舊的塗色問題，新的計算方法

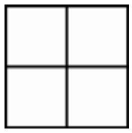
◎羅驥韡／台北市立陽明高中

塗色問題

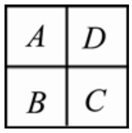
在高中「排列組合」的課程範圍中，有一類常出現的問題就是「塗色問題」。

例如：

在右圖中，如果我們利用 5 種不同的顏色來塗，然後規定「相鄰的區域不能塗同樣的顏色」，那麼總共有幾種不同的塗法？



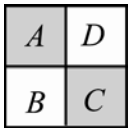
一般，如果我們用傳統的方式來分析此題，可能會這樣解題：首先，先將四個區域分成 $ABCD$ 四區（如圖）。再來，因為 A 區與 C 區不相鄰，所以可能同色也可能不同色，因此，我們分成兩個狀況來討論：



- A 與 C 同色：

  假設我們從 A、C 開始塗色，因為我們有 5 種不同的顏色可以塗，所以剛開始有 5 種選擇。再來塗 B，因為 AC 已經用掉一種顏色，所以 B 有 4 種顏色可以選。最後塗 D，因為 D 只要與 AC 不同即可，所以 D 還有 4 種顏色可以塗。所以總結，A 與 C 同色這種情況共有：

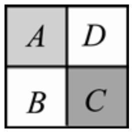
  $5\times4\times4=80$ 種不同的塗法。



- A 與 C 不同色：

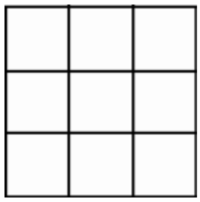
  如果還是從 A 開始，那麼 A 有 5 種顏色可以選。C 因為與 A 不同，所以只有 4 種選擇。B 又分別與 A、C 相鄰，所以選擇數只剩下 3 種。同樣道理，D 也只有 3 種顏色可以選。因此，這種情況共有：

  $5\times4\times3\times3=180$ 種不同的塗法。



綜合以上兩種情況，這個塗色問題共有：$80+180=260$ 種不同的塗法。

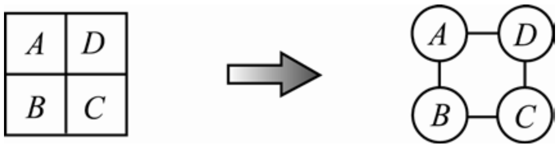
這樣的分析方法固然沒什麼不對，但是如果原來的問題是要將 5 種不同的顏色塗在如右圖的圖中呢？那麼我們又要分成多少種不同的狀況來討論呢？或者，如果我們的圖形又遠比右圖還複雜呢？這時，這樣的分析方法難道不會陷入雜亂無章的窘境嗎？



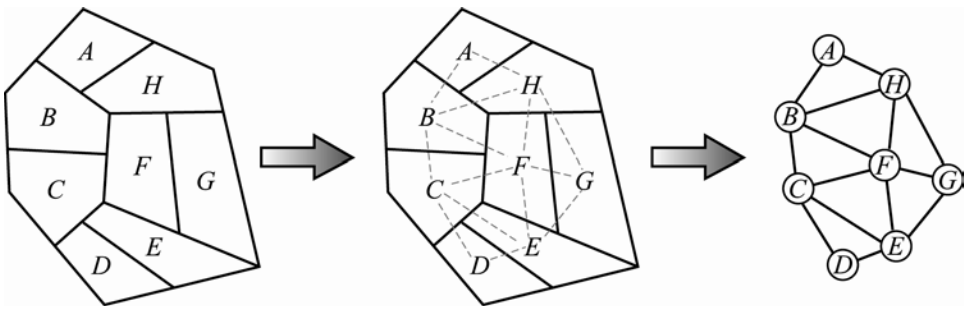
因此，在這裡我要介紹一種分析方法，不但可以對付任何種類的圖形，我們甚至可以設計出電腦可以計算的程式，讓電腦可以直接算出答案來，省下我們不少的手算時間。

