价值至少为 $(1-u^4)p(\overline{L_{K_i}^*})$。对于整数 K，$\overline{L_K^*}$ 至多有 $1/u^2$ 个 $\overline{L_{K_i}^*}$ 且每个 $\overline{L_{K_i}^*}$ 至多有 $O(n^{1+1/u^4})$ 个不同宽度，故对于 K，所有可能的总宽度为 $O(n^{(1+1/u^4)1/u^2})$。对于 $\overline{L_{K_i}^*}$ 每个可能的宽度，利用 knapsack 算法[12]从 $O(n^{1+1/u^4})$ 个可能的集合中找到价值至少为 $(1-u)(1-u^4)p(\overline{L_K^*})$ 且总宽度不大于 $w(\overline{L_{K_i}^*})$ 的集合，即我们选择的矩形总价值至少为 $(1-u)(1-u^4)\sum_{i=1}^{1/u^2}p(\overline{L_{K_i}^*})$，它大于或等于 $(1-2u)\sum_{i=1}^{1/u^2}p(\overline{L_{K_i}^*})$，即价值至少为 $(1-2u)p(S_{LWLN}')$。

3.2 BW-型和 BN-型的选择

主要思想是：引入区间 A_{remain}，首先选择 BN-型矩形装入 A_{remain}，再选择 BW-型和 BN-型矩形装入 F'。

S 在 A' 中装入 LW-型、LN-型和 F' 剩下的空间记为 A_{remain}。首先利用 knapsack 算法[12]把 BN-型装入 A_{remain}，其中 ε 作为 knapsack 算法的期望目标，A_{remain} 作为 knapsack 的输入空间，得到 B-型装箱方案的价值至少是 S' 装箱策略的价值的 $1-\varepsilon$ 倍。接着利用 Lawler 算法[13]将 BW-型和 BN-型装入 F'，装入的矩形的物品宽度至少是 F' 长度的 $1/(\varepsilon')^4$。由推论 1 可知 S' 算法得到至多 $1/u^3$ 个 F'，其中 F' 的长度为 u^2，宽度为 $w'(F)+iu^{\varphi}$。每个 F' 计算所有 $O(n^{1+1/u})$ 个可能的宽度，则利用算法[13]装入 BW-型和 BN-型的总价值至少为 $(1-\varepsilon')p(S_{F'}')$，$p(S'_{F'})$ 是算法 S' 在 F' 中装入 BW-型和 BN-型的价值。

4 u-分组优化算法的装箱策略

主要思想是：首先将由 u-分组优化算法的矩形选择策略选择出的 F' 和 K 个最大价值的 LW-型和 LN-型装入箱体 A'，装入的箱体将 A' 水平横切，通过水平横切和基于 u 间隔的垂直切割，得到细块 *slot*；装箱策略换成矩形到 *slot* 位置的映射，利用线性规划找出有效的映射函数，完成 F' 和 K 个 LW-型和 LN-型的放置，最后利用 FFDW 算法将 BN-型装入未被占用的 *slot*。

引入相同大小的新矩形 $R_{F'}$ 替代 F'，其中 F' 装的是 BW-型和 BN-型矩形。假定 V 是算法 S 生成的矩形，则 V 可变成由 LW-型、LN-型、$R_{F'}$ 和装入 A_{remain} 的 BN-型矩形组成。用 V' 表示 V 中 $R_{F'}$ 和 K 个最大价值的 LW-型、LN-型矩形，$V''=V-V'$，V''_{BN} 表示 V' 中 BN-型矩形部分，V_L'' 表示 V' 中 LW-型、LN-型矩形部分。下面分别完成 V' 和 V'' 的装箱策略。

