2009 年臺灣國際科學展覽會

優勝作品專輯

編號： 010016

作品名稱

傑克船長的心機

得獎獎項

數學科大會獎第一名

紐西蘭正選代表: 2009 年紐西蘭科技展覽會

學校名稱： 臺北市立建國高級中學

作者姓名： 王新博

指導老師： 曾俊雄

關 鍵 字： 樹、塗色、歸納

作者簡介

我喜歡閱讀、打球、摺紙，最喜歡的則是數學。

回顧第一次接觸數學競賽後，開啟了我數學美麗殿堂的大門。也因此有機會參加各式各樣不同的競賽，不但增廣見聞，我也結識了許多同好。更從此深深著迷於變化多端的數學。除了上學的時間外，假期中，我參加了一些數學活動及俱樂部，如：九章數學愛好者聯誼、數學夏令營等，最有趣的還是加拿大 Alberta 大學辦的「加拿大青少年數學夏令營」。

Abstract

The main theme of this project is the study of a game of strategy. Here is the setting.

Captain Jack and his pirates had looted an even number of treasure chests. Each treasure chest contained a number of gold coins. Captain Jack had a pirate count them and record the number on the chest where everyone could see.

Captain Jack proposed to share the gold coins with the crew according to the following plan. The crew chose the cleverest pirate to represent them. He would chain the treasure chests to one another in a row, arranging them in any order he chose. Captain Jack and he would alternate taking a treasure chest from either end, with Captain Jack choosing first, until all treasure chests had been taken.

It might appear that the crew had the advantage in being able to arrange the treasure chests in any order they chose, but Captain Jack could guarantee that his share of gold coins would be no less than that of the crew.

This is the basic scenario of our study. It will progress in three stages, according to how the treasure chests were chained together.

1. Along a path --- with two end points.

2. In the shape of a three-pronged star --- with three end points initially.

3. In a general tree diagram --- with many end points initially.

In each stage, we seek a method by which Captain Jack could guarantee to get at least as many gold coins as the crew.

The mathematical background consists of combinatorial analysis, graph theory, coloring methods and mathematical induction.

We solved the cases with 2,4,6 or 8 treasure chests last year. Now we want to solve the general cases.

中文摘要

本研究的主題是一個數學策略遊戲：

傑克船長與他的海盜們掠奪到了許多箱珍寶，每箱內有數量不等的金幣，海盜們清點後將每箱金幣的數量寫在箱子上。傑克船長為了彰顯他對弟兄們的愛護，他訂定了一個分配珍寶的方案：

由船員們推派一名最聰明者任意將寶箱排成一棵樹，由船長開始兩人流拿取寶箱，每次可拿走任何一個只和一個寶箱相鄰的寶箱（即樹上的葉）和箱內的金幣。

傑克船長表面上似乎很大方，事實上他是很奸巧的，無論這位聰明的海盜如何排列寶箱，他總有巧妙的方法能使最後所取得的金幣不少於海盜們所取得的金幣。

假設珍寶有偶數箱，作者利用塗色法、數學歸納法、組合學、圖論等數學方法，為傑克船長找到一個策略，可以保證最後所取得的金幣不少於海盜們所取得的金幣。

去年解決了寶箱數為2、4、6、8的情況，今年我們將研究的目標推廣到所有的圖形。

傑克船長的心機

一、前言

（一）研究動機

三年前我去參加數學夏令營，教授在課餘時間和我們玩了一個遊戲，規則是這樣子的：有 20 個箱子內裝了數量不等的金幣，並在箱外寫明金幣的數目。將箱子任意排成一排，並規定由雙方輪流任意選取兩端的箱子。取完後，累計雙方取得金幣的總數，如果先手不比後手少，就算先手獲勝，反之為敗。

這個遊戲及其推廣引發我的好奇心。一方面，先手占有優勢，另一方面，先手初居劣勢的情況太好構造了；一方面，先手總能在最後一步逆轉，另一方面，這些反敗為勝的方法沒甚麼規律。大部分的方法只能適用於少數的金幣分佈，我們希望能有更簡單、更完善的解答，因此著手研究此題目。

（二）研究目的

遵守以下的規則，為先手找出無論如何都不會輸的取法：

給一棵偶數階（指頂點個數）的樹（我們只討論偶數個頂點的），每一個頂點相當於一個實數。由先、後手輪流拿取葉，最後雙方分別將自己所取的數相加。對方的和較己方多為輸，反之為贏（允許雙贏）。

我們以六個頂點組成的「H」形樹為例，數字採 100 以下的隨機正整數。不過在拿取之前，我們需要知道哪些頂點是可以取的，因此我們要另外再畫一個圖，將每個頂點的度（Degree）標上去。如圖 1 所示：

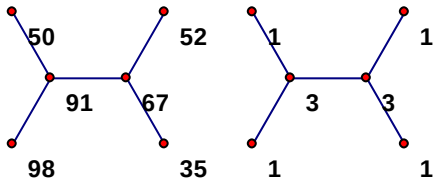

圖 1

圖形周圍那四個頂點的度小於二，因此它們是可取的，那麼先手任取其中一個。我們在數字後面加上「F」表示他是被先手（first）拿走的，並且用虛線表示已經不存在的邊。如圖 2 所示：

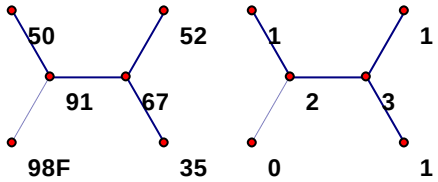

圖 2

先手取完後，注意到有個頂點的度因此改變了，這是讓這個問題有趣的地方，每一步都會影響這棵樹接下來的發展。現在可拿取的頂點剩三個，而後手（second）也任取其中一個，以「S」表示。如圖 3 所示：

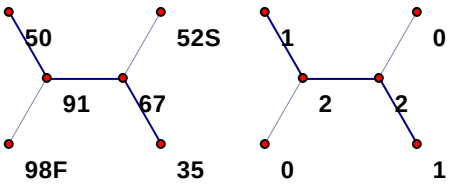

圖 3

此時可拿取的頂點剩兩個了。先手再任取一個。如圖 4 所示：

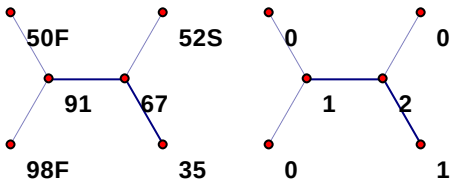

圖 4

可拿取的頂點改變了，但是數量不變。後手再任取一個。如圖 5 所示：

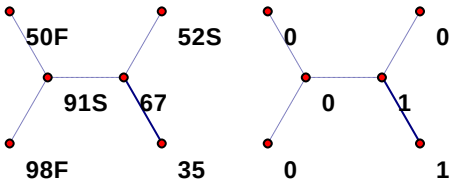

圖 5

最後兩步。如圖 6 所示：

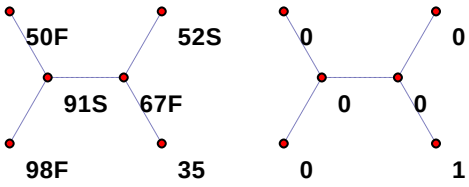

圖 6

續下頁

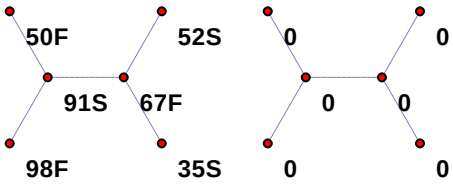

圖 6(繼續)