新的思考模式

如下圖，我們可以將原來的圖變成另一個圖。首先，不同的區域，我們用一個圈圈來代表；其次，相鄰的區域，**我們用一條線連起來，表示它們彼此相鄰**。利用這樣新的表示方法，我們就可以將原來的圖轉換成新的圖。

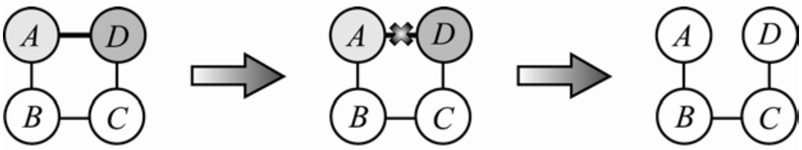


下圖再舉一個例子。



接下來，我們來討論這樣的轉變，可以帶來什麼樣的新算法。

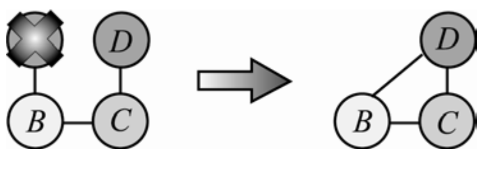
前面看到，相鄰的區域顏色不能相同，所以換句話說，有線連在一起的區域，顏色不能相同。但現在請考慮以下的想法：假設我們有 5 種顏色可以用，然後要計算下面「圖一」有幾種不同的塗法。如果我們將「圖二」中的某個連接線去掉，變成「圖三」的樣子，那麼「圖一」與「圖三」的塗法總數有什麼差別？



<圖一>　　<圖二>　　<圖三>

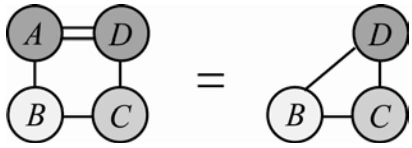
「圖一」中，A 與 D 相連，所以顏色不同；「圖三」中，我們將 A 與 D 中間的連接線去掉，因此表示它們「不相鄰」，所以顏色可以相同（當然也可以不相同）。綜合以上的觀察，我們知道「圖三」的塗法總數會比「圖一」還多，因為「圖三」的 A 與 D 顏色可以一樣，而這些塗法（A 與 D 一樣）正是「圖三」比「圖一」多出來的東西，所以我們要扣掉它們。現在我們來好好研究一下這些「多出來的東西」。

當 A 與 D 顏色相同時，我們可以讓 A 與 D「合體」，只留下 D，然後將**原來與 A 相連的連接線轉移給 D**（請看右圖）。這樣產生的新圖形，它的塗色方法就會跟「A 與 D 同色」時的塗色方法一樣多。



為什麼會這樣呢？假設我們在還沒刪掉 A 之前，塗了某些顏色到 A、B、C、D 四區（其中 A、D 同色），並且符合「有連接線的兩區不同色」的原則。現在，如果我們刪掉 A，然後將**原來與 A 相連的連接線轉移給 D**，這樣一來，本來與 A 不同色的區域就必須與 D 不同色，但這樣的作法並不會造成任何矛盾，因為 A 與 D 同色，所以與 A 不同色的區域自然就會與 D 不同色，因此原來的塗法也可以用在新的圖形上面，完全不會造成任何問題。另一方面，讓我們反過來看，如果我們塗了某些顏色到新的圖形上面，同樣也是在符合「有連接線的兩區不同色」的原則下，那麼我們會發現，這樣的塗法也可以用在原來的圖形上（其中 A、D 同色），同樣不會產生任何問題。

綜合以上的討論，我們得到一個重要的結果：「A、D 同色」的塗法剛好等於「新圖形」的塗法。我們用右圖來表達這樣的現象：其中，A 與 D 之間，我們用一個「雙鍵」來代表「A、D 同色」；兩圖之間，我們放了一個「等號」，表示兩圖的塗色方法數是一樣的。



