三 維 物 件 置 放 問 題 之 新 演 算 法

A New Approach to Three-Dimensional Objects Loading Problem

謝財明* 林佑政+ 曾淑苑* 傅文佑*

*中原大學資訊工程系所 +中原大學電子所資訊組

台灣省桃園縣中壢市普忠里普仁 22 號

hsieh@mbox.cycu.edu.tw linyu@cad.ice.cycu.edu.tw s8527234@ice.cycu.edu.tw s8526434@ice.cycu.edu.tw

摘要

本論文中，我們提出一個能有效解決三維物件堆放緊縮問題的方法。本問題的基本構成物件包括矩形體，立 T 形體及立體 L 形體。這些物件在堆放的過程中，允許平移及旋轉。首先，我們提出描述各種物件的模式，並提出能很簡單描述各種物件旋轉的模式，接著提出一以模擬退火方法爲基礎之緊縮方法，這方法是一種層次式的結構，因而可以有效地去解決一包含許多物件的緊縮問題，最後我們將所求出的結果做進一步分析，考慮每一物件堆放的順序與位置，並與現實情況中的機械手臂運用及貨櫃開啓形式相結合，以期快速有效地處理堆放問題。實驗證明，本文所提之方法有極佳效率。

關鍵詞：模擬退火法; 拓樸排序;緊縮;貨櫃;托板。

1 簡介

如何將已知形狀、大小的物件或物件放置在一最小的空間中，或者這些物件放置在一有限的空間中，使得浪費的空間最少是個重要的課題。例如，在 VLSI 線路設計中，我們希望能在滿足所有設計法則的條件下，重新安排線路佈圖的幾何位置，使得面積儘可能的縮小，以降低成本，提高產品的良率，這是一個典型的二維緊縮(2-D Compaction)問題，同時這問題已被證明是 NP-complete 問題 [8,11,14] 。另一方面，在自動化工廠設計中，貨櫃及倉儲貨物的堆放，乃是希望將很多不同大小的貨物堆放一有限的空間中，使得空間的使用率提高，以減少運貨托板 (pallet) 的數目，減少運輸配銷成本 (transportation-distribution cost) [5,6,12,13,15]，這個問題的研究，大部份局限在二維的堆放，即令每一貨物具有相同的高度 [12,5]，而所使用的解法，主要有線性規劃(linear programming) [3,4,15] 及動態規劃(dynamic programming) [12]。由於貨物大小的不同，這個問題實際上是一個三維緊縮 (3-D Compaction) 的問題，也就是說希望將一些大小不同的立體物件堆放在最小的空間。近來，在自動化的工廠中，常利用機器手臂來堆放貨物，因此，如何規劃機器手臂自動將貨物堆放在最小的空間就變得很重要。另外，如在一貨櫃或一托板置放物件，貨櫃的形式可分爲完全開放式、雙門及單門等，錯誤的堆放次序可能導致雖仍有足夠的空間卻無法再繼續堆放物件的情況發生，因此物件的堆放次序亦是一個相當重要的課題。

2 限制條件

論文中，我們引用[16]所提之描述矩形立方體模式。一矩形方體 B_i 的位置大小可由它的左下角座標 x_i,y_i,z_i 及右上角座標 $(\hat{x}_i,\hat{y}_i,\hat{z}_i)$ 來決定，其長寬高分別爲 l_i,w_i,h_i 即 $l_i=\hat{x}_i-x_i$，$w_i=\hat{y}_i-y_i$，$h_i=\hat{z}_i-z_i$，如 Figure 1 所示。

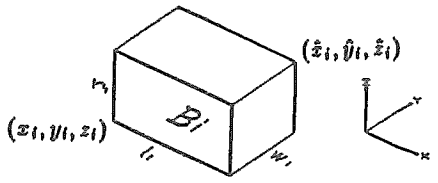

Figure 1 矩形方體

本文所討論之物件，以矩形方體及立體T形爲基本構成物件。立體 T 形體是由二個相連在一起的矩形方體所構成，L 形體則視爲 T 形體的特例，此兩矩形方體，我們稱爲配對物件 (paired components)。在緊縮的過程中，這些基本構成物件可以矩形之稜線 (平行於 X,Y,Z)做 $\pm n\pi/2$ (n=0,1,2,3) 角度的旋轉。