先手：98+50+67=215。

後手：52+91+35=178。

先手勝。

（三）研究回顧

去年的報告中，已經提出一些的取法，包括所有 8 個頂點以內的樹及任意的數字分布，先手都可以獲勝。

8 個頂點以內的樹一共有 32 種，除了其中一半可以歸納外，其他的樹都必須逐棵討論，雖然用了比較簡潔的描述方式，但是各個取法之間並無共通性，不適合推廣。

再向前看，10 個頂點的樹一共有 106 種，數量更多、更複雜。其中只有不到 50 種可以用已知的方式歸納，如果仍堅持要 case by case，不僅曠日廢時，也沒有意義。

這一次我們希望可以直接找出一般解，在這個過程中，不乏一些看似正確卻暗藏盲點的取法。經屢次修改後，終於找到完整而正確的取法了。

二、研究方法

（一）樹的代號

在研究這個問題的同時，我們需要一套代號來描述不同的樹。以「8 個頂點鏈成一串」為例，這樣的樹直接以頂點的數量命名，所以這種樹就是「8」。如圖 7 所示：

圖 7

「頂點串成一串」是一個很明確的敘述，相較之下，以下的敘述就有點冗長：「四個頂點、其中一個頂點連接另外三個」，這種樹是「(1,1,1)」。如圖 8 所示：

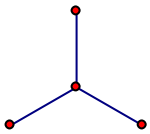

圖 8

乍看之下可能認為沒必要寫得這麼麻煩，不過，如果是「一個頂點連接另外五個頂點、這五個又分別連接到另外五個」就用到了，這是「(2,2,2,2,2)」。如圖 9 所示：

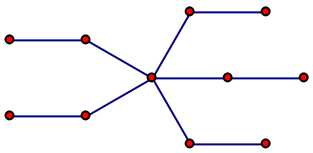

圖 9

當然，我也可以只用一個數字表示有幾組一模的頂點，但是遇到頂點數都不同就沒轍了，像「(3,4,5,6)」，也就是「一個頂點連接到四組頂點、每組分別有三、四、五、六個頂點」。如圖 10 所示：

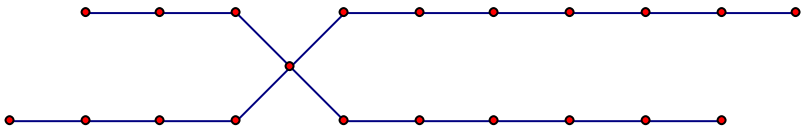

圖 10

最後依序為「(1,1)1(1,1)」「(1,1,1)(1,1)(1,1,1)」「((1,1),(1,1),(1,1))」。如圖 11 所示：

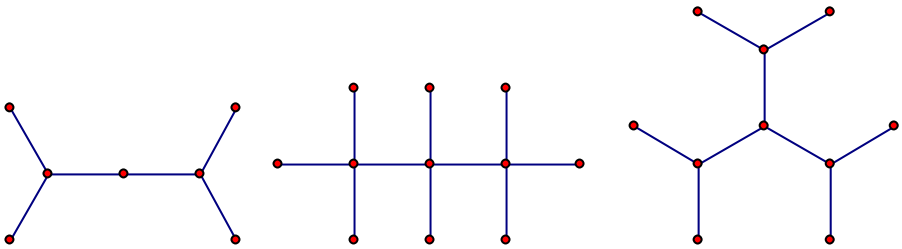

圖 11

（二）塗色法

對代號形式為「2n」的樹來說，他們是兩種最特別的樹的其中一種，也是這個問題最初的版本。我們將這樣的樹的頂點依序塗上黑、白、黑、白之後，可以看出先手一定可以拿到全部的黑頂點（如果他想的話，也可以全部都拿白的）。如圖 12 所示：

圖 12

我們認為致勝點有兩個：一方面，黑、白兩色是交錯塗的，另一方面，頂點數是偶數。這兩個結合在一起時就變成：輪到先手時總有一個黑葉，取走之後變成兩個白葉，後手任取一個白的之後又會多出另一個黑的。

如此進行，先手就可以拿到所有黑色的頂點。相反的，如果沒有規定頂點的數量是偶數，我們也可以用塗色法證明先手不一定必勝。事實上，只要將「1、3、1」依序填入「3」中，勝負就很明顯了。如圖 13 所示：

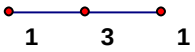

圖 13

由於先手有絕對的選擇權，哪怕是改變判定勝負的方式，只要先手選定了會贏的顏色，拿完就贏了。我們想到這個取法之後，才有了把圖形推廣的想法。把頂點排成樹狀圖、原先規定的「兩端」改為「葉子」後，才有了這個研究。

（三）歸納法

首先要介紹的是另一種最特別的樹：「(1,1,1,…,1)」，這和上述的樹剛好形成兩種代號上的極端。他最特別的地方就在於後手只能拿先手拿剩下的，因此若先手每一步都拿最大的，就贏定了。如圖 14 所示：

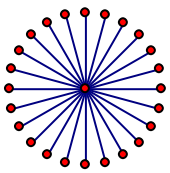

圖 14

關鍵就在於，先手每一輪都較後手取的多，所以最後一定是先手勝。換句話說：如果一棵 $n+2$ 個頂點的樹經過一輪後先手領先，而且先手知道 n 個頂點的樹該如何處理，那麼先手就征服了 $n+2$ 個頂點。這就是「歸納」的由來。

不過先手拿到剩 n 個頂點時領先時，卻可能不知道 n 個頂點的取法。但是，先手只要可以再一次、再兩次的取得領先，頂點數遲早可以到達我們可以接受的範圍。所以只要我們能證明，先手每次都可以在遊戲結束之前取得領先，就等於解出了這個問題。

我們傾向於把「歸納」想成：「先手將之前的領先（一定要是領先）一筆勾銷，把剩下的樹看成另一顆新的樹，並在這棵樹取得勝利」這個動作。經由歸納，整個遊戲過程就可以分成好幾個部分，而且每個階段先手取的都得較後手多。

我們選擇以歸納作為一般解的基礎，畢竟只要保證先手能領先一次就好。但是就如同動機所寫的，我們不乏先手初居劣勢的情況。該如何能保證先手一定能領先，並沒有表面上看起來那麼簡單。

（四）複葉

通常一棵樹可以指定一個特殊的頂點當根。但是我們要用另外一種方式去定義：把度為二的頂點稱為「莖」，度比莖多的頂點就叫「根」(root)、以「R」表示，而比莖少的就叫「葉」(leaf)、以「L」表示。如圖 15 所示：

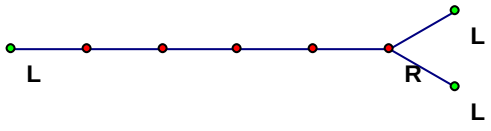

圖 15

一片葉拿走之後，原本和該葉相鄰的莖會「退化」成葉。如果再把這一片新的葉拿走，原來的地方又會「長出」另外一片。如此進行到無葉可拿時，我們把這一整串被拿掉的頂點稱作「複葉」(compound leaf)、以「C」表示。如圖 16 所示：

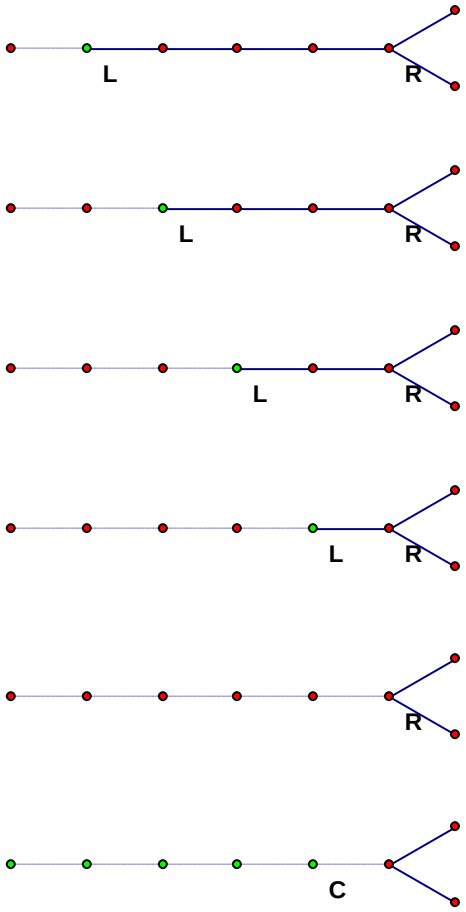

圖 16

易知，「8」沒有根、有六個莖、兩片葉，而且它本身就是一串複葉（姑且不考慮要從哪一邊開始拿）。同理，「(1,1,1)」有一個根、沒有莖、三片葉，而且這三片葉本身就是三串複葉。