牛刀小試

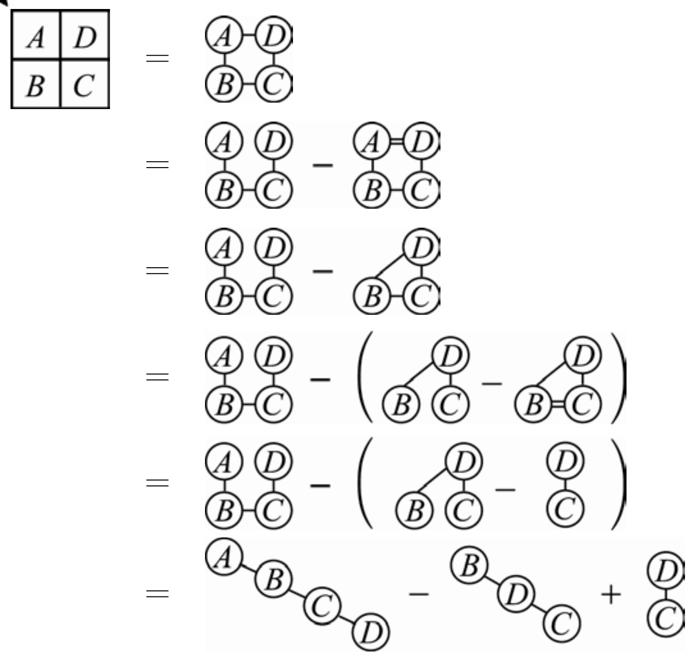
好了！介紹完一些我們所要使用的小發明，讓我們開始來計算原來的問題：

例題

1. 利用 m 種不同的顏色來塗右圖，相鄰不同色，則有幾種塗法？

A	D
B	C

解



將原來的圖轉換為新的圖

去掉 A、D 間的連線，並扣除多出來的塗法

當 A、D 同色時，可刪掉 A，並將 A 的連線轉移給 D

去掉 B、C 間的連線，並扣除多出來的塗法

當 B、C 同色時，可刪掉 B，並將 B 的連線轉移給 C

去除括號

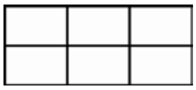
$$= m(m-1)^3 - m(m-1)^2 + m(m-1)$$

$$= (m-1)^4 + (m-1)$$

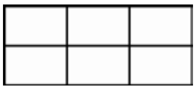
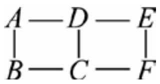
當 $m=5$ 時，$(m-1)^4+(m-1)=4^4+4=256+4=260$，而這個答案我們在前面也曾經利用傳統的方式計算過一次。

例題

2. 利用 m 種不同的顏色來塗右圖，相鄰不同色，則有幾種塗法？



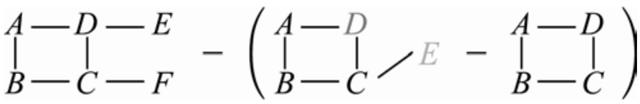
解

 $=$ 

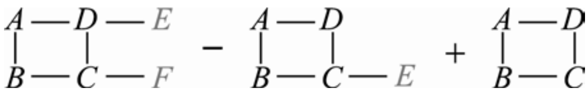
換成新圖形

$=$ 

打斷 E、F 連線，並合併 E、F，然後扣除之

$=$ 

打斷 D、E 連線，並合併 D、E，然後扣除之

$=$ 

去掉括號

$$=\left[(m-1)^4+(m-1)\right]\left[(m-1)^2-(m-1)+1\right]$$

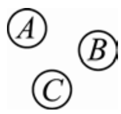
至於為什麼最後可以這樣計算，請繼續看後面的公式推導

公式推導

注意　在以下的討論中，我們都假設我們**有 m 種顏色**可用。

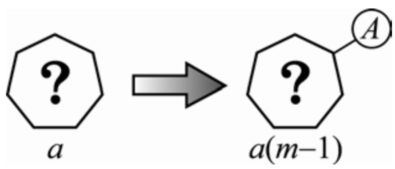
定理 **1.**

如果每個區域都**各自獨立**，彼此並不相鄰，就像右圖一般，那麼每個區域都有 m 色可用，所以如果有 3 個獨立區域，就會有 m^3 種塗色方法。一般而言，如果有 k 個獨立區域，就有 m^k 種塗色方法。