假設共有 n 個矩形方體物件，全部放置在第一象限，此 n 個矩形方體的大小及其位置，可 6 個實數集合來表示，即：

$$X=\{x_1,x_2,\cdots,x_n\} \qquad \hat{X}=\{\hat{x}_1,\hat{x}_2,\cdots,\hat{x}_n\}$$
$$Y=\{y_1,y_2,\cdots,y_n\} \qquad \hat{Y}=\{\hat{y}_1,\hat{y}_2,\cdots,\hat{y}_n\}$$
$$Z=\{z_1,z_2,\cdots,z_n\} \qquad \hat{Z}=\{\hat{z}_1,\hat{z}_2,\cdots,\hat{z}_n\}$$

其中，

$$x_i\ge 0,\hat{x}_i\ge 0,y_i\ge 0,\hat{y}_i\ge 0,z_i\ge 0,\hat{z}_i\ge 0, \qquad i=1,2,\cdots,n.$$

設 $L=X\cup\hat{X}\cup Y\cup\hat{Y}\cup Z\cup\hat{Z}$ 若我們對 L 中每一變數付予一非負之實數值，可得一非負實數集合 P，稱爲一佈圖 (layout)，而一佈圖之大小，以 $size(P)$表示，爲包含所有矩形方體之最小矩形的體積，亦即 P 之邊界方體 (bounding rectangular block)之體積，即：

$$size(P)=\max\{x|x\in\hat{X}\}\cdot\max\{y|y\in\hat{Y}\}\cdot\max\{z|\in\hat{Z}\} \qquad (1)$$

定義 $\mu(P)$ 爲佈圖 P 之體積使用率(Volume utilit)即所有的矩形方體體積的和除以佈圖 P 之邊界方體之體積。即：

$$\mu(P)=\frac{\sum_{i=1}^{n}\ell_i\omega_i h_i}{size(p)}$$

本論文主要的探討如何先藉矩形方體物件的平移、旋轉，以求得一最小體積的佈圖，亦即此佈圖的最大空間使用率，並依放置空間的不同限制條件而計算出各物件放置的次序。

物件之大小、形狀及各物件間的相對位置關係，可以一組下界(lower bound)型式之不等式來描述[14,16]，稱爲此佈圖之限制條件 (constraints)。若 $l_i \in X \cup \hat{X}, i=1,2,$ ，且 $l_1+d \leqq l_2$，$d \geqq 0$，表示 l_1 與 l_2 至少相距 d 單位長度，若 $d<0$，則表示 l_1，l_2 至多相距 $|d|$ 單位長度，此稱爲一 x-限制條件(x-constraint)。同法，可定義 y，z 的限制條件。

限制條件，可分爲：(1)大小 (size)，(2)連接 (connection)，(3)最小矩離(minimum distance)，(4)使用者指定 (user specified)，四類，分別說明如下：

(1)*大小限制條件*：用以界定一矩形方體物件之大小。一矩形方體 B_i，其長度爲 l_i，寬度爲 w_i，高度爲 h_i，如 Figure 1 所示。可用下列限制條件描述：

$$\begin{aligned} &x_i+l_i \leq \hat{x}_i, \hat{x}_i-l_i \leq x_i \\ &y_i+w_i \leq \hat{y}_i, \hat{y}_i-w_i \leq y_i \\ &z_i+h_i \leq \hat{z}_i, \hat{z}_i-h_i \leq z_i \end{aligned} \qquad (2)$$

(2)*連接限制條件*：用以界定矩形方體物件間相連之關係。本論文中，除矩形方體外之另一種基本結構，立體 T 形體，L 型方體，即以此類限制條件來達成。兩矩形體 B_i，B_j 相連，我們稱之爲配對物件，其關係依 B_i 在 B_j 之左面、右面、前面、後面、下面及上面可分爲 6 種情形。如 Figure 2 所示，B_i 在 B_j 之左面可表爲

$$\hat{x}_i \leq x_j, x_j \leq \hat{x}_i, y_i+r_{ij}^1 \leq \hat{y}_j,$$
$$y_i+r_{ij}^2 \leq \hat{y}_i, z_i+r_{ij}^3 \leq \hat{z}_j, z_j+r_{ij}^4 \leq \hat{z}_i,$$