但是「(2,2)2(2,2)」雖然有六個莖，卻只有四串複葉，因此莖和複葉之間並沒有一一對應。至於葉和複葉之間雖然有一一對應，但是葉是頂點、複葉是點集。

如圖 17 所示：

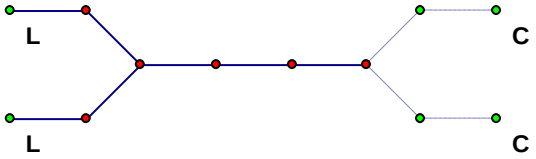

圖 17

我們令「*Ce*」代表有偶數（even）個頂點的複葉，所以$|Ce|\equiv 0 \pmod 2$（絕對值在此指集合的元素個數、或者說是複葉的頂點個數）。類似的，用「*Co*」代表有奇數（odd）個頂點的複葉，所以$|Co|\equiv 1 \pmod 2$。

此外用「*V*」代表頂點（vertex），特別是複葉中的頂點。在一串複葉中，最先被拿走的頂點是「V_1」，第二個被拿走的是「V_2」……依此類推。而「*v*」（小寫）則代表頂點對應到的實數：「V_1」對應到「v_1」、「V_2」對應到「v_2」……。如圖 18 所示：

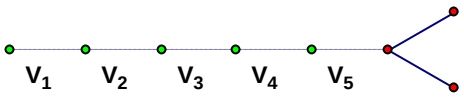

圖 18

（五）和與交錯和

有複葉的概念之後，接著再來介紹一些我們需要用到的函數。（文末附有簡表）

拿走某一串複葉第一片葉的人，我們稱他為這串複葉的「主動者」（active），

另一方則稱為「被動者」(passive)。不同的複葉可能有不同的主、被動者，但彼此之間並無影響，也和先、後手無關。我們假設主、被動者是輪流拿取一串複葉的。如圖 19 所示：

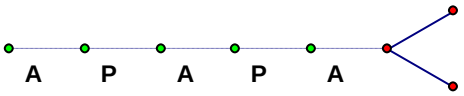

圖 19

在一串複葉上，令「*Sa(n)*」是主動者在該複葉上取的前 n 個頂點的和（*Sum*），定義域是不大於[*(|C|+1)/2*]的非負整數，其中「*|C|*」依然是該複葉的頂點個數、「[]」是高斯符號，值域是實數。所以 $Sa(n)=Sa(n-1)+v_{2n-1}=\Sigma v_{2i-1}$。如圖 20 所示：

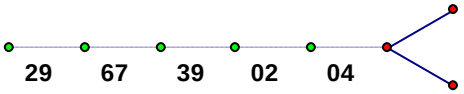

圖 20

Sa(0)=0、*Sa(1)=29*、*Sa(2)=29+39=68*、*Sa(3)=29+39+04=72*、*Sa(4)*沒有定義。

同理，令「*Sp(n)*」是被動者在該複葉上取的前 n 個頂點的和，不同的是，它的定義域是不大於*[|C|/2]*的非負整數。那麼 $Sa(n)=Sa(n-1)+v_{2n}=\Sigma v_{2i-1}$。

如圖 21 所示：

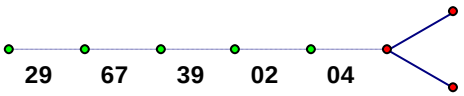

圖 21

Sp(0)=0、*Sp(1)=67*、*Sp(2)=67+02=69*、*Sp(3)*沒有定義。

如果想比較兩數的大小，也可以比較零和兩數之差的關係。想知道主、被動者誰拿得多，就直接把兩個拿來減減看。而減出來的結果就是主動者的「淨利」了（把被被動者拿走的當成本的話）。

令 *So(n)=Sa(n)-Sp(n-1)*、*Se(n)=Sa(n)-Sp(n)*。他們的意義在於，算出這個複葉的淨利有助於先手決定誰是「*Cs*」。依定義可以展開成 $So(n)=So(n-1)-v_{2n-2}+v_{2n-1}=\Sigma(v_{2i-1}-v_{2i})+v_{2n}$ 以及 $Se(n)=Se(n-1)+v_{2n-1}-v_{2n}=\Sigma(v_{2i-1}-v_{2i})$。

如圖 22 所示：

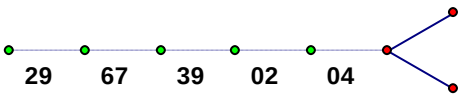

圖 22

So(1)=29-0=29、*So(2)=68-67=1*、*So(3)=72-69=3*、*So(4)*沒有定義。

Se(1)=29-67=-38、*Se(2)=68-69=-1*、*Se(3)*沒有定義。

兩者的差異在於，「So(n)」的展開式中總共只有奇數個頂點、計算到的最後一個頂點是主動者拿的（所以比被動者多拿一個）。相對的，「Se(n)」就比較「公平」，主、被動者拿的頂點一樣多。

（六）研究方向

我們之前遇過一個瓶頸：目前的研究方向設定為所有種類的樹，也就是說，如果我們真的找到一個拿法的話，這個拿法的描述必須是一般性的、適用於所有情況。例如「取當時最大的葉」就很明確，「取位於中間的葉」就令人不知所云。

更進一步地，對於後手每一種可能的拿法，都要有一個標準化的過程可以教先手應對後手的方式。但是可能的樹、可能的拿法時在太多了，我們不知道一般

化後的應對到底會長成什麼樣子。

所以我們乾脆立下一個規定：除了第一步以外，先手每一步都得跟著後手拿，後手碰哪一串複葉，先手就接著去拿那串複葉的下一個葉。這樣一舉解決了先手應對的問題，而且它恰好符合先手使用塗色法時的應對方式。

同時，我們只要考慮先手第一步該取哪片葉（或哪一串複葉）即可，剩下的就交給歸納了。由於這一片葉是如此重要，重要到我們要用「*Ls*」來代表它，指「那個被選中的」（selected）葉。同理，「*Cs*」就是「那個被選中的」複葉。

