動態規畫的推廣

高中組數學科第二名

臺灣省立新店高級中學

作　　者：應承諺
指導教師：黃敏榮

一、研究動機

我對動態規畫(Dynamic Programming)的認識起於奧林匹亞研習營，有一次的專題便是動態規畫，其主要的內容是說在解決某些問題時，要求的答案其值決定於前面的值，而「前面」的值又決定於更前面的值。於是須要從前面一步一步的求值。

舉一個簡單的例：費氏數列在數學上定義成

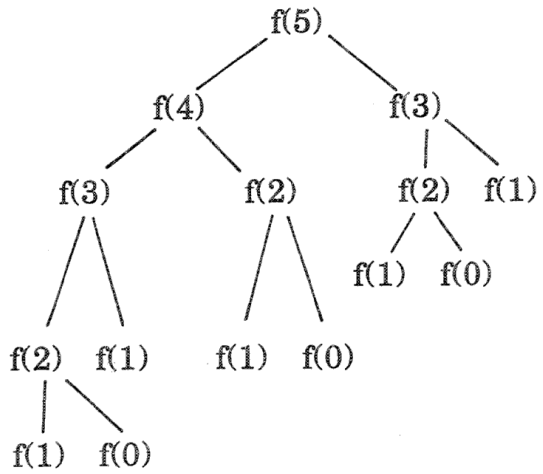
$f(0)=1$

$f(1)=1$

$f(x)=f(x-1)+f(x-2)$

如果需要寫一個程式來計算的話，直覺的方法便是依照原本的定義，用遞迴來解決：

```
int=f(intx)
{
  if (x<2)return 1;// ⇐所謂的遞迴終止狀態
  return f(x-1)+f(x-2);
}
```



若要計算 f(5) 的話，其過程可由右圖來表示。f(5) 的值取決於 f(4) 及 f(3) ，f(4) 的值取決於 f(3) 及 f(2) ，這裡我們可以發現到，f(3) 在 f(5) 及 f(4) 中分別計算了一次，這便是浪費。再看得更詳細一點，f(1) 計算了 5 次，f(0) 計算了 3 次，如此算下來，其增長的速度是很可怕的。

事實上，既然 f(x) 的值取決於 f(x-1) ，f(x-2) 而 f(x-1) 的值又取決於 f(x-2) ，f(x-3) ，那麼，我們便可由 f(0) 、f(1) 、f(2) 、f(3) …算到 f(x) ，這樣的話，所花的時間就變成線性的了。

這時候，程式中可放置一陣列，以儲存所得到的經驗，例如：

```
int f(x)
{
    int dp[max-x];
    int, i;
    dp[0]=1;
    dp[1]=1;
    for(i=2; i<=x; i++)
    {
        dp[i]=dp[i-1]+dp[i-2];// 不必再遞迴了
    }
    retum dp[x];
}
```

此外，動態規畫還有很多極爲有趣的應用，因而，我展開了我的研究。

二、研究目的

更深入地瞭解動態規畫，並試圖將之更廣泛地應用在各種問題上，例如迷宮問題、杯子問題（後述）、智慧拼盤等等，都是極爲有趣的應用。

三、研究設備及器材

㈠電腦。

㈡ C++ 編譯器。

四、研究過程

研究的過程，大多數的步驟是，想想那些以前紋盡腦汁都解不出來的問題是否可以用動態規化解決？想到問題之後再用動態規畫的方法不斷的嘗試，然後寫

出計畫表，再應用在程式上。依照這種方法，我找到了這一些從前覺得很不可思議，如今都能夠解決的問題：

㈠郵票問題。

㈡最長遞增子序列。

㈢迷宮最短路徑問題。

㈣柱子問題。

㈤空間跳躍問題。

㈥智慧拼盤的最短解。

五、研究結果

㈠郵票問題

所謂郵票問題，就是給你固定種數的郵票，來湊出某一值，要求「最少張數」的湊法。假設現在有 4 張郵票，分別爲 1,3,6,7 。則定義 R(x) 爲湊成 x 元所需要的最少張數。則顯然