r_{ij}^1：B_i 左邊至 B_j 右邊之距離。

r_{ij}^2：B_j 左邊至 B_i 右邊之距離。

r_{ij}^3：B_i 下邊至 B_j 上邊之距離。

r_{ij}^4：B_j 下邊至 B_i 上邊之距離。

此符號前兩項表示 B_i 與 B_j 兩者 x 座標之關係，此符號第三、四項表示兩者 y 座標之關係，此符號第五、六項表示兩者 z 座標之關係。

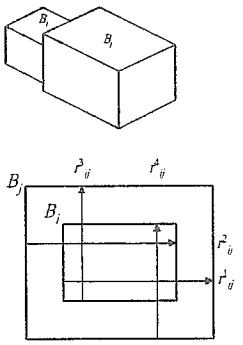

Figure 2 矩形方體相之關係

(3)*最小距離限制*：用來描述不相連物件之位置關係。若物件 B_i 與 B_j 不允許相疊(overlap)，且至少要相距某一距離，則它們間可建立 6 個最小距離限制關係：

$$\begin{aligned} &C_{ij}^1 \ (B_i \text{在} B_j \text{之左面}) : \hat{x}_i+d_{ij}^1 \leq x_j \\ &C_{ij}^4 \ (B_i \text{在} B_j \text{之右面}) : \hat{y}_j+d_{ij}^4 \leq y_i \\ &C_{ij}^2 \ (B_i \text{在} B_j \text{之前面}) : \hat{x}_j+d_{ij}^2 \leq x_i \\ &C_{ij}^5 \ (B_i \text{在} B_j \text{之後面}) : \hat{z}_i+d_{ij}^5 \leq z_j \\ &C_{ij}^3 \ (B_i \text{在} B_j \text{之下面}) : \hat{y}_i+d_{ij}^3 \leq y_j \\ &C_{ij}^6 \ (B_i \text{在} B_j \text{之上面}) : \hat{z}_j+d_{ij}^6 \leq z_i \end{aligned} \qquad (3)$$

其中：

$$d_{ij}^k \geq 0, k=1,2,3,4,5,6$$

若上述 6 條件中至少有一條件成立，則表示 B_i 與 B_j 不相疊。當 $d_{ij}^k=0$，k=1,2,3,4,5,6 時，則表示 B_i，B_j 可相連，但不能相疊。

(4)*使用者指定限制*：允許使用者自行訂定佈圖之條件，可界定矩形方體間最大、最小相連關係，或某方形放置時的特殊要求等。

所謂一有效佈圖 (valid layout)，即滿足上述基本限制及最小距離限制，及使用者指定限制之佈圖。此外，本論文之各基本構成物件，允許沿 X 軸，Y 軸，Z 軸作±π/2、±π、±3π/2 之旋轉，或限制不可旋轉[16]。

3 堆放程序

堆放程序依物件的類型可分爲基本物件堆放程序與配對物件堆放程序兩類，這兩類物件在堆放時，程序上會有所不同，分別說明如下：

(1) *基本物件堆放程序*：

在[16]中，先輸入各矩形方體的編號、長、寬、高，以及各種限制條件，並建立一限制圖，再以模擬退火法，去求一具有最佳空間利用率之解。此解包括各矩形方體所在的座標(x_i,y_i,z_i)及$(\hat{x}_i,\hat{y}_i,\hat{z}_i), i=1,2,...,n$。接著仍有一個重要的問題需要考慮，置放物件的貨櫃或托板未必是完全開放式的，也就是說也許貨櫃僅有兩個甚至一個出入口可供物件堆放及搬運。以 Figure 3 爲例，有七個基本物件欲置入一單門貨櫃中，如果物件 2、3、7 先放入貨櫃中，則很明顯地雖然貨櫃內有足夠的空間，但物件 6 卻無法置入櫃中。我們以 X 軸代表貨櫃的前後深度，Y 軸代表貨櫃的寬度而 Z 軸則代表貨櫃的高度，在堆放過程中，我們須對三軸考慮其堆放影響性，再視貨櫃開啓的形態決定三軸影響之優先次序。就 Figure 3 的範例而言，由於貨櫃僅於右方開啓，因此堆放的次序應先以 X 軸的影響爲優先。而因重力影響的緣故，因此堆放的次序其次應以 Z 軸之影響爲主。最後，爲了便利減少機械臂移動距離，再考慮物件在 Y 軸的影響。