但是這樣就等於白白浪費了先手「優先選擇」的優勢，規定之初我們一直對它抱著疑惑。但是事實證明它是有用的，而且我們也還想不到其他更好的方法。

三、研究過程

（一）無根的樹

我們以（只有）八個頂點的「8」為例，再解釋一次塗色法。如圖 23 所示：

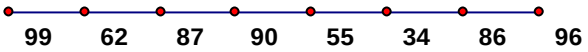

圖 23

將頂點交錯黑白塗色，意即將頂點塗成一黑一白、相鄰不同色的形式。如圖 24 所示：

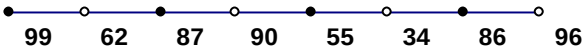

圖 24

比較黑色頂點的和及白色頂點的和。如圖 25 所示：

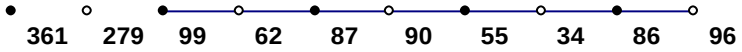

圖 25

先手取總和較大的顏色的頂點。在例子裡是黑色，於是先手取黑色的頂點。如圖 26 所示：

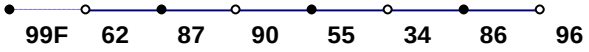

圖 26

輪到後手時，葉都是白色的。在這裡我們暫時以貪心法則模擬後手的取法。如圖 27 所示：

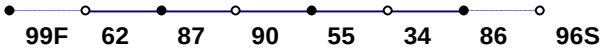

圖 27

第二輪換成先手了，葉中恰有一個是黑色的頂點，於是先手取這個頂點。如圖 28 所示：

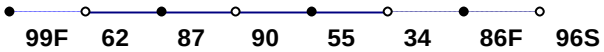

圖 28

發現了嗎？輪到後手時葉又變成兩白了，所以後手還是只能取白點。

如圖 29 所示：

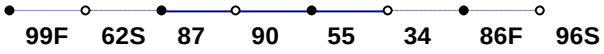

圖 29

此時此刻，又有一個黑色的葉，則先手將其取走。如圖 30 所示：

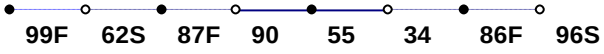

圖 30

依此原則，最後三步。如圖 31 所示：

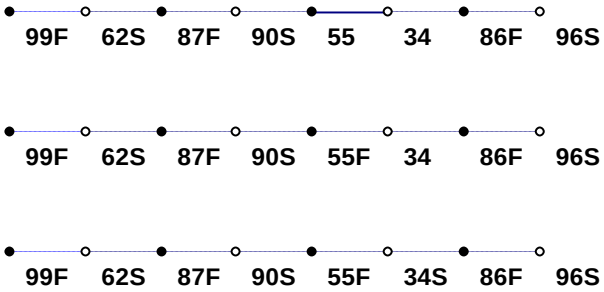

圖 31

先手：99+86+87+55=361。

後手：96+62+90+34=279。

先手勝。

在此可以很明顯的看到先手「每次」都能拿到黑色的頂點，而後手「只能」拿到白色的頂點。雖然它不適用於有根的樹，卻非常值得介紹。我們可以用（傳統的）歸納法一般化這個過程。

（二）有根的樹

我們以（只有）四個頂點的「(1,1,1)」為例，再解釋一次歸納法。

如圖 32 所示：

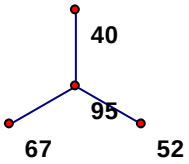

圖 32

簡單嘗試後發現，無論先手第一輪怎麼取，中間那個頂點都不會變成葉。如圖 33 所示：

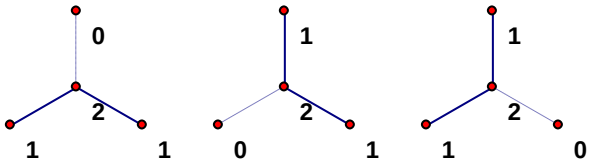

圖 33

也就是說，我們只要考慮其他三個頂點就好。如圖 34 所示：

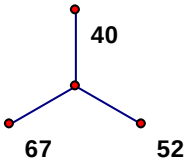

圖 34

那先手當然是取最大的。如圖 35 所示：

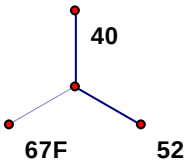

圖 35

輪到後手，它取到的一定不比先手大。如圖 36 所示：

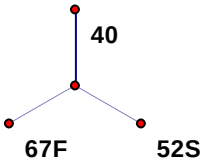

圖 36

最後剩下兩個頂點，先手就可以用比較簡單的方式拿取。如圖 37 所示：

圖 37

最後兩輪。如圖 38 所示：

圖 38

先手：67+95=162。

後手：52+40=178。

先手勝。

再來看一個例子：「(1,1,1,1,1)」。如圖 39 所示：

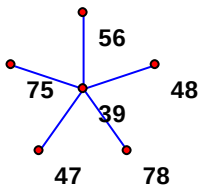

圖 39

同樣地，我們發現第一輪後，中間的頂點也不會變成葉。如圖 40 所示：

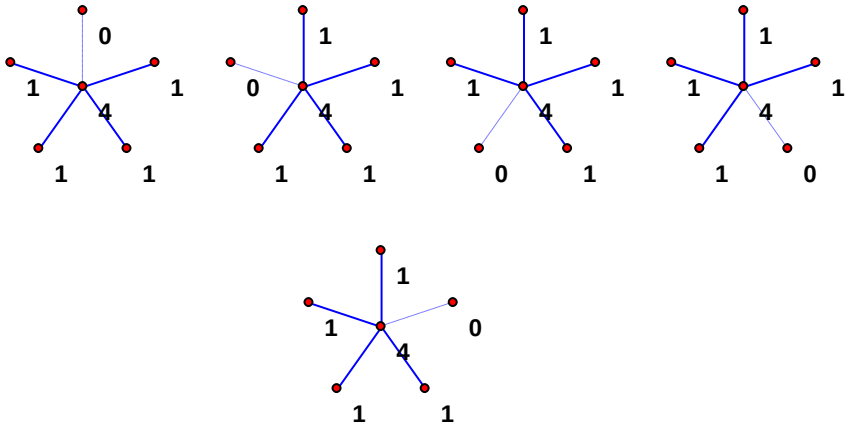

圖 40

所以，我們也只要考慮其他五個頂點即可。如圖 41 所示：

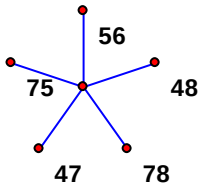

圖 41

同樣的道理，雙方都盡量取大的。如圖 42 所示：

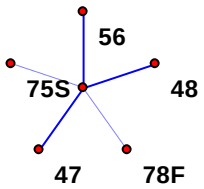

圖 42

仔細觀察剩下的圖形，感覺有點眼熟。如圖 43 所示：

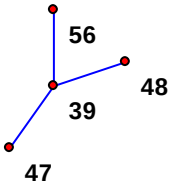

圖 43

它就是「(1,1,1)」啊！我們只要直接套用剛剛我們發現的取法就可以了。如圖 44 所示：

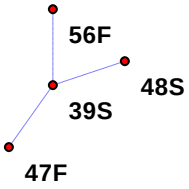

圖 44

先手：78+56+47=181。

後手：75+48+39=162。

先手勝。

當然，我們也可以不用把先、後手的總和算出來，就可以確定先手必勝了。但是並不是每種樹都只有輪生葉（指很多葉直接長在同一個根上）。只要將「1、2、3、0」依序填入「4」中，就不能歸納了。如圖 45 所示：

