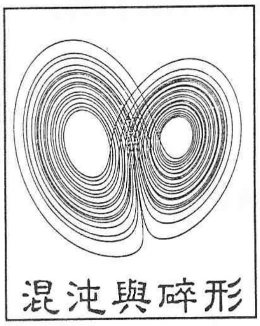


利用編號處理碎形的演算法

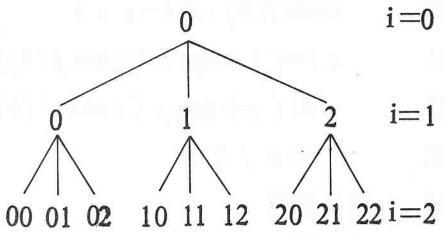
翁義聰

一、前言

一個函數或程序（ procedure ）不論是直接或間接呼叫它們自己，皆稱之為遞迴（ recursive)，使用遞迴描述的演算法簡潔易懂；我們希望一問題的演算法的時間繁度是低次方的，而且是有平行（ parallel) 解 。全教授於「迷你的殘形（**附註**）程式」〔 1 〕亦以遞迴處理；本文用編碼（ code ）先將具連續性（ sequential)的問題分割成獨立的子問題，若 n 為最終描點數，則每個子問題需時 O（ $\log n$ ）。第三段運用編碼及對稱的觀點給碎形一個更快速的演算法 。

二、用編碼分割連續性的問題

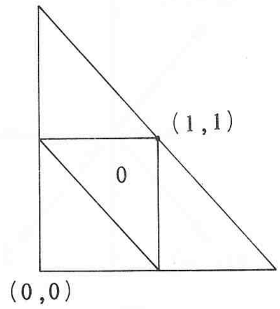
設以樹表示畫碎形的遞迴情形，若樹的深度為 i 時，在螢幕上顯示恰為一點，則此時顯示的點為 $n = 3^i$ 個 。



圖一

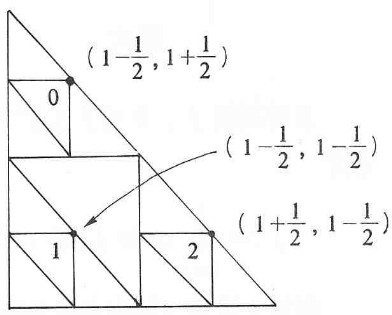
……

1.當 $i = 0$ 時，畫 3^0 個股長為 1 的直角三角形，此三角形的直角頂點坐標為：



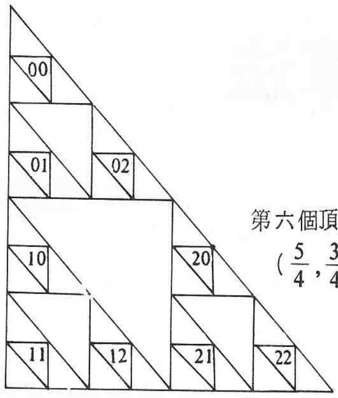
圖二

2.當 $i = 1$ 時，畫 3^1 個股長為 $\frac{1}{2}$ 的直角三角形，依序以 3 進位數 0 、 1 、 2 表示，直角頂點坐標為：



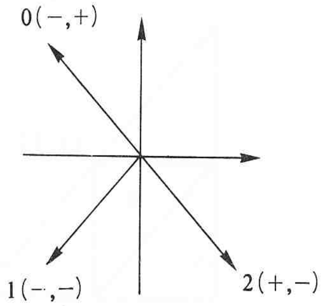
圖三

3.當 $i=2$ 時，畫 3^2 個股長為 $(\frac{1}{2})^2$ 的直角三角形，依序以 3 進位數表示：00、01、02、10、11、12、20、21、22。



圖四

至此，我們查覺三角形的順次為：



圖五

故定義函數 sgn x 與 sgn y 如下：

	sgn x	sgn y
0	-1	$+1$
1	-1	-1
2	$+1$	-1

圖六

且將編號 j，$0 \leq j \leq 3^i-1$，以 3 進位表示

$$j := \sum_{k=1}^{i} \text{code } j\,[k] \cdot 3^{k-1}$$

則第 i 層第 j 個直角三角形 bycode (i,j) 的直角頂點坐標為：

$$x := 1 + \sum_{k=1}^{i} \text{sgn } x\,[\text{code } j\,[k]] \cdot (\frac{1}{2})^{i-k+1}$$

$$y := 1 + \sum_{k=1}^{i} \text{sgn } y\,[\text{code } j\,[k]] \cdot (\frac{1}{2})^{i-k+1}$$

