

中華民國第四十八屆中小學科學展覽會  
作品說明書

---

高中組 數學科

第二名

040421

相異電路組合數的探討

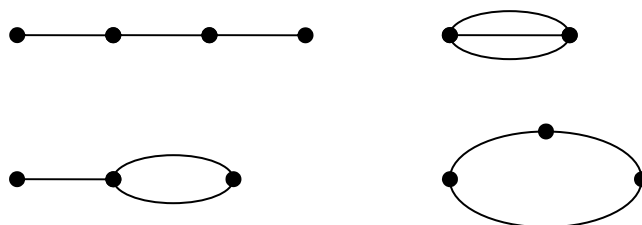
學校名稱：國立臺南第一高級中學

作者： 高二 吳哲仰 高二 林伯昱	指導老師： 蔡仲彬
-------------------------	--------------

關鍵詞： 排列組合、電路組合數、生成函數

## 壹、 摘要

有一個電路網絡，其中含有 N 個相同的電路元件，因為每個電路元件皆是相同的，所以會使得兩個電路網絡不相等的唯一原因，僅在於元件的排列和組合方式。一個電路元件有兩端點可以與其它的元件串聯或並聯。



圖：電路網絡的樣例，線段代表電路元件，點則是元件的接合端點

其中所串聯的數個元件，是可以改變串聯的次序是不影響該電路網絡的運作的，改變並聯次序時亦同。因此若兩個電路網絡 A, B，A 可以由調換串聯或並聯次序後跟 B 相等，則稱此兩個電路網絡是等價的；不等價的電路網絡，稱為相異的。



圖：三個等價的電路網絡（可以藉由改變串聯次序而相等）

本文最後給出了一個方法以求得：給 n 個相同的電路元件，可以造出幾種相異的電路網絡的個數。

最後推得：有 n 個電路元件的相異電路網絡個數，n=1 時為 1；當 n>1 時為  $2a_n$ ，其中數列  $\{a_n\}$  的遞迴關係為：

$$a_n = \frac{2 \sum_{n>k>0} \sum_{t|n-k} ta_t a_k + \sum_{t|n, t<n} ta_t - \sum_{t|n-1} ta_t}{2n}, a_1 = 1$$

## 貳、 研究動機

有  $N$  個相同的電路元件，可以組成幾種不同的網路呢？這很明顯是一個排列組合的問題，剛好與最近的課程相關，在查了一些資料後發現這個問題雖然已經被討論過，但卻沒有比較深入的討論，大部分的解法都是接近於直接枚舉所有的組合方法，或是經過簡單的分解後枚舉。這種方法對於  $N$  不算相當大時是可行的，但隨著  $N$  的成長，計算所花費的時間是呈指數增加的。因此我們把目前「排列與組合」一章的內容加以延伸，利用課餘的時間來研究。

## 參、 研究目的

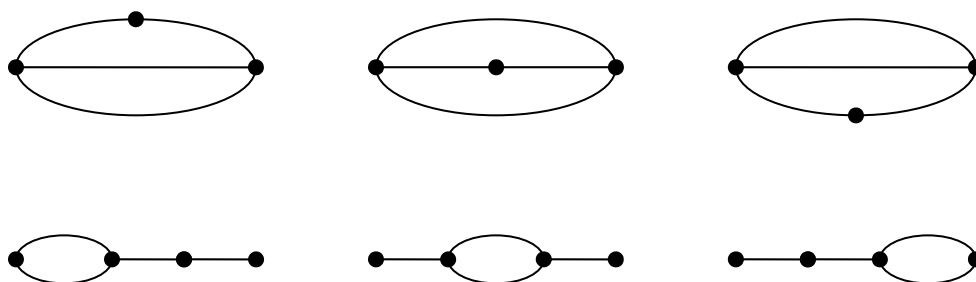
找出一個計算不等價電路網絡數的方法，從原始的枚舉所有的排列，改進並且重新設計一個計算方法。這個方法必須要在相對於  $N$  的增長之下，能在多項式的時間內得出。