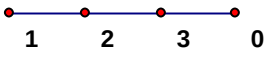

圖 45

雖然沒有辦法使用歸納，不過這棵樹太簡單了（連塗色都不需要），一看就知道應該先取左邊的葉。但是，這一類的例外可以無限放大。如圖 46 所示：

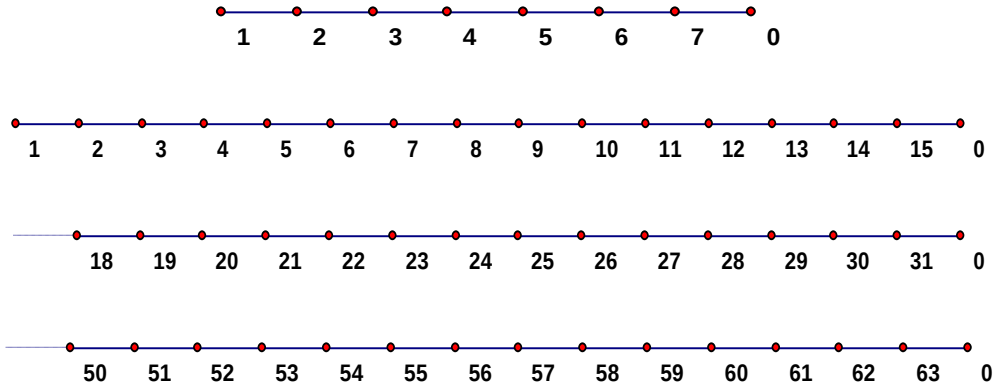

圖 46

好吧，現在也許需要塗色了，但是不使用歸納又有什麼關係？

之前提到：塗色法不適用於有根的樹，我們再換一個：「(1,3,3)」。如圖 47 所示：

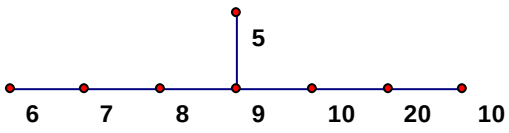

圖 47

同理，這類的例外也可以無限放大。如圖 48 所示：

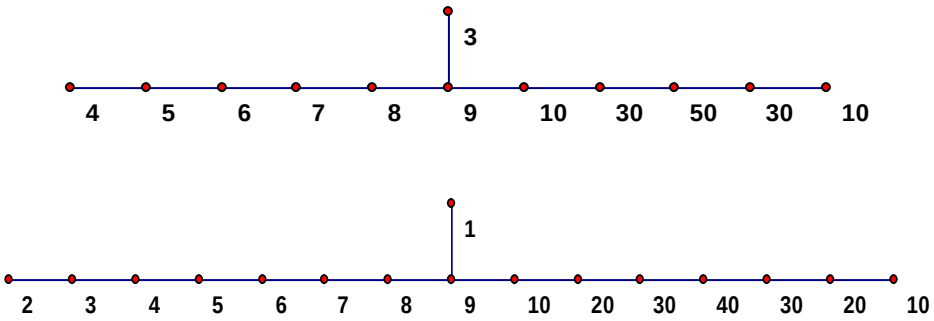

圖 48

看來是大有關係。

（三）假根

如果是複葉都很長的樹、例如上面的「*(1,n,n)*」系列或是「*(4,4,4,5,5,5)*」，我們都會下意識的希望外面的葉比裡面的莖還大、一輪就可以歸納。如圖 49 所示：

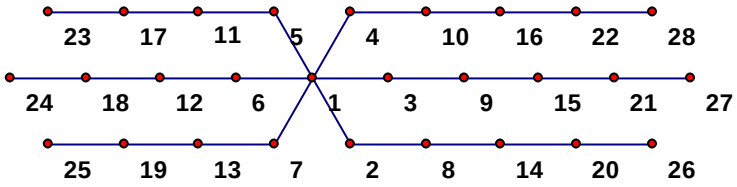

圖 49

但是有一個問題：後手知道跟著拿的話，先手就會歸納（雖然先手歸納與否並不影響後手進行遊戲），所以會跑去拿別的葉，該怎麼辦？如圖 50 所示：

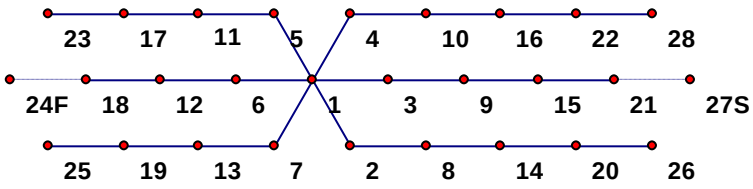

圖 50

先手：24=24。

後手：27=27。

先手落後。

那簡單，只要一開始拿最大的就行了。如圖 51 所示：

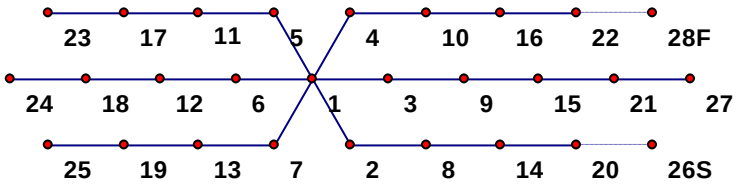

圖 51

先手：28=28。

後手：26=26。

先手歸納。

不過一般情況下，樹才不會長得那麼剛好。但是如果先、後手輪流拿同一串複葉，先手還是有超過一半的機率在拿完前領先。（從現在起我們要忽略複葉之間的結構，單獨畫出每一串複葉。）如圖 52 所示：

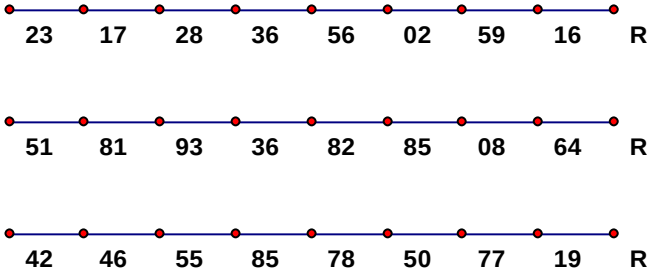

圖 52

簡單計算後發現，拿完第三串複葉後會贏最多，於是先手往這個方向努力。如圖 53 所示：

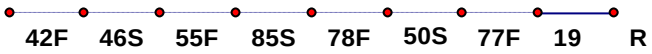

圖 53

顯然後手並不會想去拿那個頂點，所以先手不能這樣取，並不是在單一複葉上就都可以歸納的。不過既然後手不會拿它、先手也不會動它，乾脆把它當成（假）根好了。其他的複葉也要比照辦理。如圖 54 所示：

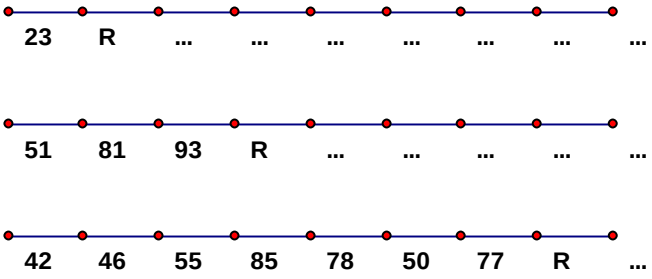

圖 54

我們把它標準化：在一串複葉上，令「*u*」（upturn）是所有滿足「*Se(n)>0*」的「*n*」中，最小的那一個。根據定義，假根就是「V_{2u}」，而且對於每串複葉，在被動者拿假根之前，主動者在該複葉都不可能領先。

所以對於給定的複葉，最多只會有一個「固定的」「*u*」、一個「固定的」「*Se(u)*」，不同於「*n*」是一個變數。而如果該複葉所有的「*Se(n)*」都是負的，那「*u*」就不存在，「*Se(u)*」當然也不存在。

（四）最小值

我們再來看另外一組例子，其中假根已經標示出來了。如圖 55 所示：

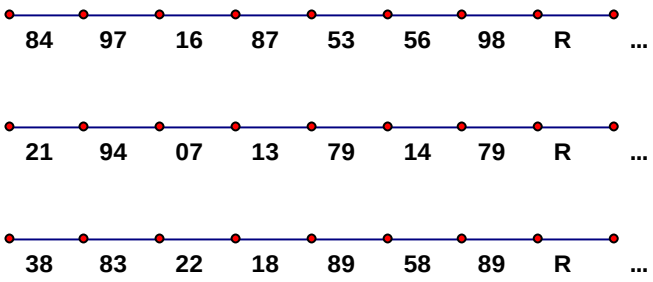

圖 55

三串複葉的「*So(u)*」分別是「11、65、79」。

最初的想法是：由於第三串複葉的「*So(u)*」最大，所以把它當「*Cs*」。如圖 56 所示：

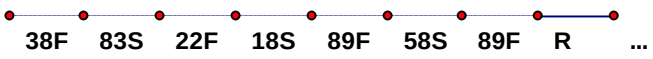

圖 56

然後後手取另外一個「*So(u)*」沒那麼大的分支。如圖 57 所示：

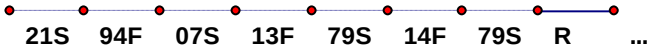

圖 57

不過真的在拿第三串複葉的時候，後手不太可能這麼合作。如圖 58 所示：

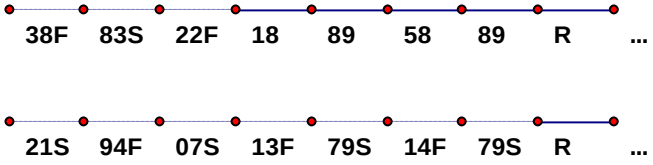

圖 58

先手：38+22+94+13+14=181。

後手：83+21+07+79+79=269。

先手大落後。

這個地方也出現類似「假根」的情形，後手總會「卡」在「*So(n)*」最小的地方，這樣就算這串複葉的「*So(u)*」是最大的也於事無補。不過，先手當然也能依樣畫葫蘆、提早歸納。如圖 59 所示：