4.1 V' 的装箱策略

A' 按间隔 u^2 垂直划分成 $(1+2u)/u^2$ 个竖块，由 u-分组优化算法的矩形选择策略挑选出的 V' 将 A' 水平分割形成的横切块记为 *hor*，*hor* 从底到顶依次标号为 $hor(1),hor(2),\cdots$。横切线和竖切线形成的细块记为 *slot*。V' 在 A' 的位置可以转换成 V' 中矩形到 *slot* 的映射 f。引入 $V'\to2^z$ 映射函数 f，其中 $z=\{1,2,\cdots,(1+2u)/u^2\}$。对于每个 $r_j\in V'$，其中 $l(r_j)=n_ju^2$，$f(r_j)$ 对应的是 r_j 在 A' 中占用的 n_j 个连续的 *slot* 块。f 所有可能的映射函数为 $O(n^{(1+2u)/u^2})$，计算所有可能的 f，如果装箱方案 V 未能找到一映射函数 f 与它对应，则放弃 V，再重新选择一个新的装箱方案 V，直到找到有与 f 匹配的 V，最后找出具有最大价值的装箱方案 V 以及映射函数 f。

4.2 V'' 的装箱策略

对于 $\forall r_j\in V'$，r_j 存在于一串连续的 $hor(b_j),\cdots,hor(e_j)$

块中，b_j，e_j 分别表示 r_j 在 *hor* 的起始块编号。对每个 $hor(i)$，V' 占用的 *slot* 集合记为 $Occupied(i)$，用 FS 表示空闲 *slot* 所有可能的组合集合。用 V_L'' 表示 V'' 中的 LW-型和 LN-型矩形，用 V''_{BN} 表示 V'' 中的 BN-型矩形。对于每个 FS'，$FS'\in FS$，引入划分 $patition_{FS',i}=(FSBN_{FS',i},\Pi_{FS',i})$，$i=1,2,\cdots,n_{FS'}$，其中 $FSBN_{FS',i}\subseteq FS'$，用于 V''_{BN} 的装箱。$\Pi_{FS',i}$ 是 $FS'-FSBN_{FS',i}$ 连续的 *slot* 集合，用于 V_L'' 的装箱，而 $\Pi_{FS',i}$ 势为 t 的集合，用于对 V_L'' 中长为 tu^2 矩形进行装箱。对每一个 $patition_{FS',i}$，用 $x_{FS',i}$ 表示 $patition_{FS',i}$ 空闲块的总宽度，则 $patition_{FS',i}$ 中用于装 BN-型的空间大小为 $|FSBN_{FS',i}|u^2x_{FS',i}$。用 $w_i(V_L'')$ 表示 V_L'' 中长度为 iu^2 的矩形的总宽度，其中 $i=1/u,1+1/u,\cdots,1/u^2$。因 $|V'|\leqslant K+u^{-3}$，故 *hor* 个数至多为 $g=2(K+u^{-3})$。对于每个 $r_j\in V'$，为确定 r_j 所属的 *hor*，由于 V' 的个数为 $|V'|\leqslant K+u^{-3}$，故我们可以在时间复杂度为 g^2 时穷举计算得到它的起始编号 b_j，e_j。利用线性规划 $LP(f,b,e)$ 给出可行解 (h^*,q^*,x^*)，h^* 表示 *hor* 中的位置编号，q^* 表示 *hor* 空闲 *slot* 的宽度。

(1)V_L'' 的装箱策略：在每个 $[h_j^*,h_{j+1}^*)$ 进行 V_L'' 的装箱，其中 $0\leqslant j\leqslant g^*$。首先对所有的划分 $partition_{FS',i}$ 按 $FSBN_{FS',i}$ 进行主关键字排序，使得 *hor* 中相邻 $|FSBN_{FS',i}|$ 的 *slot* 块能装入 BN-型矩形。用 w_{j+1}^* 表示 $[h_j^*,h_{j+1}^*)$ 的宽度。对于 V_L'' 中长为 au^2 的矩形集合 $r_a=\{r_{a,1},\cdots,r_{a,n_a}\}$，$a$ 取值范围为 $1\sim 1/u^2$，对排序好的 $partition_{FS',i}$ 在 $\Pi_{FS',i}$ 中选择满足条件 X，使得 $X\in\Pi_{FS',i}$，其中 $r_{a,j}\in r_a$ 且 $a=|X|$。将 $r_{a,j}$ 装入 $[h_j^*,h_{j+1}^*)$ 中的 X 集合中，并将 $r_{a,j+1},r_{a,j+2},\cdots$ 依次装入，直到装入的总宽度不超过 w_{j+1}^*，对于最后一个刚好超过 w_{j+1}^* 的矩阵，把它切成两块，将超过宽度的那半块并入剩余的矩形中，重复上述过程直至所有的矩形装入或 $[h_j^*,h_{j+1}^*)$ 无空间再装入更多的矩形；对后续的 $[h_{j+1}^*,h_{j+2}^*)$ 采用同样的处理策略。