## 肆、 研究設備及器材

紙、筆、電腦

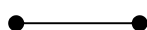
## 伍、 研究過程與方法

有  $N$  個相同的電路元件，可以將它們以端點相接的方式串聯或並聯起來，其中在數個被串聯的電路元件中，調換次序並不影響電路網絡的結構，並聯時也有類似的結果。下圖中上排的三个網絡是互相等價的，下排的三个亦然。如果兩個電路網絡是不等價的，則說它們是兩個相異的電路網絡。



我們將探討這樣一個問題：有  $n$  個電路元件，問可以組成幾種相異的電路網絡。一個很簡單的方法是枚舉所有的組合成電路網絡的方式，在  $n$  比較小時是可行的，底下分別為  $n=1,2,3,4$  時的例子：

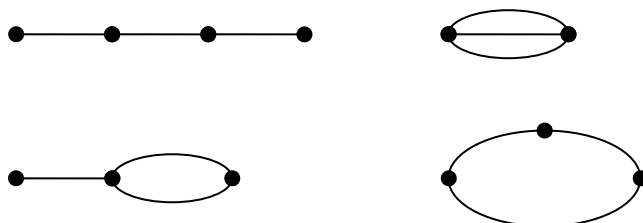
1.  $n=1$  : 1 種



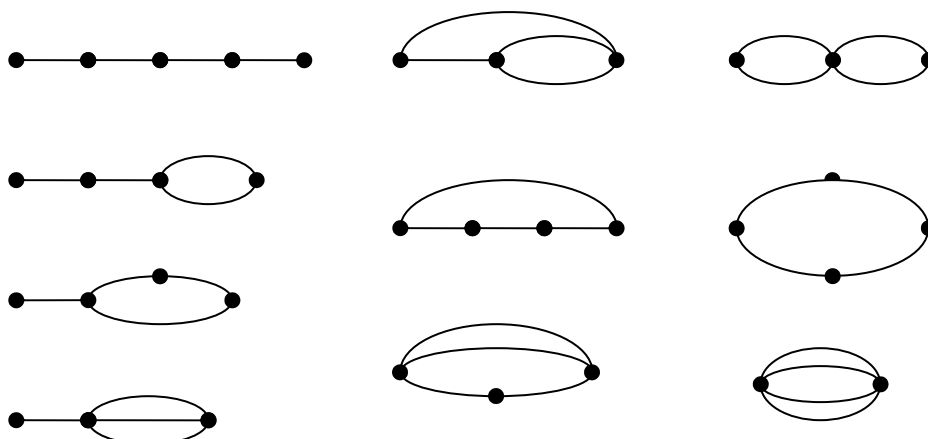
2.  $n=2$  : 2 種



3.  $n=3$  : 4 種



4.  $n=4$  : 10 種



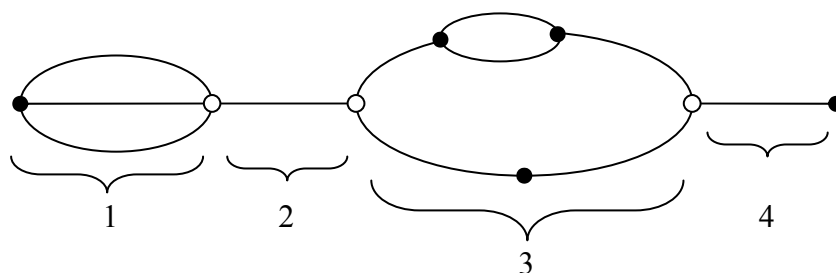
不過我們發現隨著電路元件數量  $n$  的成長，網絡數量成長得太快了<sup>1</sup>。使用枚舉的方式來處理這個問題將是不切實際的，因為枚舉所花的時間將會和答案大小成現正向的關係。進一步分析這個問題，我們並沒有要求找出所有的「解」，只是要求出「解的數量」，在枚舉的過程中還附加了多餘的「解」的訊息，這似乎是沒有必要的，因此我們猜測更好的計算方法是存在的。