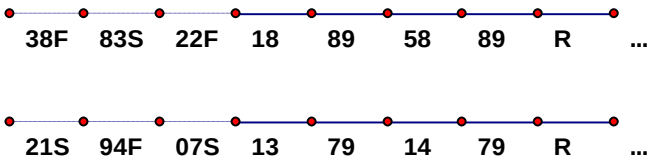

圖 59

先手：38+22+94=154。

後手：83+21+07=111。

先手歸納。

我們把它標準化：在一串複葉上，令「*m*」(minimum)是所有滿足「$n \leqq u$」(「*u*」不存在則無此限制）的「*n*」中，使「*So(n)*」最小的那一個。根據定義，對於給定的複葉，恰一個「固定的」「*m*」、一個「固定的」「*So(m)*」，不同於「*u*」可有可無。

「*Cs*」就是「*Se(m)*」最大的複葉（不考慮沒有「*u*」的「*Ce*」，這點還會再解釋)。歸納的時機是：後手拿到「*Cs*」的假根或其他任何一串複葉的「V_{2m-1}」的那一輪。(注意：先手從第二輪開始就「跟著後手拿」）可以歸納的原因如下：

如果後手先拿到「*Cs*」的假根，根據「*u*」的定義，*Sa(u)>Sp(u)*，故可歸納。若後手先拿到的是其他任何一串複葉的「V_{2m-1}」，根據「*Cs*」的定義，該複葉的「*So(m)*」較「*Cs*」的小，故可歸納。

同時，先手也不必擔心後手拿其他的複葉，還拿到很大的。前面提到：根據「*u*」的定義，在被動者拿假根之前，主動者在該複葉都不可能領先。此時，這些複葉的主動者正是後手，這代表後手涉獵的複葉越多，先手就會領先越多。

（五）例外

在上述的取法中，後手似乎連一個根都拿不到，其實是可以的。對一個沒有「*u*」的「*Ce*」來說，雖然後手拿它會有損失，但是先手沒辦法（用歸納）阻止他拿完（而先手歸納之前都得跟著後手拿)。如圖 60 所示：

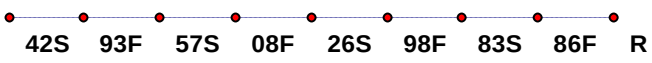

圖 60

如果後手還不放棄（或者他可能很有遠見），又拿了另一串複葉。如圖 61 所示：

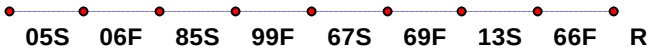

圖 61

再一串。如圖 62 所示：

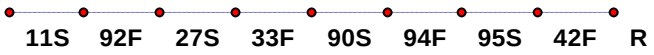

圖 62

現在來看整棵樹的樣子，這是「*(2,(1,1),(1,1),(8,8,8))*」。如圖 63 所示：

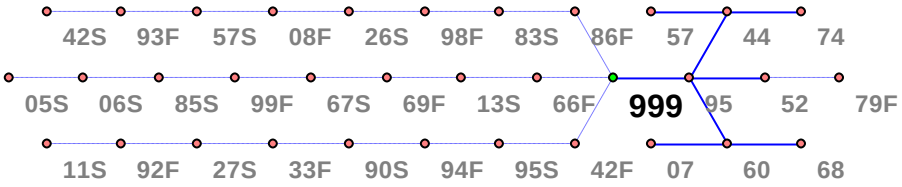

圖 63

顯然後手是贏定了，完全不考慮根的確會造成大災難。

除此之外，如果一個根只連接一串「*Co*」、其他都是「*Ce*」的話，該根也可以（在歸納之前）被後手拿到。不過只要一個根連接到兩個以上的「*Co*」、或是連接

到任何一個存在「*u*」的複葉，這個根（在歸納前）就碰不到了。

儘管如此，我們還是可以保證先手必勝，但是有幾個理由我們要跳過他：

1. 若根恰連接兩串複葉，則「*Co*」少於兩個的機率是 3/4，三串是 1/2，越多越小。

2. 其中一串複葉不存在「*u*」的機率小於 1/2，「|*C*|」越大機率越小。

3. 另外一串複葉不存在「*u*」的機率也小於 1/2，「|*C*|」越大機率越小。

4. 如果有第三串，不存在「*u*」的機率也小於 1/2，「|*C*|」越大機率越小。

5. 後手拿根之前，先手還不能歸納的機率也小於 1/2。複葉越多機率越小。

6. 就算根「解開」了，它幫得到後手的機率也小於 1/2，複葉越多機率越小。

（六）例子

我們以「*(4(5,6),(1)(2,3))8(4(5,6),(1)(2,3))*」為例，講解全部的流程。為了適應頁面大小，右半邊的數字調整至頂點的左上方。如圖 64 所示：