(2)V''_{BN} 的装箱策略：对于每个 $[h_j^*,h_{j+1}^*)$，其中还存在长度为 $|FSBN_{FS',i}|u^2$ 的空闲 *slot*，而它被 V' 和 V_L'' 分割成至多 $1+z/2$ 个小块 *block*。去掉其中宽度小于 $4u^{\varphi-1}$ 的小块，对每个长为 tu^2、宽 w_b 大于或等于 $4u^{\varphi+1}$ 的块 b_k，利用 FFDW 算法[14]对 V''_{BN} 装箱。由 FFDW 算法[14]可将 V''_{BN}（长宽至多为 u^φ）装入长为 tu^2、宽至多为 $u^\varphi+AREA(V''_{BN})(1+u^2)/tu^2$ 的区域，即长为 tu^2、宽至多为 $u^\varphi+(tu^2w_b+u^{2\varphi})(1+u^\varphi)/tu^2\leqslant w_b+3u^\varphi$。对 b_k 按间隔 $4u^\varphi$ 进行水平切割，则至少可分为 $1/u$ 个水平切割，移走装入最小价值的一块，则损失的价值至多为 $up(V''_{BN})$，且它使得 FFDW 算法能全部将 V''_{BN} 装入在这些块中，故处理 V''_{BN} 的装箱时损失的价值至多为 $up(S'_{BN})$。

5　算法和实验分析

下文证明本文算法能在多项式时间内将价值至少为 $(1-\varepsilon)OPT$ 的箱体装入，大小为 $[0,1]\times[0,1+\varepsilon]$ 的箱体中，并通过实验验证算法的高效性。

定理 1　对于矩形 $R=\{R_1,R_2,\cdots,R_n\}$，本文算法 S' 的时间复杂度是多项式时间，对于任意 $\varepsilon>0$，能找到 $R'\subseteq R$，使得 R' 能装入大小为 $[0,1]\times[0,1+\varepsilon]$ 的箱体，且总价值 $p(R')$ 至少为 $(1-\varepsilon)OPT$，OPT 为最优解。

证明：首先证明所需箱体的大小为 $[0,1]\times[0,1+\varepsilon]$。在装箱 V_L'' 矩形时，有些矩形可能分成两个以装入不同的 *hor*，如果移走这部分矩形则可以保证失去价值至多为 $3\varepsilon'OPT$。在装箱 V''_{BN} 矩形时，对每个 *block* 去掉宽小于 $4u^{\varphi-1}$ 的块，总的浪费空间至多为 u，因此至多有大小为 u 的 V''_{BN} 未被装入，由 4.2 节可知，利用 FFDW 算法装入这些未装入的 V''_{BN} 时至多需要 $2u$ 的空间，再加上 A' 的空间，箱体所需长度至多为 $1+4u\leqslant 1+\varepsilon$。

其次证明 $p(R')$ 至少为 $(1-\varepsilon)OPT$，本文算法 S' 获得的价值有：1)装入 K 个 LW-型和 LN-型矩形的价值；2)F' 中装入 BW-型和 BN-型矩形的价值；3)A_{remain} 处理 V''_{BN} 时装入 BN-型矩形的价值。本文算法 S' 损失的价值有：1)放大 LW-型和 LN-型长度的损失的价值；2)处理 F' 损失的价值；3)忽略 $M_L'\cup M_W'$ 类型装箱的价值。故总价值为 $(1-2u)p(S'_{LWLN})+(1-\varepsilon')p(S'_F)+(1-u)p(S'_{BN})-(3\varepsilon'/4)OPT\geqslant(1-2\varepsilon')p(S'_{LWLN})+(1-\varepsilon')p(S'_F)+(1-\varepsilon')p(S'_{BN})-(3\varepsilon'/4)OPT>(1-2\varepsilon')p(S'_{LWLN})+(1-2\varepsilon')p(S'_F)+(1-2\varepsilon')p(S'_{BN})-(3\varepsilon'/4)OPT>(1-2\varepsilon')p(S')-(3\varepsilon'/4)OPT\geqslant(1-2\varepsilon')(1-\varepsilon')OPT-(3\varepsilon'/4)OPT$，即 $(1-3\varepsilon')OPT-(3\varepsilon'/4)OPT>(1-4\varepsilon')OPT=(1-\varepsilon)OPT$。而算法 S' 的时间复杂度为多项时间 $O(n^{(4/u^2)^{5/u^2}})$。