爲解決堆放次序問題，我們發展了一新的堆放程序演算法，此演算法可針對不同堆放環境而求出可行之堆放次序解。

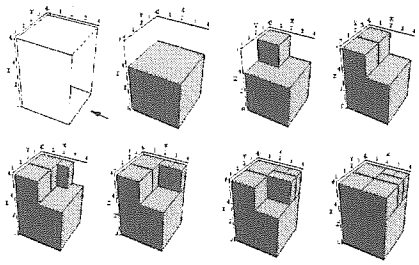

Figure 3 單門貨櫃之基本物件堆放

在本程序，輸入爲已緊縮後的基本構成物件座標位址，而輸出爲合法的堆放次序。定義堆放影響限制圖(Loading Constraint Graph ,LCG) $G(V,E)$ 爲 $V=\{v \mid v$ 爲物件編號$\}$，$E=\{(v_i,v_j) \mid v_i,v_j \in V, v_i$ 須比 v_j 先放置$\}$。Figure 3 範例中的七個物件座標位址分別爲：

V_i	(x_i,y_i,z_i)	$(\hat{x}_i,\hat{y}_i,\hat{z}_i)$
1	(2,3,4)	(3,4,6)
2	(3,3,4)	(4,4,6)
3	(2,2,4)	(4,3,6)
4	(0,0,0)	(4,4,4)
5	(0,2,4)	(2,4,6)
6	(0,0,4)	(2,2,6)
7	(2,0,4)	(4,2,6)

我們依單門貨櫃的物件出入限制因素及重力因素加以考慮後，可得到如圖 4 的堆放影響限制圖 $G(V,E)$。在 Figure 4 中，每個節點代表物件的代號，而有向邊則代表兩物件間的堆放次序限制。如節點 4 與節點 5 之間有一條標註爲 Z 的有向邊，其代表意義爲物件 4 與物件 5 因受到重力因素(影響到 Z 軸)的緣故，在堆放過程中，物件 4 一定要比物件 5 優先堆放入貨櫃中。當得到此有向無迴圈圖(Directed Acyclic Graph, DAG)後，我們再依拓樸排序法(topological sort)決定出物件堆放次序。

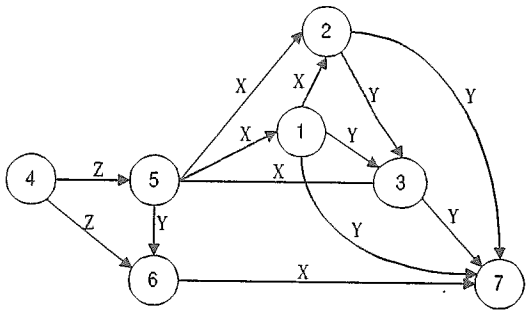

Figure 4 七個基本物件在單門貨櫃的堆放影響限制圖

Figure 5 中將 10 項物件置入一單門貨櫃中，依前述之演算法可得到一合理次序解(feasible ordering solution)，其堆放依序爲 2,1,3,6,4,5,7,9,10,8。然而，合理次序並未必唯一，如物件 1 與物件 2 即使置放次序相反也不會影響堆放過程。只是我們爲了降低機械臂的移動距離，故讓物件儘可能由右邊先堆置。若考慮皆由左方優先堆置，仍屬一組合理次序解。

Algorithm LOB (Loading-Order for Basic Blocks)

Input: Coordinates of blocks

Output: Loading sequence

Step1: Construct LCG;

Step2: Apply topological sorting algorithm to LCG;

Step3: Output loading sequence;

其中 Step1 之 LCG 會因托板或貨櫃開啓情況有些差異，以下爲以單門貨櫃爲例之 Construct LCG algorithm：演算法中(xh[i]、yh[i]、zh[i])即代表之前所定義之$(\hat{x}_i,\hat{y}_i,\hat{z}_i)$。若考慮將物件堆放在一托板上，座標考慮的次序就不再是以 X 座標爲優先，則 Algorithm C 中 X、Y、Z 座標的考慮次序就可交換。

