數學傳播　38卷3期, pp. 13-24

# 九連環與格雷碼

郭君逸

## 1. 前言

九連環, 是個家喻戶曉的童玩, 相信很多人小時候都花了很大的功夫才將其解出。過程中, 頭腦裡一直環繞著一些遞迴模型, 不小心還會出錯, 後來只好拿紙出來, 一面解一面紀錄當前的狀態, 才能夠不出錯的將「環」與「劍」成功分開。分開後, 應該很少人還有力氣把它立刻還原, 通常還原都是幾天後的事了。後來, 在大學時期, 學了一些代數、與編碼理論後, 才體會到, 經由數學的幫忙, 解出九連環不但不難, 其實還蠻容易的, 只是時間問題而已。最近無意間看到了幾篇相關文章, 又引發了我當年的興趣, 將我的一些心得與大家分享。

九連環的起源, 有人說是戰國時代的惠施發明, 但依戰國策中提到的「玉連環」只有「兩環相貫」, 與九連環並不相同; 另外有人說是三國時期諸葛亮爲排遣其妻之寂寞而發明, 至於實際如何, 已不可考。目前坊間的九連環, 其外觀如下圖:

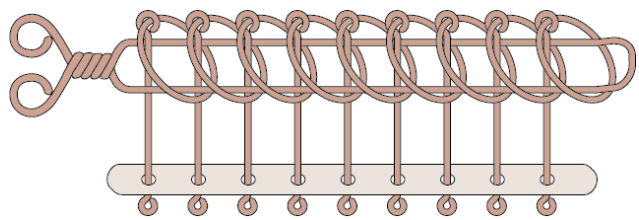


圖 1: 圖片取自 [1]

原始的玩法是要把上方的「劍」與「九個環」分離。每個環有兩種狀態, 一個是被劍穿過(掛在劍上), 一個是與劍分離。每次改變一個環的狀態, 稱作一步。但是並不是每個環都可以任意改變狀態的, 只有符合下列兩條件之一的環 (依上圖的左右方向), 才能任意改變狀態:

($R$) 最右邊的環

($S$) 劍上最右邊環之左邊的環

這兩個動作其實不容易觀察出來, 很多已能將九連環解開的人, 還是沒觀察出所有的動作完全都是由這兩個步驟組成。然而這個觀察, 對於下一節的數學模型非常非常的重要。除了單純

把九連環解出來外, 我們還希望能回答或證明下列的問題:

1. 若每個環一開始可以自由指定狀態, 是否都能解開?
2. 有解的狀態, 是否都是唯一解?
3. 有解的狀態, 幾步可以解出?
4. 任給兩個狀態 $x$ 與 $y$, 是否可以在有限步中把 $x$ 移動成 $y$, 幾步?
5. 任給兩個狀態 $x$ 與 $y$, $x$ 移動成 $y$ 的過程中, 第 $k$ 步是移動第幾個環?
6. 任意狀態 $x$, 移動了 $k$ 步後, 會變成什麼狀態?

最後, 我們還會提到與九連環相關的數學遊戲與其推廣。

## 2. 數學模型與解法

我們用 1 表示環在劍上, 用 0 表示環不在劍上, 而九個環的狀態, 連續寫成一個 9 位二進位數, 以此來表示九連環目前的狀態。而兩種操作方式 $R$ 與 $S$, 視爲狀態函數。舉例來說: $R(101011010_2) = 101011011_2$, 或 $S(101011010_2) = 101011110_2$。九連環的玩法, 轉成此模型, 相當於是在問「是否存在一連串 $R$ 或 $S$ 的操作, 使得 $111111111_2$ 會轉變成 $000000000_2$」? 進而我們將討論更一般的情況, $n$ 個環, 任意狀態 $x$, 是否能將所有的環打開?