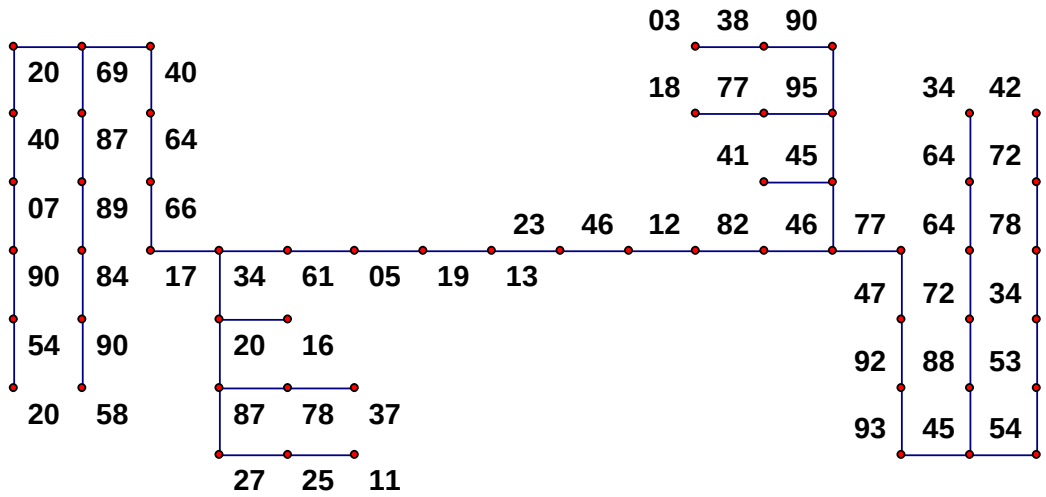

圖 64

先把所有的根和複葉都找出來。如圖 65 所示：

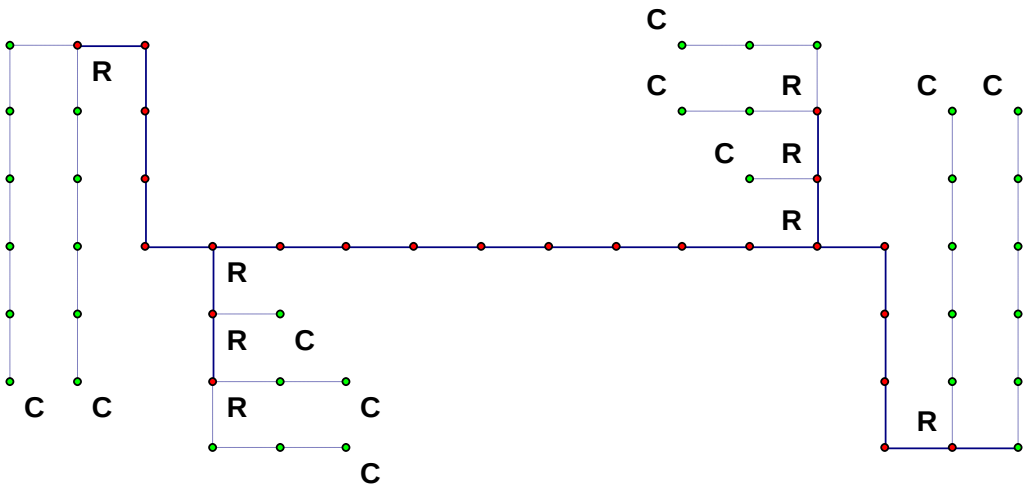

圖 65

接下來找假根和「*So(m)*」。如圖 66 所示：

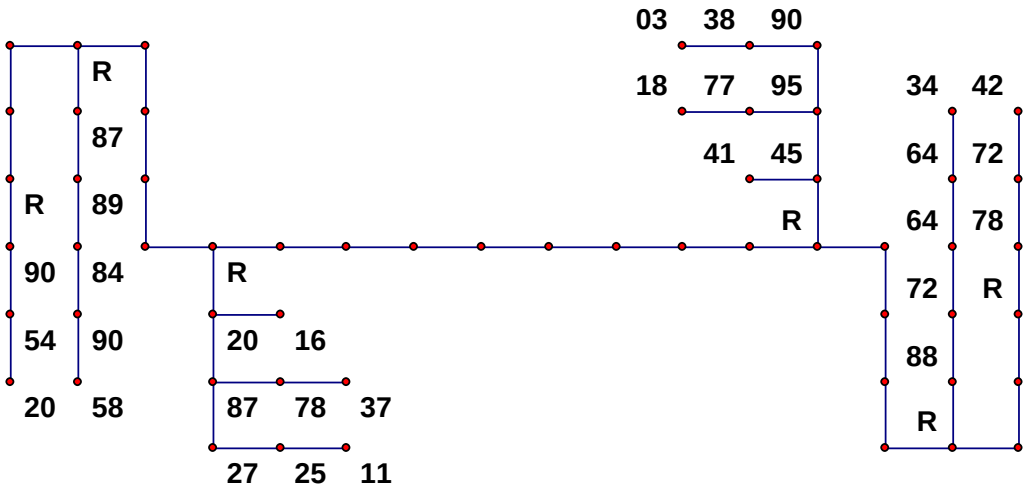

圖 66

左上角由左至右：*So(1)=20*、*So(3)=50*。

左下角由上至下：*So(1)=16*、*So(1)=37*、*So(1)=11*。

右上角由上至下：*So(1)=03*、*So(1)=18*、*So(1)=41*。

右下角由左至右：*So(1)=34*、*So(1)=42*。

故「*Cs*」是左起第二串複葉。

先、後手依序拿取「58」、「90」、「84」、「89」、「87」，將先手的領先縮減至 50。後手高興之餘還拿了左下角的複葉，這個地方就是所謂會拿到根的地方，只是在這裡拿根沒有辦法平反就是了。