若物件欲置入的貨櫃僅有一個位於 X 軸正向的門，所開啓的門超過一個以上時，本演算法仍可輕易延伸套用。假設貨櫃僅於於 X 軸正向開啓一門時之堆放影響限制圖爲 $G(V,E)$，。若此貨櫃於 X 軸正向與 X 軸負向皆開啓一門時，則除了 $G(V,E)$外，由 $E'=\{(v_j,v_i) \mid (v_i,v_j) \in E\}$，我們可得另一組堆放影響限制圖 $G'(V,E')$，經由針對 $G(V,E)$ 與 $G'(V,E')$做拓樸排序，則可得到至少兩組合理次序解。

```
Algorithm CL (Construct LCG)
for(i=1;i<=n;i++)
  for(j=1;j<=n;j++)
    if (i!=j)
      {
       if (x[i]<x[j])
          if   (((yh[i]>y[j])&&(yh[j]>y[i]))
                &&((zh[i]>z[j])&&(zh[j]>z[i])))
                     addedge(i,j);
          else {
               if (y[i]>y[j])
                    if   (((xh[i]>x[j])&&(xh[j]>x[i]))
                          &&((zh[i]>z[j])&&(zh[j]>z[i])))
                               addedge(i,j);
                    else {
                         if (z[i]<z[j])
                              if   (((xh[i]>x[j])&&(xh[j]>x[i]))
                                    &&((yh[i]>y[j])&&(yh[j]>y[i])))
                                         addedge(i,j);
                              };
               };
       }; /* end of (i !=j) */
```

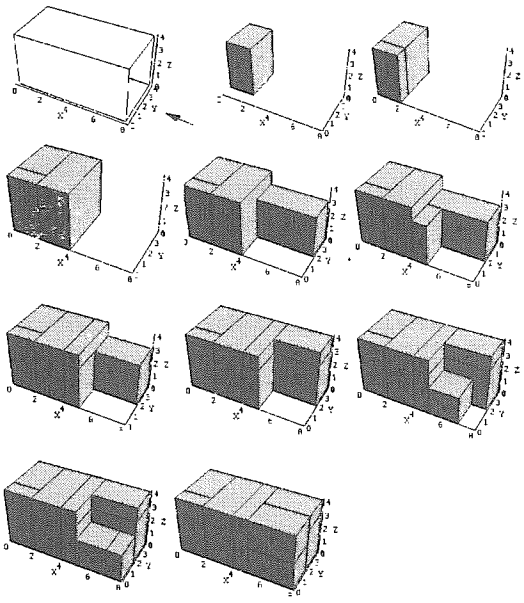

Figure 5 十個基本物件堆至單門貨櫃的堆放流程

若考慮單門貨櫃開啓位置，假設開啓方向在正上方(即 Z 軸正向位置)，則建堆放影響限制圖時就須先以 Z 軸次序爲優先，其次再視需求以 Y 或 X 軸次序爲第二優先。建出圖後，經由拓樸排序即可得到合理次序解。

(2) *配對物件堆放程序*

配對物件比起基本物件須考慮到更多堆放時的限制條件，以 Figure 6 爲例，假設物件 3、4 爲配對物件，物件 5、6 爲另一組配對物件，也就是說共有四個物件欲堆放至 Figure 6(a)中的單門貨櫃中，其中包含兩個 L 形體物件，如 Figure 6(a)所示。

在[16]中，我們可將 L 形體的物件視爲兩個矩形立方體的組合，並加上連接限制條件以得到緊縮解。而在堆放程序時，連接限制條件亦須納入考慮，在此例子中，以基本物件堆放程序而言，3,2,1,6,4,5 爲一組合理次序解，但卻違反了連接限制條件。

定義(x,y)表示物件 x 與物件 y 爲配對物件，如 Figure 6 中的(6,5)、(3,4)皆屬配對物件。考慮另一組次序解：(3,4),(6,5),1,2，雖然滿足連接限制條件，但當配對物件(6,5)與(3,4)先置入櫃中時，物件 1 與物件 2 則無法再置入，因此序列(3,4),(6,5),1,2 並非配對物件的合理次序解，