$R(1)=1$

$R(3)=1$

$R(6)=1$

$R(7)=1$

上面 R(3)=1 的意思就是說要湊成 3 元最少張數就是 1 （拿「 1 」張 3 元的郵票）。那麼對於任意的 x ，我們可以先拿一張 1 ，則

$$R(x)=R(x-1)+1$$

或者先拿一張 3 ，則

$$R(x)=R(x-3)+1$$

或者先拿一張 6 ，則

$$R(x)=R(x-6)+1$$

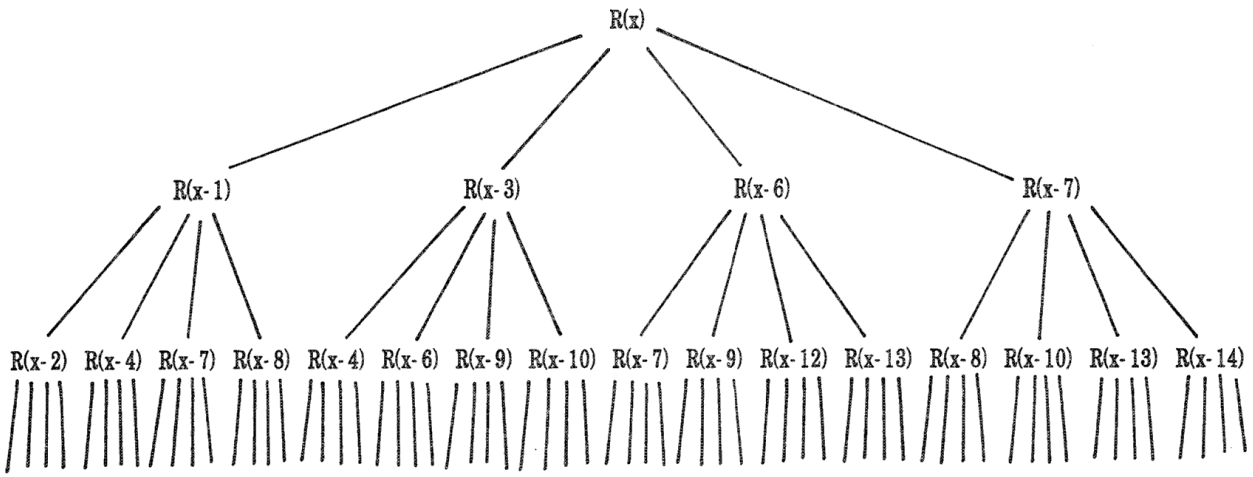
亦或者先拿一張 7 ，則

$$R(x)=R(x-7)+1$$

說了那麼多的「或」，那麼到底是先拿那一張呢，應該拿那個能夠使 R(x) 成爲最小的那個，則

$$R(x)=\text{Min}\begin{Bmatrix} R(x-1)+1 \\ R(x-3)+1 \\ R(x-6)+1 \\ R(x-7)+1 \end{Bmatrix}$$

於是，若是要求 R(x) 的值，需要先求 R(x-1) 、 R(x-3) 、 R(x-6) 、 R(x-7) 故其依存性可以下圖表示：



從上面，可以看出若是用遞迴解的話，則平均每一層皆有 4 個分支，有 n 層就有 4^n 個 R 值要算。但是仔細觀察，可以發現並不需要眞的從 R(x) 算回來，因爲這樣會產生許多重覆，例如，上圖中的 R(x-6) 就重覆了兩次。

㈡最長遞增子序列

所謂「最長遞增子序列」，就是在由一些數所組成的數列中，找出一個由右往左逐漸遞增的子序列。其開頭隨意，結尾也隨意，中間可斷開，但一定要是包含元素最多的。

例如：數列

6,11,8,15,5,13,3,10,1,4,16,12,9,17,7,2,14

則最長遞增子序列長度爲 5 ，有 7 組解如下：

6,8,10,12,14
6,8,10,12,17
6,8,10,16,17
6,8,13,16,17
6,8,15,16,17
6,11,13,16,17
6,11,15,16,17

動態規畫來解：定義 P(i) 爲第 i+1 個元素 ($0 \leqslant i < n$ ， f(x) 代表以某個

在 P(x) 前面元素做開頭，但以 P(x) 做結尾的最長遞增子序列的長度，則

$$f(x)=\underset{0\le i<x}{Max}\{f(i)+1\}\forall p(i)<p(x)$$

假如根本沒有 p(i) 小於 p(x)，則設 f(x)=1（就是只包含 p(x)）長度爲〝1〞爲子序列）

而若是不用動態規畫而用 brute force 法的話，則對於每一個元素，而能選也可能不選，則其時間是與 2^n 成正比。

但動態規畫則是跟 n^2 成正比，當 n 很大時，差別就很明顯了。

n	2^n	n^2
2	4	4
5	32	25
20	1048576(0 至 20 秒)	400(瞬間)
50	1125899906842624(當機)	2500(1 至 2 秒)
100	1267650600228229401496703205376(當機)	10000(4 秒至 5 秒)

㈢迷宮最短路徑問題

設 g(x,y)爲在格子狀迷宮從起點走到（x,y）最短路徑長度，則若終點爲 (x_0,y_0)，則

$$g(x_0,y_0)=\text{Min}\left\{\begin{array}{l} g(x_0-1,y_0)+1 \\ g(x_0,y_0-1)+1 \\ g(x_0+1,y_0)+1 \\ g(x_0,y_0+1)+1 \end{array}\right\}$$