為了將問題分解，我們先將電路網絡分類：串聯型網絡和並聯型網絡

定義電路網絡的節數：一個電路網絡的節數為，電路上歧接點的數量加一。

定義歧接點：一個電路網絡上的一個點若且唯若移除這個點後會使得該電路網絡不連通。

下圖中白色點為歧接點，此電路的節數為 4。

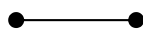


<sup>1</sup> 見附表一

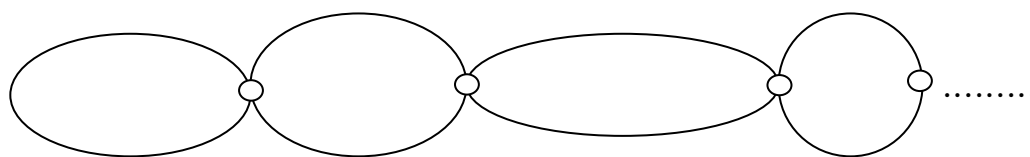
定義串聯型網絡為，電路的節數大於 1；並聯型網絡為，電路的節數等於 1，即只含單一個節。下圖中左者有三個節為串聯型，右者僅有一個節為並聯型。



其中當電路元件數等於一時是一個例外，此網絡可以視做串聯型亦可為並聯型。



注意到串聯型網絡中，每一個節都會是一個並聯網絡(圖一)，也就是說串聯型可以由數個並聯型的網絡串聯而成，且不同的串聯排列順序將被視為等價電路網絡。因此我們得到了一種構造串聯型網絡的方法：串聯型網絡的數量為使用並聯型網絡組成的組合數，且使用來組成該串聯型網絡的同一個並聯型網絡可以被多次取用。因此可以構造出的方法數為  $H_m^n = C_m^{n+m-1}$ 。又每個並聯型網絡組件中的電路元件數總和必須等於該串聯型網絡中的元件總數  $n$ 。



圖一：串聯型網絡中，每一個節都會是一個並聯網絡

設元件數有  $n$  個的串聯型網絡有  $a_n$  個，元件數有  $n$  個的並聯型網絡有  $b_n$  個，考慮從元件數為 1 的並聯型網絡中取出  $j_1$  個，方法數為  $C_{j_1}^{b_1+j_1-1}$ ，從元件數為 2 的並聯型網絡中取出  $j_2$  個，方法數為  $C_{j_2}^{b_2+j_2-1}$ ，以此類推。又合法的組合方法為所取的元件總數為  $n$ ，因此有  $j_1 + 2j_2 + 3j_3 + \dots = n$ ，故所有的組成方法數總和為

$$a_n = \sum_{j_1+2j_2+3j_3+\dots=n} \binom{b_1+j_1-1}{j_1} \binom{b_2+j_2-1}{j_2} \binom{b_3+j_3-1}{j_3} \dots \binom{b_{n-1}+j_{n-1}-1}{j_{n-1}}$$

再考慮並聯型網絡的組成方式，可以由數個串聯型網絡並聯而成(圖二)



圖二：由數個串聯型網絡並聯而成的並聯型網絡

其組合關係是與上述串聯型電路對偶的，最後可以推得相似的關係式

$$b_n = \sum_{j_1+2j_2+3j_3+\dots=n} \binom{a_1+j_1-1}{j_1} \binom{a_2+j_2-1}{j_2} \binom{a_3+j_3-1}{j_3} \dots \binom{a_{n-1}+j_{n-1}-1}{j_{n-1}}$$

因  $a_1 = b_1 = 1$  且  $a, b$  之間具有對稱的遞迴關係，故  $a_n = b_n$ ，改寫遞迴式為

$$a_n = \sum_{j_1+2j_2+3j_3+\dots=n} \binom{a_1+j_1-1}{j_1} \binom{a_2+j_2-1}{j_2} \binom{a_3+j_3-1}{j_3} \dots \binom{a_{n-1}+j_{n-1}-1}{j_{n-1}} \quad (1)$$

於是有  $n$  個電路元件的電路網絡數為  $a_n + b_n = 2a_n$  (當  $n > 1$  時)。

我們可以藉由枚舉所有的  $j_1 + 2j_2 + 3j_3 + \dots = n$  的組合，代入  $a_n$  的關係式來計算。經過實際的測試，在  $n < 100$  時，這種計算方法所花費的時間勉強可以接受<sup>2</sup>(見表一)。但經過觀查注意到，隨著  $n$  的等差增加，所花費的時間是呈指數成長的。