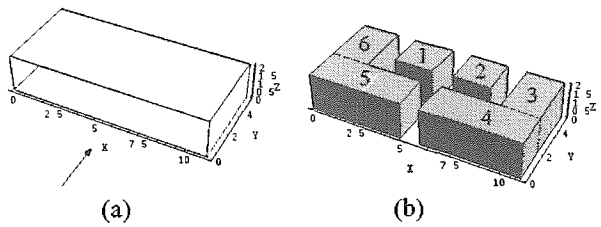

Figure 6 配對物件堆至單門貨櫃的堆放流程

由於配對物件堆放程序須考慮到連接限制，因此合理次序解未必存在。以 Figure 7 爲例，假設欲將 Figure 7(b)的四個物件置入 Figure 7(a)的單門貨櫃內，Figure 7(b)中的四個物件座標分別爲{1：(0,0,6),(2,2,8)，2：(0,0,3),(3,2,5)，3：(0,0,0),(3,2,2)，4：(3,2,0),(5,2,8)}，由於重力的影響，因此物件堆放次序 3,2,1；因堆放空間影響，故次序 1,4、2,4、3,4；但是由於考慮連接限制，物件 2 與物件 4 堆放次序之間不可介入其他物件。物件 2 與物件 4 必須視爲一物件。Figure 8(a)中的虛線表示物件 2 與物件 4 爲配對物件，因此我們可將節點 2 與節點 4 加以合併，如 Figure 8(b)所示。由 Figure 8(b)可發現：節點 1 與節點(2,4)形成迴圈(loop)，因此合理堆放次序解必不存在。

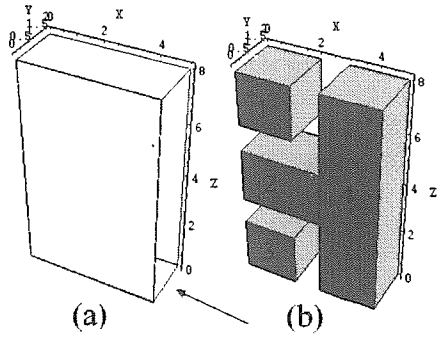

Figure 7 四個配對物件堆放程序

[表 1]

例	No. of block/pair	Loading order	CPU time (μsec)
Ex1	4/1	(1,2),4,3	3.3
Ex2	5/1	1,5,(4,3),2	5.5
Ex3	5/2	(1,5),(4,3),2	5.5
Ex4	6/0	2,4,3,1,5,6	6.6
Ex5	7/1	4,6,(5,1),2,3,7	8.8
Ex6	10/5	(2,1),(3,6),(4,5),(7,9),(10,8)	18.5
Ex7	15/6	(11,12),(14,15),2,13,(1,3),(6,4),(5,7),9,(10,8)	34.6
Ex8	45/0	11,12,14,15,2,13,1,3,6,4,5,7,9,10,8,26,27,29,30,17,28,16,18,21,19,20, 22,24,25,23,41,42,44,45,32,43,31,33,36,34,35,37,39,40,38	286

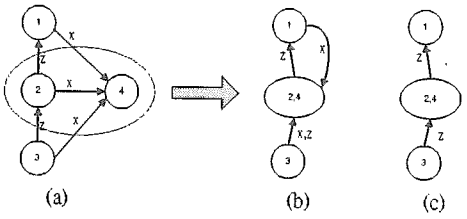

Figure 8 配對物件堆放影響限制圖

將 Figure 7(b)的物件置入 Figure 7(a)的單門貨櫃中，由 Figure 8 堆放影響限制圖可知必無合理堆放次序解。但若 Figure 7(a)爲前後皆有出入口(X,-X 方向)的雙門貨櫃，則合理堆放解卻存在，理由是影響限制圖中不會有因 X 軸限制所形成的邊了，形成如 Figure 8(c)，而可得到一合理堆放次序解：3,(2,4),1。Figure 8(b)及 Figure 8(c)是由原 LCG 中將原爲配對物件之兩節點 2，4 加以合併(merge)所得之圖，我們稱之爲合併堆放影響限制圖(Merged Loading Constraint Graph,MLCG)。

Algorithm Loading

Input: Blocks Information and user-specified constraints
Output: Loading sequence
Step1: Compute the coordinates of blocks by SA[16];
Step2: Construct loading constraint graph (LCG);
Step3: Construct Merged LCG (MLCG) by merging vertices of paired component;
Step4: **if** MLCG contains loop(s) **then** output"no feasible solution" and **stop**;
Step5 : Apply Topological sorting algorithm to MLCG to get the loading sequence;
Step6 : Output loading sequence;

4 實驗結果

在[16]中之緊縮程序，實驗顯示大都獲得最佳體積使用率。將此緊縮結果各物件之座標輸入，我們均迅速獲得各物件堆放次序。我們的實驗結果如表 1 所列，第二欄表示物件個數及配對物件的對數。第三欄表示所得到的 loading order，第四欄則爲所需的 CPU time。