因爲 $R$ 或 $S$ 連續運作兩次, 則會還原回原來的狀態, 因此可以得到下列性質:

**性質 1.** $R$ 與 $S$ 皆爲 involutions。亦即對於所有狀態 $x$, $R(R(x)) = S(S(x)) = x$。

還有另一個觀察:

**性質 2.** 除了 $000\cdots00_2$ 與 $1000\cdots00_2$ 不能做 $S$ 的操作外, 其餘的狀態都能夠操作 $R$ 與 $S$。

因爲性質 1的關係, 可以知道, 其實九連環的解法, 不外乎就是「$R \to S \to R \to \cdots$」, 或是「$S \to R \to S \to \cdots$」兩種而已, 差別就是誰先開始。因爲所有的狀態有限, 又這些動作都是可逆的, 所以馬上可以有一個初步的解法「策略一」產生:

(A) 先由 $R$ 開始,「$R \to S \to R \to \cdots$」交替, 因爲狀態有限, 所以一定會發生下列情況之一:
   1. 出現「$000\cdots0_2$」即解開, 並結束
   2. 出現循環, 表示無解
   3. 出現「$1000\cdots00_2$」無法再進行 $S$ 操作, 此時逆操作所有的動作還原回 $x$, 然後進行 (B) 策略。

(B) 改成由 $S$ 開始,「$S \to R \to S \to \cdots$」交替, 因爲狀態有限, 所以一定會發生下列情況之一:

1. 出現「$000\cdots0_2$」即結束
2. 出現循環, 表示無解
3. 出現「$1000\cdots00_2$」無法再進行 $S$ 操作, 亦無解。

以下開始, 在不會與十進位混淆的情況下, 我們將省略二進位的下標 2, 以求簡潔。

**例 1.** 假設要解四連環 $x = 1010$ 的狀態, 先做 $R$ 得 $R(x) = 1011$, 再做 $S$ 得 1001, 再做 $R$ 形成了1000, 此時已經無法再做 $S$ 了, 因此將剛剛的動作逆回去還原成 $R(S(R(1000))) = x$, 改成先做 $S$, 得 $S(x) = 1110$, 再做 $R$ 得 1111, 再做 $S\ldots$, 共 12 步之後, 得到 0000。

這樣的解法策略, 其實不用懂數學, 只要能夠清楚知道 $R$ 與 $S$ 這兩個步驟在做什麼, 交替進行, 就可以把九連環解出來。而上面的策略, 每種狀態, 只會有「無解、一個解、兩個解」三種情況; 但經過了幾次的試驗, 發現每次都能夠解開, 而且先做 $R$ 或先做 $S$, 只有其中一個能解開; 因此我們不禁推測, 是不是每種狀態都有解, 而且唯一解?

我們借「連環圖」的輔助, 來回答此問題。$n$ 連環對應的連環圖 $G_n = (V, E)$ 爲一個簡單圖, 其中點集合 $V = \{0,1\}^n$ 爲所有狀態, 而邊集合 $E = \{\{x, y\} \mid y = R(x)$ 或 $y = S(x)\}$。

若 $n$ 連環中任兩個狀態 $x, y$, 都可以經過一連串 $R$ 或 $S$ 的操作互相達成的話, 等同於「$G_n$爲連通圖 (connected)」; 若這步驟是唯一解的話, 等同於「$G_n$ 中任兩點有唯一的路徑 (Path) 相連」, 兩個性質合起來, 即「$G_n$ 是一顆樹 (Tree)」。又根據性質 2, 可以知道只有 $000\cdots0$ 與 $100\cdots0$ 兩個點的度 (degree) 是 1, 所有條件合起來, 等同於要證明:

**定理 1.** 對於所有正整數 $n$, 連環圖 $G_n$ 皆爲路徑 (Path)。

**證明**: 對 $n$ 做數學歸納法。

當 $n = 1$ 時, $G_1 = P_2$, OK。

假設 $G_n$ 是一條路徑, 而且 $x = 00\cdots0$ 與 $y = 100\cdots0$ 爲其端點。

在 $G_{n+1}$ 中, 假設

$$x' = 00000\cdots0 = 0x$$
$$y' = 10000\cdots0 = 1x$$
$$w = 01000\cdots0 = 0y$$
$$z = 11000\cdots0 = 1y$$