N	40	50	60	70	80	90	100	110
秒數	0.15	0.94	0.59	2.77	11.69	44.92	159.36	529.80

表一：枚舉  $j_1 + 2j_2 + 3j_3 + \dots = n$  的組合來計算的時間花費

因為需枚舉所有的  $j_1 + 2j_2 + 3j_3 + \dots = n$  的組合，使得這種形式的關係式將難以快速的計算，因此必得將  $j_1 + 2j_2 + 3j_3 + \dots = n$  消除，才可能得出有效的計算方法。

<sup>2</sup>處理器為 Pentium D 2.8 GHz，所使用的程式原始碼見附件二

以下我們將試圖解出  $a_n$  的生成函數。

由二項式定理，我們可以將  $\frac{1}{(1-z^r)^a} = (1-z^r)^{-a}$  展開為

$$\frac{1}{(1-z^r)^a} = \sum_{-a \leq j \leq 0} \binom{-a}{j} (z^r)^j = \sum_{-a \leq j \leq 0} \binom{a+j-1}{j} z^{rj} \quad (2)$$

我們構造一個函數  $\prod_{0 < k} \frac{1}{(1-z^k)^{a_k}}$ ，以上式 (2) 代入

$$\prod_{0 < k} \frac{1}{(1-z^k)^{a_k}} = \prod_{0 < k} \sum_{-a_k \leq j_k \leq 0} \binom{a_k + j_k - 1}{j_k} z^{kj_k}$$

分析這個函數中  $z^n$  項的係數，顯然右式中  $z$  的指數項部分需有

$j_1 + 2j_2 + 3j_3 + \dots = n$ ，因此  $z^n$  的係數可以表示為

$$\sum_{j_1 + 2j_2 + 3j_3 + \dots = n} \prod_{0 < k \leq n} \binom{a_k + j_k - 1}{j_k}$$

這個式子中包含了  $a_n$  這一項，將它提出

$$\sum_{j_1 + 2j_2 + 3j_3 + \dots = n} \prod_{0 < k < n} \binom{a_k + j_k - 1}{j_k} + a_n$$

與  $a_n$  的關係式 (1) 比較，發現

$$\sum_{j_1 + 2j_2 + 3j_3 + \dots = n} \prod_{0 < k < n} \binom{a_k + j_k - 1}{j_k} + a_n = a_n + a_n = 2a_n$$

因此將  $\prod_{0 < k} \frac{1}{(1-z^k)^{a_k}}$  展開後，對於  $a$  的第  $n > 1$  項， $z^n$  的係數恰等於  $2a_n$ ，令一多

項式  $A(z) = a_2 z^2 + a_3 z^3 + \dots = \sum_{n > 1} a_n z^n$  為數列  $a_n$  的生成函數，比較係數的結果，即

可得到：

$$2A(z) + 1 - z = \prod_{0 < k} \frac{1}{(1-z^k)^{a_k}} \quad (3)$$

這並不是一個很好的形式，因為數列  $a_k$  仍然存在等號右邊，且連乘積的形式也不容易展開。但我們已經可以利用這個式子一項項的計算每一個  $a_n$ ，在上面的

式子中  $a_n$  僅與  $a_k, k < n$  相關，故在計算  $a_n$  僅需代入  $k < n$  次項，則  $z^n$  的係數將會是正確的  $a_n$ 。

進一步的處理，對等式 (3) 兩邊取自然對數，有

$$\begin{aligned}\ln[2A(z)+1-z] &= \ln\left[\prod_{k>0} \frac{1}{(1-z^k)^{a_k}}\right] \\ &= \sum_{k>0} \ln \frac{1}{(1-z^k)^{a_k}} \\ &= \sum_{k>0} a_k \ln \frac{1}{1-z^k}\end{aligned}$$

爲了將 $\ln$ 展開，可以將其中的 $\ln$ 用 $\int \frac{1}{z} dz$ 來取代

$$\begin{aligned}\ln \frac{1}{1-z^k} &= \int \frac{1}{\frac{1}{1-z^k}} dz^k \\ &= \int (1-z^k) dz^k \\ &= \int (1+z^k+z^{2k}+\dots) dz^k \\ &= \left[ \sum_{j>0} \frac{(z^k)^j}{j} \right] + C, C=0\end{aligned}$$