例：當 $i=2$，$j=6$（如圖四），則 bycode（2，6）的直角頂點的 x 坐標為：

解：$j=6_{10}=20_3$

code $j\,[1]=0$，code $j\,[2]=2$

sgn x [code j[1]] = sgn x [0] = -1

sgn x [code j [2]] = sgn x [2] = $+1$

$$x = 1 + (-1)\cdot(\frac{1}{2})^{2-1+1} + (+1)\cdot(\frac{1}{2})^{2-2+1}$$

$$= 1 - \frac{1}{4} + \frac{1}{2} = \frac{5}{4}$$

以下給一個用編碼技巧的 bycode(i,j) 程序：

1. 開始程序
2. 設定好 sgn x 及 sgn y 函數
3. 輸入 bycode（i，j）中的 i 與 j
4. 若 $i=0$，則畫一個頂點坐標為（1，1），兩股長為 1 的直角三角形
5. 否則做下列工作：
6. 開始
7. $x=0$，$y=0$，$d=1$
8. 執行 $k:=1$ 到 i 的迴圈
9. $q:=\text{Int}\ \lfloor j/3 \rfloor$
10. code$j[k]:=j-q*3$
11. $x:=(x+\text{sgn } x\,[\text{code } j\,[k]])/2$
12. $y:=(y+\text{sgn } y\,[\text{code } j\,[k]])/2$
13. $d:=d/2$
14. 結束迴圈
15. 畫一個頂點為（$1+x$，$1+y$），兩股

長爲 d 的直角三角形

16. 結束否則

17. 結束程序 bycode（i，j）

至此，我們以編碼技巧將這原本是連續性的問題，分割成 n 個獨立的子問題；執行的時間主要是 8～14 行迴圈， 子問題的時間繁度爲 O（$\log_3 n$）。

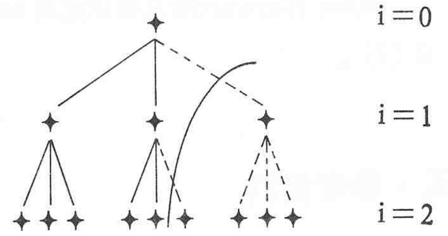
三、更快速的碎形程式

全教授於〔1〕文末加上「你能寫個更短的程式來畫殘形嗎？」這句話是很誘惑的；本文雖不能達到更短，但以圖形的對稱性及編碼的技巧，我們只須執行圖一樹的左半邊及中心線的 $1+\log_3 n$ 個遞迴，而達到更快速的程式。

讓我們重新觀察圖二～四中的第 i 層第 j 個三角形；若它的直角頂點爲（x，y），兩股長爲 d，則樹中第 $i+1$ 層與它對應的是頂點坐標（$x-\frac{d}{2}$，$y+\frac{d}{2}$）、（$x-\frac{d}{2}$，$y-\frac{d}{2}$）、（$x+\frac{d}{2}$，$y-\frac{d}{2}$），兩股長爲 $\frac{d}{2}$ 的三個直角形。現在以 tri（x，y，d）程序敍述如下：

1. 開始程序
2. 輸入 $tri(x,y,d)$ 中的 x、y、d
3. 若 $d\leq 1$，則描繪點（x，y）
4. 否則做下列工作：
5. 　開始
6. 　$d:=d/2$
7. 　tri（$x-d$，$y+d$，d）
8. 　tri（$x-d$，$y-d$，d）
9. 　tri（$x+d$，$y-d$，d）
10. 結束否則
11. 結束程序 tri（x，y，d）

若程序 tri 第 9 行所得的 $x+d>y-d$ 時略去，而第 3 行改爲描繪點（x，y）與（y，x），則每層呼叫 tri 程序的次數分別爲 1、2、5、…、$\frac{3^i+1}{2}$（如圖七）。



圖七

………

緊接著，我們附一個以 Turbo Pascal 4.0 版語言所寫的程式，它所完成的圖形爲兩股長 256 的等腰直角三角形的碎形

```
program sierpinski5;
uses crt, graph;
var
     gd, gm: integer;
     x, y, d: integer;

procedure opengraph;
begin
   gd := detect;
   initgraph(gd,gm,'');
   if graphresult <> grok then halt(1);
end;  {end procedure opengraph}

procedure tri(x,y,d:integer);
begin
   if d<2 then
     begin
       putpixel(x,y,1); putpixel(y,x,1);
     end
   else
     begin
       d := d shr 1;
       tri(x-d,y+d,d);
       tri(x-d,y-d,d);
       if x+d <= y-d then tri(x+d,y-d,d);
     end;
end;  {end procedure tri}

begin  {main program}
   opengraph;
   tri(128,128,128);
   readln;
   closegraph;
end.
```

四、結論

前段所附的程式，若在螢幕上描繪 $n=3^i$ 個亮點，則呼叫 tri 程序的次數爲 $\frac{3}{4}n+\frac{1}{2}\log_3 n+\frac{1}{4}$，遠少於〔1〕的 $\frac{3}{2}n-\frac{1}{2}$；而且程

序中所使用的加減乘除亦很精簡。

Ryan Hayward曾以編碼處理heap的建立〔3〕。

五、參考資料

1.全任重，迷你的殘形程式，「數學傳播」第14卷第4期（1990），pp 46-47。

2. A.V.Aho, J.E.Hopcroft, and J.D. Ullman, *The design and analysis of computer algorithms*, Addison Wesley, 1974.

3.Ryan Hayward, Average case anslysis of heap building by repeated insertion, *J. of algorithms* 12(1991), pp.126-153.

4.Zvi Galil, Optimal parallel algorithms, *Proc. of the inter. workshop on parallel* compu.(1984), pp.3-12.

附註：碎形（fractal）又譯作殘形。

——本文作者任教於私立崑山工專——