則同前，應從起點開始算值，方法爲：視爲很多人從起點起跑，每個人跑過的地方都會留下「腳印」，由於有「很多人」，是故碰到岔路時必定爲有人往每條路去。他們的速度完全一樣，且若發現前面有別人已留下的足跡，便停止。如此便可證明第一位跑到終點的人必定是走過最短路徑的人。此外，還可循其來時留下的足跡追蹤回去，便可得一解。

㈣杯子問題

此問題爲使用二個定容的容器，以及無限量的水，來湊成某一水量，例

如有 3 升和 5 升的容器要湊出 4 升的水，則可能有如下方法：將 5 升裝滿後倒進 3 升，由於 3 升裝不下 5 升的水，是故 5 升中還剩 2 升，將 3 升中的水倒掉，再倒進那位於 5 升中的 2 升，將空的 5 升裝滿水倒進 3 升，由於 3 升中早已有 2 升，是故只能再倒進 1 升，於是便得 4 升的水在 5 升的容器中。

將所有的狀態化爲一個狀態空間，空間中的每一位置代表不同的狀態，如上例，就是用一矩陣 $A_{3\times5}$ 代表，其元素爲 $a_{ij}(0 \leqslant i \leqslant 3, 0 \leqslant j \leqslant 5)$。則元素 a_{ij} 代表當 3 升中有 i 升，而 5 升中有 j 升的狀態。

將處理方式簡化，可得此問題十分類似迷宮問題，因有以下方法可以從 a_{ij} 走到 a_{ij} ：(1)將 3 升裝滿(2)將 3 升中的水倒掉(3)將 5 升裝滿(4)將 5 升的水倒掉(5)將 3 升的水倒進 5 升中(5)將 5 升的水倒進 3 升中。

於是可使用「多人賽跑」的機制解出一串步驟可完成目標。

㈤空間跳躍

所謂空間跳躍，B 指藉由限定的一些向量在一平面（或空間）中從某一點跳到另一點。如圖 A ，若有這三個向量，則 B 圖中爲由起點至終點可能的走法：

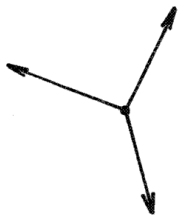


圖 A

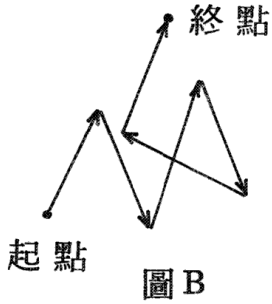


圖 B

爲了方便起見，我們限定平面（或空間）皆由整數點構成，向量的各分量亦爲整數。於是此問題便可依照「多人賽跑」的方法利用限定的一些向量從起點到終點找到一最短路徑。

㈥智慧拼盤的最短解

智慧拼盤是一個益智遊戲，如圖，爲一 n × m 的盒中裝 n × m-1 個棋子，利用這剩餘的一個空格，可將盒中的棋子移動，然後可以到達另一盤面（如右下圖）。

3	2	6
1	4	7
8	5	

由於電腦的計算速度及計憶體的考量，動態規畫只能夠討論 3 × 3 的智慧拼盤。

1	2	3
4	5	6
7	8	

3 × 3 的智慧拼盤由 8 個棋子一個空格構成，將其排列組合，總並有 9!=362880 種不同的狀況，

於是可定義 Pe 爲終點狀態 K(P) 爲 P 狀態的最短路徑的步驟數，則：

$$K(Pe)=Min\left\{\begin{array}{l}K\ (\ Pe\ 空格往上\ )\ +1\\K\ (\ Pe\ 空格往下\ )\ +1\\K\ (\ Pe\ 空格往左\ )\ +1\\K\ (\ Pe\ 空格往右\ )\ +1\end{array}\right\}$$

於是便發現應從起點做「多人賽跑」，找到其最短路徑。

六、結　論

㈠利用動態規畫，可以解決很多困難而不易思考的問題。

㈡智慧拼盤其實是無解的狀況的，因用空格移動的方式，經觀察並無法走完所有的狀態。

七、參考資料

㈠人工智慧與專家系統（張勳）。

㈡演算法與資料結構（郭台生）。

㈢Fundamentals of Data Structures in C(Ellis Horowitz, Sartaj Sahni, Susan Anderson Freed)。

評　語

動態規劃是遞迴運算中節省大量計算量的一個方法，作者將之應用在數個不同的問題中，所得的結果十分有趣。

尋找新的問題及將新（或舊）的方法運用至新的問題而能提供解決的途徑，是數學中一個極爲重要的課題，作者大膽的用計算機嘗試解決及創造一些新的題目，精神十分值得鼓勵，能用自己設計的程式來完成一些有趣的問題並將之清楚，有條理的顯示在計算機銀幕上，可見作者的努力及工夫。

作者的獨立性及原創性令人讚賞。