通过常用的测试数据集验证 u-分组优化算法的效率。该数据集来自文献[15]，记为 A 类测试集，并将其与几种针对 strip packing 问题的目前较新的算法进行对比。所有算法使用 C 语言实现，使用计算机平台（CPU 为 Inter2. 0GHz，内存为 2GB）进行测试。测试集的所需箱体的高度最优解是已知的，用 S^* 表示；用 n 表示装载矩形的个数，用 W 表示箱体的宽度，用 *Avg* 表示多次实验结果的平均值，用 *Best* 表示几次实验获得的最优结果，实验结果对比如表 1 所列。

表 1　A 类测试集上 u-分组与其他算法计算结果的比较

				hyper-heuristic[4]		R. Harren[10]		u-分组优化	
数据	n	w	S^*	Avg	Best	Avg	Best	Avg	Best
A1	25	40	16	17.56	16	16.37	16	16.45	16
A2	35	40	28	30.43	28	29.49	28	30.12	28
A3	45	50	49	53.43	50	52.21	49	51.35	49
A4	75	60	58	62.43	61	61.35	59	60.14	58
A5	95	60	72	79.38	76	78.20	75	75.96	72

从表 1 可以看出，在装载矩形个数较少时，3 种算法都可以得出最优解，但后两种算法的平均解略优于第一种算法；随着装载矩形个数不断增大，第一种算法无法求得最优解，而后两种还能求得最优解；当矩形个数达到 95 时，3 种算法都无法取得最优解，但 u-分组优化算法的平均解优于前两种算法。当问题规模较大时，u-分组优化算法具有优势。因此该测试也验证了 u-分组优化算法是一种高效的算法。

结束语　本文通过计算最优分组参数，提出了一种新的 u-分组优化算法。利用分组参数对装载矩形进行分组，并对不同组别的矩形采用不同的数据结构，结合箱体 u 等分进行填充，并从理论上证明了算法能将所需箱体大小减小到 $[0,1]\times[0,1+\varepsilon]$，最后实验也表明了本算法的高效性。但本算法处

（下转第 303 页）

zation of the NL-Means: application to image and video denoising[J]. IEEE Transactions on Image Processing, 2014, 23(8): 3506-3521.

[9] MAHMOUDI M, SAPIRO G. Fast image and video denoising via non-local means of similar neighborhoods[J]. IEEE Signal Processing Letters, 2005, 12(12): 839-842

[10] ZHANG X D, FENG X C, Wang W W. Two-direction nonlocal model for image denoising[J]. IEEE Transactions on Image Processing, 2013, 22(1): 408-412.

[11] XIAO J S, LI W H, JIANG H, et al. Three dimensional block-matching video denoising algorithm based on dual-domain filtering[J]. Journal on Communications, 2015, 36(9): 91-97. (in Chinese)
肖进胜，李文昊，姜红，等. 基于双域滤波的三维块匹配视频去噪算法[J]. 通信学报，2015，36(9)：91-97.

[12] DABOV K, FOI A, KATKOVNIK V, et al. Image denoising by sparse 3-d transform-domain collaborative filtering[J]. IEEE Trans. on Image Process, 2007, 16(8): 2080-2095.

[13] ZHANG L, DONG W S, ZHANG D, et al. Two-stage image denoisng by principal component analysis with local pixel grouping [J]. Pattern Recognition, 2010, 43(4): 1531-1549.

[14] HE Y M, GAN T, CHEN W F, et al. Adaptive denoising by singular value decomposition[J]. IEEE Signal Processing Letters, 2011, 18(4): 215-219.