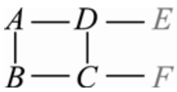
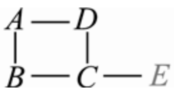
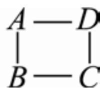
定理 **2.**

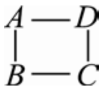
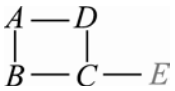
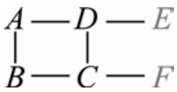
假設某個圖形原來有 a 種塗色的方法。如果在這個圖形外面再連接一個區域（如右圖），那麼新形成的圖形有 $a(m-1)$ 種塗色方法。



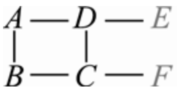
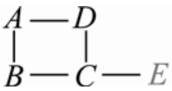
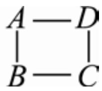
為什麼會這樣呢？因為圖中新增的區域 A 只要**跟它所連接的區域不同色**即可，因此區域 A 還有 $(m-1)$ 種顏色可以用，所以原有的圖形有 a 種塗色方法，再乘上區域 A 的 $(m-1)$ 種顏色，便會產生 $a(m-1)$ 種塗色方法。

到這裡，我們回頭來說明一下，之前我們是如何計算出下面的圖形的。

 −  + 

	在前面，我們已經算過左圖的塗色方法有 $(m-1)^4+(m-1)$ 種。
	現在這個圖長了一根毛（E），所以根據「定理 2」，塗色方法變成 $\left[(m-1)^4+(m-1)\right]\cdot(m-1)$ 種。
	這個圖又多長了一根毛，所以還是根據「定理 2」，塗色方法有 $\left[(m-1)^4+(m-1)\right]\cdot(m-1)\cdot(m-1)$ 種。

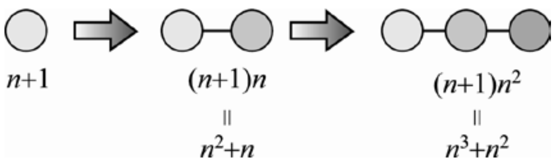
所以這也就是為什麼

 −  +  $= \left[(m-1)^4+(m-1)\right]\left[(m-1)^2-(m-1)+1\right]$

的道理了！

注意 從上面的許多計算中，可以清楚的看到 $(m-1)$ 這個數字一直出現，所以在後面的討論中，為了讓計算式看起來比較簡潔，我們在此設定：$n=m-1$。

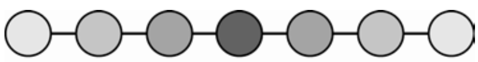
如果一個圖形長成類似下圖一樣的長鏈式的圖形，那麼它會有幾種塗法呢？



如果剛開始只有一個區域，當然我們會有 m 種顏色可以塗，或者寫成 $(n+1)$ 種。如果我們在這個區域後面接上另一個區域，那麼透過「定理 2」可知，原來的塗法總數必須再乘上 $(m-1)$，也就是再乘上 n，因此塗法總數會變成 $(n+1)n$。同樣的道理，如果我們繼續在後面又接一個區域，當然又要再乘上 n，因此塗法總數又會變成 $(n+1)n^2$。這個過程可以一直持續下去，所以我們將它整理成一個定理。