Step 1 爲模擬退火演算法，在[16]中已證明非常有效，而 Step 2 至 Step 5 爲獲得物件座標後之排序處理，各步驟之複雜度均爲 O(n)，n 爲物件個數，以 Pentium-166,Windows95 爲測試環境，因在 Windows95 下，CPU tim 準確到 1/100 秒，爲實際量測所需時間，我們共執行 100000 次，再求其平均值。由實驗結果可知，本論文所提之方法有頗佳的效率。

5 結論

本論文中，我們提出了一新的三維物件堆放緊縮之問題及其緊縮演算法並解決堆放程序中的物件堆放次序問題。本文討論了矩形體、T 形體及 L 形體。不同於傳統方法，本方法先以模擬退火法求得各物件放置最佳位置，再考慮貨櫃或托板及重力的影響，求得堆放影響限制圖，再經由拓樸排序法求得各物件堆放次序。實驗證明，本方法非常有效，且能與各種自動化裝置，如機械手臂等結合。

Reference:

[1] M. ADAMOWICZ and A. ALBANO,"A Solution of the Rectangular Cutting-Stock Problem", IEEE Trans. on Systems, Man and bernetics, *SMC-6*, No. 4,April 1976.

[2] N. CHRISTOFIDES and C. WHITLOCK, "A Algorithm for Tw -Dimensional Cutting Problems", *Operations Research*, Vol. 25, No 1,1977.

[3] P.C. GILMORE and R.E. GOMORY, "A Linear Programming Approach to the Cutting-Stock problem", *Operations Research*, Vol. 9.pp. 849-859,1961.

[4] P.C. GILMORE and R.E. GOMORY,"A Linea Programming Approach to the Cutting-Stock problem, Part II " *Operations Research*, Vol. 11.pp. 863-888,1963.

[5] T.J. HODGSON,"A Combined Approach to the Pallet loading Problem", *IIE Trans.*,14,pp.175-182,1982.
[6] T.J. HADGSON, D.S. HUGHES and L.A.MARTIN-VEGA,"A Note on a Combined Approach to the Pallet loadin g Problem", *IIE Trans.*, 15,pp.268-271, 1983.
[7] T.M.HSIEH, H.W. LEONG and C.L. LIU, "Two-Dimensional Layout Compaction by Simulated Annealing", *proc. of IEEE Intl. Symp. on Circuits and Systems*, 1988.
[8] M.Y. HSUEH, "Symbolic Layout and Compaction of Integrated Circuits", Ph.D thesis, *UCB/ERL Report*, M79/80 U.C. Berkeley 1979.
[9] S.KIRKPATRICK, C.D. GELATT, Jr. and M.P. VECCHI, " Optimzation by Simulated Annealing ", *Science*, Vol. 220, pp.671-680,1983.
[10] Y.Z. LIAO and C.K. WANG, "An Algorithm to Compact a VLSI Symbolic Layout with Mixed Constraints", *IEEE trans. on CAD of Circuits and Systems*, No. 2 , pp.62-69,1983.
[11] D.A. MLYNSKI and C.H. SUNG, "Layout Compaction", *Advanced in CAD for VLSI*, Vol. 4,Layout Design and Verification (T. Ohtsukied), North Holland,pp.199-253, 1986.
[12] R.A. PENINGTON and J.M.A. TARCHOCO, "Robotic Palletization of Multiple Box Sizes", *Int. J. Prod. Res.*,1988, Vol. 26 ,No. 1,pp.95-105.
[13] F.M. PULS and TANCHOCO, "Robotic Implementation of Pallet Loading Patterns", *International of Production Research*, 24,pp. 635-546, 1986.
[14] M. SCHLAG, Y.Z. LIAO and C.K. WONG, "An Algorithm For Optimal Two-Dimensional Compaction of VLSI layouts", *Integration*, 1983, pp. 179-209.
[15] R.D. TSAI, E.M. MALSTROM and H.D. MEEKS," Two-dimensional Palletizing Procedure for Warehouse Loading Operations", *IIE Trans.*, Vol. 20, No. 4, 1988.
[16] 謝財明, 張璁,"The Study of 3-D Block Packing Problems", *The CHUNG YUAN JOURNAL* ,vol XXI, December, 1992