[15] GU S, ZHANG L, ZUO W, et al. Weighted nuclear norm minimization with application to image denoising[C]// 2014 IEEE Conference on Computer Vision and Pattern Recognition. Columbus, OH, 2014: 2862-2869.

[16] JIN J G. Review of clustering method[J]. Computer Science, 2014, 41(11A): 288-293. (in Chinese)
金建国. 聚类方法综述[J]. 计算机科学，2014，41(11A)：288-293.

[17] LUXBURG U V. A tutorial on spectral clustering[J]. Statistics & Computing, 2007, 17(17): 395-416.

[18] CHUNG F. Spectral Graph Theory[M]. Am. Math. Soc, 1997.

[19] ZHENG Q, LIU Z. Research on improved normalized cut spectral clustering algorithm[C]// 2016 Chinese Control and Decision Conference (CCDC). Yinchuan, 2016: 1981-1984.

(上接第 293 页)

理装载矩形时是基于不旋转的情况的，仍未对可以旋转的物品进行研究，后续研究工作将着眼于处理可以旋转的装载矩形。

参 考 文 献

[1] HAR-PELED S, KAPLAN H, SHARIR M. Approximating the k-level in three-dimensional plane arrangements[C]// Acm-siam Symposium on Discrete Algorithms. 2016: 1193-1212.

[2] NADIRADZE G, WIESE A. On approximating strip packing with a better ratio than 3/2[C]// Acm-siam Symposium on Discrete Algorithms. 2016: 1491-1510.

[3] FERNAU H, LÓPEZ-ORTIZ A. Using Parametric, Transformations Toward Polynomial Kernels for Packing Problems Allowing Overlaps[J]. ACM Transactions on Computation Theory, 2015, 7(3): 1-29.

[4] LÓPEZ-CAMACHO E, TERASHIMA-MARIN H. A unified hyper-heuristic framework for solving bin packing problems[J]. Expert Systems with Applications, 2014, 41(15): 6876-6889.

[5] ASTA S, OZCAN E. A Tensor Analysis Improved Genetic Algorithm for Online Bin Packing[C]// Genetic & Evolutionary Computation Conference. New York: ACM Press, 2015: 799-806.

[6] ZHANG D F, PENG Y, ZHANG L L. A Multi-Layer Heuristic Search Algorithm for Three Dimensional Container Loading Problem[J]. Chinese Journal of Computers, 2012, 35(12): 2553-2561. (in Chinese)
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[J]. 计算机学报. 2012，35(12)：2553-2561.

[7] LIU S, ZHU F H, LV Y S, et al. A Heuristic Orthogonal Binary Tree Search Algorithm for Three Dimensional Container Loading Problem[J]. Chinese Journal of Computers, 2015, 38(8): 1530-1543. (in Chinese)
刘胜，朱凤华，吕宜生，等. 求解三维装箱问题的启发式正交二叉树搜索算法[J]. 计算机学报，2015，38(8)：1530-1543.

[8] SOSA-ASCENCIO A, TERASHIMA H. Conant-Pablos Grammar-based Selection Hyper-heuristics for Solving Irregular Bin Packing Problems[C]// Genetic and Evolutionary Computation Conference. New York: ACM Press, 2016: 111-112.

[9] JANSEN K, ZHANG G. Maximizing the number of packed rectangles[M]// Scandinavian Workshop on Algorithm Theory (SWAT). 2004: 362-371.

[10] HARREN R. Approximating the orthogonal knapsack problem for hypercubes[C]// International Colloquium on Automata, Languages and Programming(ICALP). 2006: 238-249.

[11] FISHKIN A V, GERBER O, JANSEN K, et al. Packing weighted rectangles into a square[C]// Symposium on Mathematical Foundation of Computer Science(MFCS). 2005: 352-363.

[12] LAWLER E. Fast approximation algorithms for knapsack problems[J]. Mathematics of Operations Research, 1977, 4(4): 339-356.

[13] FISHKIN A V, GERBER O, JANSEN K. On weighted rectangle packing with large resources[C]// Theoretical Computer Science Conference(TCS). 2004: 237-250.

[14] COFFMAN E G. Performance bounds for level-oriented two-dimensional packing algorithms[J]. SIAM Journal on Computing, 1980, 9(4): 808-826.

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