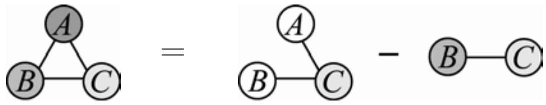
定理 3.（鏈狀圖形）

如果有 k 個區域（$k \ge 1$）串成一長鏈（如下圖），則它的塗法總數為 n^k+n^{k-1} 種。

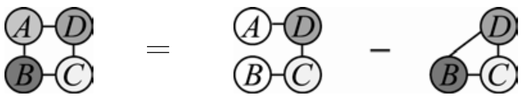


$(n+1)n^{k-1}=n^k+n^{k-1}$

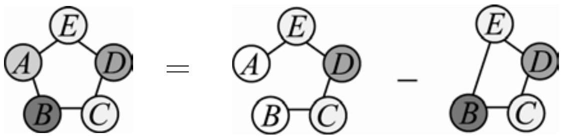
現在我們來考慮另一種圖形：環狀圖形。



我們將原來的圖形拆解成**兩個鏈狀**圖形，所以我們知道 3 個區域的環狀圖形可以利用 $(n^3+n^2)-(n^2+n^1)=n^3-n$ 來計算。



4 個區域的環狀圖形可以拆解成**一個鏈狀**，再扣掉**一個環狀**，所以塗法總數為：$(n^4+n^3)-(n^3-n)=n^4+n$



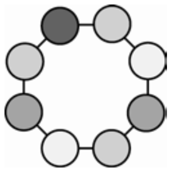
5 個區域的環狀圖形也是拆解成一個鏈狀，再扣掉一個環狀，所以塗法總數為：$(n^5+n^4)-(n^4+n)=n^5-n$

從以上的討論，我們可以整理出一個有關環狀圖形的算法：

定理 4.（環狀圖形）

如果 k 個區域（$k\geq 3$）圍成一個環狀，則塗法總數為

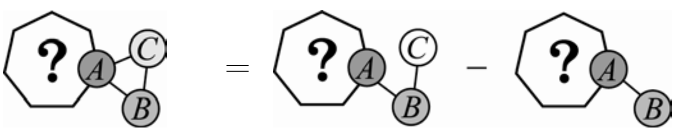
$$n^k+(-1)^k n$$



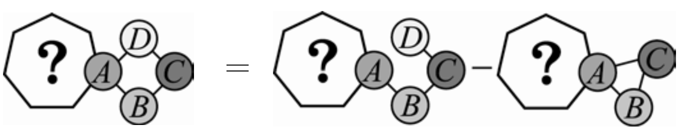
接下來，我們來討論一種新的情況：如果從原來的圖形「長出」一些新的結構，那麼塗色的方法又會如何改變。

注意 在以下的討論中，我們將**原來的圖形**用右圖的樣子來表示，並將它**原來的塗色方法數**設為 a。

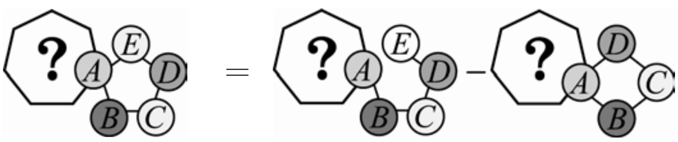
a



利用等號右邊的算法與「定理 2」來計算，得塗法數為：$an^2-an=an(n-1)$ 種



利用同樣的推理方式，可推得塗法有：$an^3-an(n-1)=an(n^2-n+1)$ 種



同理，這個環狀結構的塗法總共有：

$$an^4-an(n^2-n+1)$$
$$=an(n^3-n^2+n-1)$$

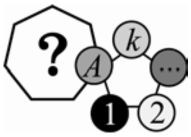
由於以上的觀察，我們可以歸納出一個公式。

定理 5.