代回原式，得到

$$\begin{aligned}\ln[2A(z)+1-z] &= \sum_{k>0} \sum_{j>0} \frac{a_k (z^k)^j}{j} \\ &= \sum_{j>0} \frac{1}{j} \left[ \sum_{k>0} a_k (z^j)^k \right] \\ &= \sum_{j>0} \frac{1}{j} A(z^j)\end{aligned}$$

在等式兩邊以指數  $e$  換回

$$\begin{aligned}2A(z)+1-z &= \exp\left[\sum_{j>0} \frac{A(z^j)}{j}\right] \\ &= \exp\left[A(z) + \frac{A(z^2)}{2} + \frac{A(z^3)}{3} + \frac{A(z^4)}{4} \dots\right]\end{aligned}\tag{4}$$

等式 (4) 兩邊對  $z$  取導數

$$\begin{aligned}\frac{d}{dz}[2A(z)+1-z] &= \frac{d}{dz} \exp\left[\sum_{j>0} \frac{1}{j} A(z^j)\right] \\ \left[\frac{d}{dz} 2A(z)\right]-1 &= \exp\left[\sum_{j>0} \frac{A(z^j)}{j}\right] \left[\frac{d}{dz} \sum_{j>0} \frac{A(z^j)}{j}\right] \\ &= \exp\left[\sum_{j>0} \frac{A(z^j)}{j}\right] \left[\sum_{j>0} \frac{jz^{j-1} \frac{dA(z^j)}{dz^j}}{j}\right] \\ &= \exp\left[\sum_{j>0} \frac{A(z^j)}{j}\right] \left[\sum_{j>0} z^{j-1} \frac{dA(z^j)}{dz^j}\right]\end{aligned}$$

左式中以  $A(z) = \sum_{n>1} a_n z^n$  代入，右式中由 (4) 有  $\exp\left[\sum_{j>0} \frac{A(z^j)}{j}\right] = 2A(z)+1-z$ ，分別代入展開後可以得到

$$\begin{aligned}2 \sum_{n>1} n a_n z^{n-1} &= [2(\sum_{k>1} a_k z^k) + 1 - z] \left[\sum_{j>0} (z^{j-1} \sum_{t>0} t a_t z^{j(t-1)})\right] \\ &= [2(\sum_{k>1} a_k z^k) + 1 - z] \left[\sum_{j>0} \sum_{t>0} t a_t z^{jt-1}\right]\end{aligned}$$

分析  $z^{n-1}$  的係數，左式中為  $2na_n$ 。右式中  $z^{n-1}$  的指數將由右式中左右兩個括號中分別提供一部分得到，如果左括號中已取  $k$ ，則右括號中  $z$  的指數必取  $tj-1 = (n-1)-k$ ，化簡得  $tj = n-k$ ，即  $t | (n-k)$ 。右式左括號中額外的  $+1-z$  必須提出來處理。比較左右式中  $z^{n-1}$  的係數，可得到

$$2na_n = 2 \sum_{n>k>0} \sum_{t|n-k} t a_t a_k + \sum_{t|n, t<n} t a_t - \sum_{t|n-1} t a_t$$

即

$$a_n = \frac{2 \sum_{n>k>0} \sum_{t|n-k} t a_t a_k + \sum_{t|n, t<n} t a_t - \sum_{t|n-1} t a_t}{2n}, a_1 = 1$$

此為  $a_n$  的遞迴關係式，對於計算  $a_n$ ，可直接枚舉所有的  $0 < k < n, t | (n-k)$ 。已知  $a_k, k < n$  時計算  $a_n$  大約只需要與  $n^2$  呈正比的枚舉量，相對於前文中的枚舉所有  $j_1 + 2j_2 + 3j_3 + \dots = n$  組合方法的指數成長時間，最後的這個遞迴式，是十分快速的<sup>3</sup>。計算出  $a_n$  後，由於  $b_n = a_n$  因此有  $n(n > 1)$  個電路元件的電路網絡數量為  $a_n + b_n = 2a_n$ 。

<sup>3</sup> 以此遞迴式做計算的程式原始碼見附表三

## 陸、 結論

令一數列  $\{a_n\}$ ， $a_1 = 1$ ， $a_n$  的關係式為

$$a_n = \frac{2 \sum_{n>k>0} \sum_{t|n-k} ta_t a_k + \sum_{t|n, t<n} ta_t - \sum_{t|n-1} ta_t}{2n}$$