接著後手再取當時葉中最大的「42」，算是拿到該複葉的「V_{2m-1}」。如圖 67 所示：

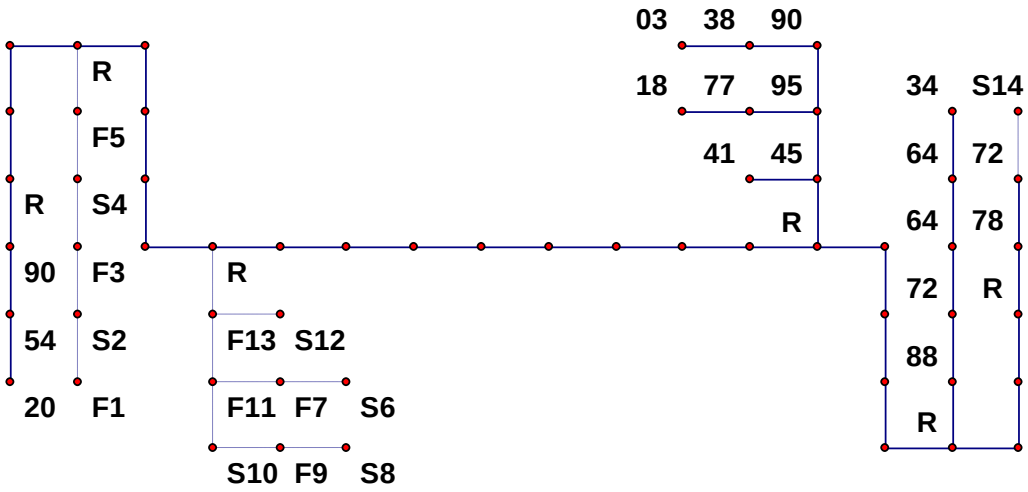

圖 67

先手：*58+84+87+37+25+87+20=398*。

後手：*90+89+37+11+27+16+42=312*。

先手歸納。

剩下的樹是「(5,5)3(20)(1)(2,3)」。如圖 68 所示：

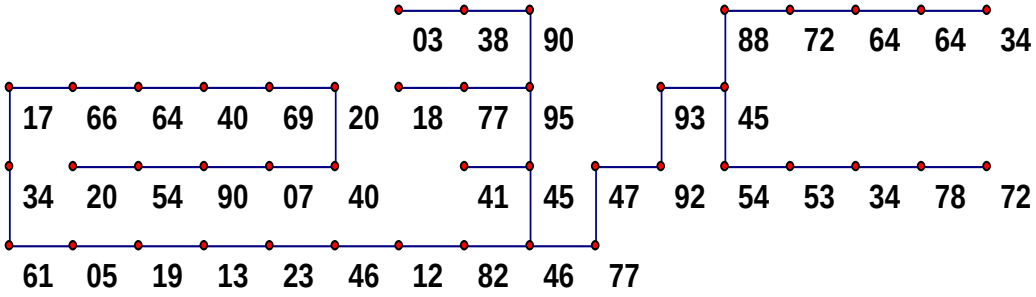

圖 68

同樣的，找出複葉和根。如圖 69 所示：

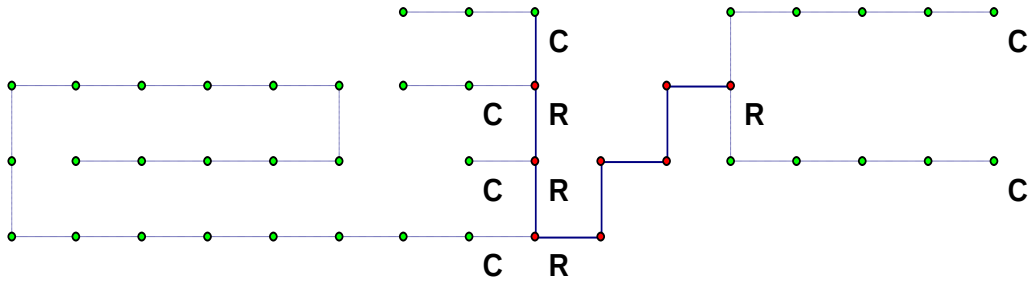

圖 69

再找假根和「*So(m)*」。如圖 70 所示：

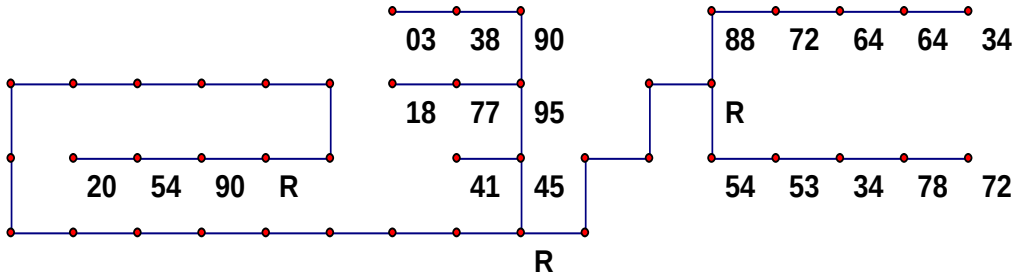

圖 70

左邊：So(1)=20。

中間上起：So(1)=03、So(1)=18、So(1)=41。

右邊上起：So(1)=34、So(2)=28。

故「*Cs*」是中間最下面的複葉。

先手取「41」，後手只能取「72」避免歸納，接著依序取「78」、「34」。如圖 71 所示：

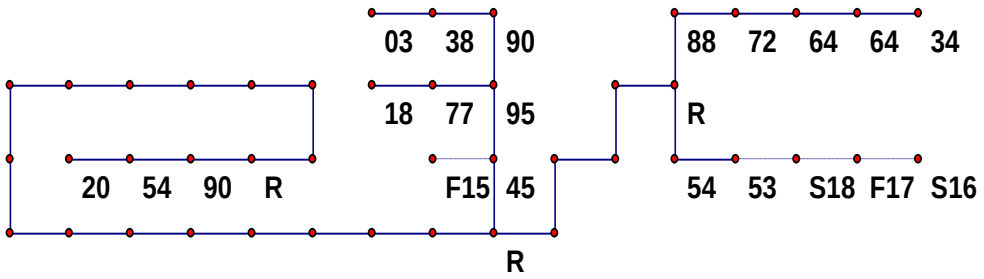

圖 71

先手：41+78=119。

後手：72+34=106。

先手再歸納。

四、研究結果與討論

（一）研究結果

1. 對任何一棵偶數階的樹，先手必勝。

2. 有無限多棵奇數階的樹，先手不能必勝，而且很容易構造出來。

3. 不考慮後手拿根的情況，我們可以在多項式時間（$O(|V|)$）內算出先手的第一步。

4. 拿完一棵樹需要歸納多次，但仍然只需多項式時間（$O(|V|^2)$）。

5. 討論所有情況的話，複雜度將介於指數（$\Omega(2^{|V|})$）和階乘（$O(|V|!)$）之間。

（二）討論

我們看到，如果把頂點數改為看似沒那麼「公平」的奇數，先手反而不能必勝了。除了讓一個先手拿不到的頂點非常大以外，還有另一種保證可行的辦法，那就是將數字同時減去一個很大的數。先手多拿一個頂點的結果就是多一份損失。

除了頂點個數，勝負判定的方式也可以改變。例如用絕對值相乘來代替相加，甚至保留正負號的相乘。絕對值相乘可以用取對數的方式轉換為相加，缺點是不能處理零。保留正負號的相乘就更刺激了，目前還沒有研究出來。

依此也可以延伸出允許頂點代表複數，以相加後的絕對值較大為勝，或者是更高維空間中的向量相加後的長度、多重積的大小等等。

度的限制也可以改變，把可以拿取的頂點擴大至三以下，或是限制當時度最少的那些頂點才可取。如此一來就可以在所有的圖上進行遊戲了。

我們也可以允許玩家「退貨」，把不要的頂點再放回去。或是用拿一個頂點的機會交換重排整個圖形的權利，不過先手只要把後手重排過的圖排回來就好了，沒有什麼太大的意義。

五、結論與應用

（一）結論

去年，因為我們只著重在低階樹上，對樹的巨觀性質（例如「$Se(u)$」）沒有深入研究，因此對大場面的應對沒有概念，自然找不到一般性的取法。

曾經也試過限制頂點代表的數只能是「0」或「1」，尋找不限頂點個數的解。

雖然最後失敗了，但是對樹的巨觀性質總算有一些認識，也萌生一些有關「*S*」系列函數的想法。

而引入「*Se(n)*」、「*So(n)*」之後雖然稍有突破，但就像「研究過程」中寫的，中間一度卡在「*Se(u)*」的地方。但後來發現它是很重要的觀念，相當於將一棵樹分成好多部分去分析。

（二）應用

從演算法的觀點來看，這個一般解可以將指數時間縮短至多項式時間，為一大進步。儘管討論所有情況可以一並處理先手可以取到的極大值，且優化後（例如以較大的頂點做為優先搜尋）也能在較短的時間內給出答案。

但是討論所有情況後得到的取法沒有規律、長期來說時間複雜度也不夠穩定，對某些設計過的樹，我們必須算到最後一刻才知道答案。更重要的，我們不會知道先手是不是永遠必勝。

我們希望這個研究可以催生新的演算法，甚至為 NP 問題貢獻一點心力。

六、參考文獻

1. 王新博　2008　傑克船長的心機　2008 國際科學展覽會 參展作品專輯
2. 林福來　離散數學初步　四版　台北市　九章　P71~P147
3. 邵慰慈、潘建強　基礎離散數學　一版　台北市　九章　P113~220　2005
4. 張遠南　使人聰明的數學遊戲　一版　台北市　九章　P205　1996
5. Peter Winkler　Mathematical Puzzles　Canada　AK Peters　P1~P2　2004

附錄：代號一覽

D　度（degree）與該頂點相鄰的頂點個數。

F　先手（first）先拿的人，通常落後都可以逆轉，奇數階的樹則不盡如此。

S　後手（second）後拿的人，有很種多樹會讓後手領先，但是都維持不久。

R　根（root）度大於二的點，和慣例不同，我們允許一棵樹有很多根。

L　葉（leaf）度小於二的點，和慣例相同，也是玩家可以拿取的頂點。

C　複葉（compound leaf）從一片葉走到離它最近的根之前，經過的所有頂點的集合。

Ce　偶數（even）個頂點的複葉，$|Ce|\equiv 0 \pmod 2$。

Co　奇數（odd）個頂點的複葉，$|Co|\equiv 1 \pmod 2$。

V_n　頂點（vertex）足碼表示從葉往根數的第 n 個頂點。

A　主動者（active）拿「V_1」的人，通常會接著拿「V_3」、「V_5」、「V_7」、……

P　被動者（passive）主動者的對手，通常會接著拿「V_2」、「V_4」、「V_6」、……

Sa(n)　主動者在該複葉上取的前 *n* 個頂點的和（*Sum*），$Sa(n)=Sa(n-1)+v_{2n-1}=\Sigma v_{2i-1}$。

Sp(n)　被動者在該複葉上取的前 *n* 個頂點的和（*Sum*），$Sa(n)=Sa(n-1)+v_{2n}=\Sigma v_{2i-1}$。

Se(n)　主動者的淨利，$Se(n)=Sa(n)-Sp(n)=Se(n-1)+v_{2n-1}-v_{2n}=\Sigma(v_{2i-1}-v_{2i})$。

So(n)　主動者的淨利，$So(n)=Sa(n)-Sp(n-1)=So(n-1)-v_{2n-2}+v_{2n-1}=\Sigma(v_{2i-1}-v_{2i})+v_{2n}$。

Se(u)　$Se(u)>0$，且 u 是所有符合條件的自變量中最小的，可能不存在。

So(m)　$m \leqq u$，且 *So(m)*是所有符合條件的應變量中最小的，一定存在。

評語

本文主要在研究樹狀圖上的一種對局，原來前人能做的只在路徑的特例能解。這篇文章把所有樹狀圖都解決了，是一個不錯的結果。建議在文章的撰寫上可以有一些改進。

(1)在一開始把題目就說清楚，並 survey 別人的結果，說明本文的結果，做一比較，讓人知道貢獻何在。

(2)將樹的演算法寫得更清楚一點

(3)對有奇數點的樹，提供一演算法，決定先拿的人是否會贏。

(4)將演算法的複雜度降低。

(5)提一下以後可以做的問題。