假設一個圖形原有 a 種塗法。如果從這個圖形的某個節點 A 多**長出 k 個節點**並以**環狀**的方式連回 A 點，那麼這個新形成的圖形會有

$$an\left(n^{k-1}-n^{k-2}+n^{k-3}-\cdots+(-1)^{k-1}\right)$$

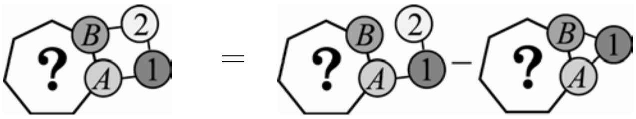
種塗色的方法。



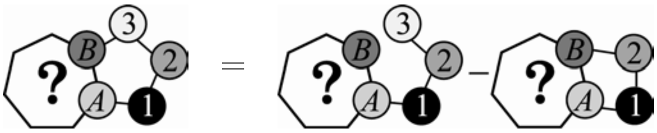
再來，我們來看看另一種「出芽」的情形。

假設我們從原來圖形中的某個節點長出一個環狀結構，但並不是連回同一個節點，而是**連回它隔壁的節點**，這時塗法總數又會如何變化呢？請看以下的分析。

把節點 B 和節點 1 拆開,並扣掉節點 B 和節點 1 合併的情形，可得：
$an-a=a(n-1)$ 種塗法



把節點 B 和節點 2 拆開,並扣掉節點 B 和節點 2 合併的情形，可得：
$an^2-a(n-1)=a(n^2-n+1)$ 種



同理，這個圖形的塗法有：
$an^3-a(n^2-n+1)=a(n^3-n^2+n-1)$

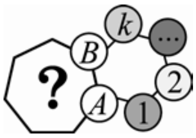
統整以上的經驗，我們可得以下的公式：

定理 6.

假設一個圖形原有 a 種塗法。如果從這個圖形的某個節點 A 多**長出 k 個節點**並以**環狀**的方式連回隔壁的 B 點，那麼這個新形成的圖形會有

$$a\left(n^{k}-n^{k-1}+n^{k-2}-\cdots+(-1)^{k}\right)$$

種塗色的方法。

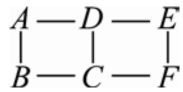


跟定理 5 一樣，這個公式也可以利用「等比級數」的公式來整理：

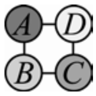
$$a\left(n^{k}-n^{k-1}+n^{k-2}-\cdots+(-1)^{k}\right)=a\cdot\frac{n^{k+1}-(-1)^{k+1}}{n+1}=\frac{a}{m}\left(n^{k+1}-(-1)^{k+1}\right)$$

好了！發展了一堆公式，現在是好好試試身手的時候了。

我們之前曾經花了很多力氣來計算右圖的塗法。現在讓我們來看看利用公式如何計算：

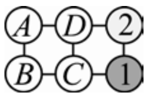


利用「定理 4」，我們知道：



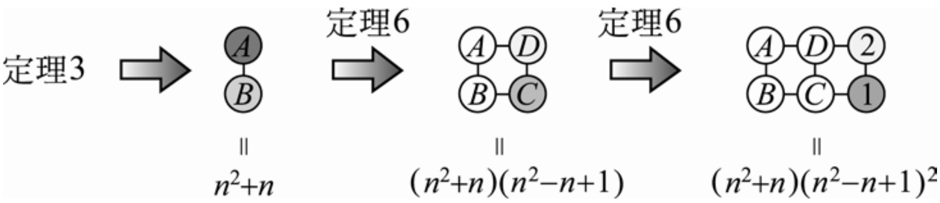
$$= \quad n^4 + (-1)^4 n \quad = \quad n^4 + n$$

假設 $a = n^4 + n$ ，利用「定理 6」，我們知道：



$$= \quad a\left(n^2 - n + 1\right) \quad = \quad \left(n^4 + n\right)\left(n^2 - n + 1\right)$$