則有  $n$  個電路元件的相異電路網絡數量，除了  $n=1$  時為 1，當  $n>1$  時為  $2a_n$ 。

## 柒、 討論

一、直接的枚舉是簡單但確沒有效率的。將組合方法簡單的分類後可以直接寫出直觀的關係式，對於  $n$  不相當大時是可以接受的。

二、利用生成函數，把冗長的關係式壓縮到簡潔的  $2A(z) + 1 - z = \prod_{0<k} \frac{1}{(1-z^k)^{a_k}}$

裡，解出最後的遞迴關係式，使得計算的成本大幅下降。使對於  $n$  的增長，計算第  $n$  項的時間成長能保持在  $n^3$  的多項式時間內，比起直接枚舉的指數成長的時間有著相當大的差異。

三、使用最後的遞迴關係式計算第  $n$  項時，可以發現分子的不同  $\sum$  值是受到多次重覆計算的，可以通過預處理將已計算過的部分儲存下來，則計算的時間花費可以降至  $n^2$ 。

## 捌、 參考資料及其它

1. Ronald L. Graham, Donald E. Knuth, Oren / World Scientific / Concrete mathematics
2. Donald E. Knuth / Pearson Education, Inc / The Art of Computer Programming Vol.1 / chapter 2.3.4.4
3. 曹亮吉 / 遠哲科學教育基金會 / 阿草的葫蘆：文化活動中的數學

附表一：電路網絡組合數的部分項值

1	1
2	2
3	4
4	10
5	24
6	66
7	180
8	522
9	1532
10	4624
11	14136
12	43930
13	137908
14	437502
15	1399068
16	4507352
17	14611576
18	47633486
19	156047204
20	513477502
100	5940657225988390865225693961584860944632501010661006
200	299274689802693037665262918358727913215493804218913260120170240314689 14084620838159314648498611694480795664
500	220804058372166947601061023249779566702607248929421136017728874842154 807482928315435154099077923543195537204575413308184218352604489055668 306982947947921717855188847162938442373224517089460399761357580592806 30111348135842950329545889868845508816207689623378052808617699896

附件二:直接枚舉  $j_1 + 2j_2 + 3j_3 + \dots = n$  組合方法之計算方法原始碼 (不考慮數字過大溢位的問題)

```
#include <stdio.h>
#include <time.h>

long long a[101];
long long sum;

void dfs(int x,int n,long long s) {
    int i;
    long long j,k;
    if(x==1) {
        sum += s;
    }else {
        dfs(x-1,n,s);
        for( i=x,j=1,k=a[x]; i<=n; i+=x,j++,k++ ) {
            s = s*(long long)k/(long long)j;
            dfs(x-1,n-i,s);
        }
    }
}

int main() {
    int n,i;
    a[1] = 1;
    a[2] = 1;
    for( n=3; n<=101; n++ ) {
        sum = 0;
        dfs(n-1,n,1);
        a[n] = sum;
    }
    for( n=2; n<=101; n++ ) a[n]*=2;
    while(scanf("%d",&n) && n) {
        printf("%I64d\n",a[n]);
    }
    return 0;
}
```

附件三：使用解出的遞迴關係式計算的原始碼 (不考慮數字過大溢位的問題)

```
#include <stdio.h>

long long a[501];

int main() {
    int i,j,n;
    a[1] = 1;
    for( n=2; n<501; n++ ) {
        a[n] = 0;
        for( i=1; i<n; i++ ) {
            for( j=1; j<n; j++ ) {
                if((n-i)%j==0) {
                    a[n]+=a[i]*a[j]*j;
                }
            }
        }
        a[n]*=2;
        for( j=1; j<n; j++ ) {
            if(n%j==0) {
                a[n]+=a[j]*j;
            }
        }
        for( j=1; j<n; j++ ) {
            if((n-1)%j==0) {
                a[n]-=a[j]*j;
            }
        }
        a[n]/=n;
    }
    for( i=2; i<501; i++ ) a[i]*=2;
    while( scanf("%d",&n) && n ) {
        printf("%I64d\n",a[n]);
    }
    return 0;
}
```