當然，這個圖形我們也可以用下面的推理方式來計算：



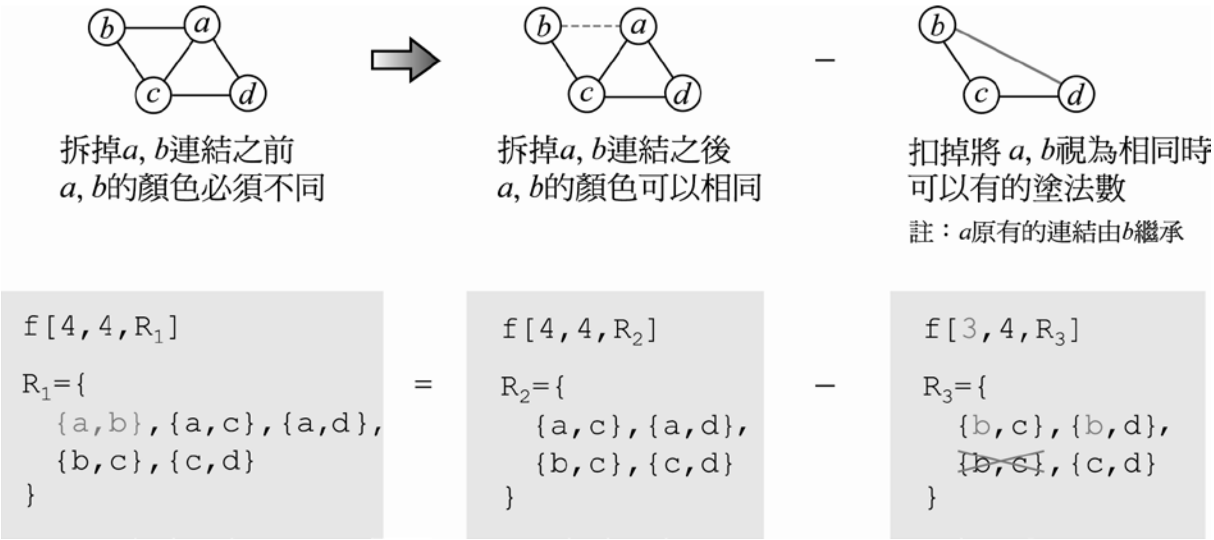
計算程式

有時，當圖形過於複雜，利用公式來計算還是太費時了，這時我們就不得不借助電腦的幫忙了。以下我們提供一個 Mathematica 的計算程式，來協助我們計算複雜的圖形：

程　　式　　碼	解　　說
`f[k_,m_,{}]:= m^k;`	k：代表圖中有個 k 節點。 m：代表有 m 種顏色可以用。 {}：空「集合」代表目前的節點都是獨立的，這也就是「定理 1」的情況。

`f[k_,m_,s_]:= Module[{s1,s2,node1,node2},`
 	
`node1 = s[[1]][[1]];`
`node2 = s[[1]][[2]];`
`s1 = Delete[s,1];`
`s2 = Union[s1/.node1->node2];`
 	
`f[k,m,s1]-f[k-1,m,s2]`
`];`
	s：表示節點之間的連結集合。
	node1：表示第一個連結的第一個節點。
	node2：表示第一個連結的第二個節點。
	s1：表示刪掉 s 的第一個連結後的集合。
	s2：表示 s1 中，如果有 node1 那就把它換成 node2。

這個程式的算法，其實說穿了，就是我們在「新的思考模式」那個段落所說明的計算方式。下圖進一步解說了此程式的執行方式：



下面我們來看一個實際的執行範例：

```
In[1]: s={{1,2},{1,4},{2,3},{2,5},{3,6},{4,5},{4,7},{5,6},{5,8},{6,9},{7,8},{8,9}};
In[2]: f[9,5,s]
Out[2]: 142820
```

右圖有 9 個節點，由電腦執行的結果得知，如果我們用 5 種顏色來塗，相鄰不同色，則會有 142820 種塗法。

1	2	3
4	5	6
7	8	9

透過這樣的分析方法，不管多麼複雜的圖形，不管用多少種顏色，我們都可以將所有的塗色方法數計算出來，甚至我們可以將這樣的算法寫成電腦程式，讓電腦幫我們計算所有複雜的過程，這樣一來，面對「塗色問題」，我們就可以完全高枕無憂了。