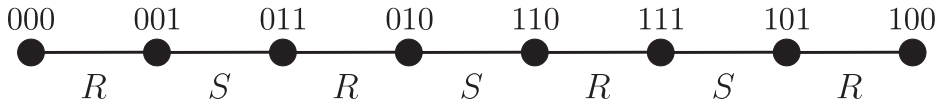
而 $x'$ 可以依照 $G_n$ 的連線方式, 連到 $w$, 此路徑包含所有最高位數爲 0 的點;

而 $y'$ 可以依照 $G_n$ 的連線方式, 連到 $z$, 此路徑包含所有最高位數爲 1 的點;

又 $S(w) = z$, 所以 $w$ 與 $z$ 在 $G_{n+1}$ 中有邊相連;

所以 $G_{n+1}$ 爲 $x' - w - z - y'$ 的一條路徑, 其中 $x'$ 與 $y'$ 爲端點。 □

**例 2.** 三連環的連環圖 $G_3$



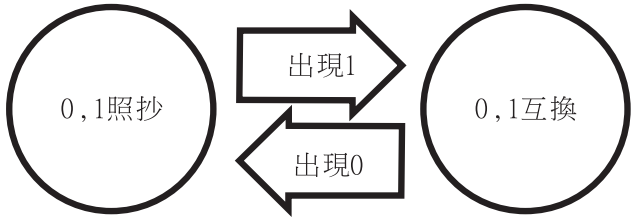

由於 $G_n$ 的圖形就是一條路徑, 因此對於 $n$ 連環的任何狀態都有解, 而且是唯一解。而且知道,「策略一」中並不會有「無解」的情況發生, 所以我們把它精簡成解法「策略二」:

> 先由 $R$ 開始,「$R \to S \to R \to \cdots$」交替進行, 若達不到 $000\cdots0$, 就會達到 $100\cdots0$, 此時再「$R \to S \to R \to \cdots$」交替進行即可。

另外, 由連環圖 $G_n$ 可知, 其實狀態 $111\cdots1$ 並不是需要最多步才能解出的, $1000\cdots0$ 才是距離 $000\cdots0$ 最遠的點。

接下來, 若我們有辦法計算出任意狀態 $x$, 先 $R$ 還是先 $S$, 才不會「走回頭路」的話, 那解法就完美了。想法是這樣的, 若要將狀態 $x = 011001001 = x_8x_7x_6\cdots x_0$ 解出的話, 最左邊的 $1 = x_7$ 就要改成 0, 但要改變 $x_7$ 的話, 須先 $x_6x_5\cdots x_0 = 1000000$ 才行, 而 $x_6$ 已經是 1, 所以須先 $x_5x_4\cdots x_0 = 00000$ 才行, 而 $x_0$ 到 $x_5$ 中, 最左邊的 $1 = x_3$, 須先改成 0 才行, 但要改變 $x_3$ 的話, 必須 $x_2x_1x_0 = 100$ 才行, 所以 $x_2 = 0$ 要先改成 1, 但要改變 $x_2$ 的話, 必須 $x_1x_0 = 10$ 才行, 所以 $x_1 = 0$ 必須先改成 1, 又因爲 $x_0$ 已經是 1 所以可以直接改變 $x_1$, 此動作相當於 $S$, 經過了這樣的推導, 可以得知, 狀態爲 $x = 011001001$ 的話, 先做 $S$, 可以不走回頭路的解出九連環。

這樣一連串的判斷方式, 可以將其寫爲一個 Finite Automaton 的函數 $f$:




**例 3.** 若 $x = 011001001_2$, 由左至右照抄直到出現 1, 得「$01\cdots$」; 然後變成 01 互換直到出現 0, 得「$010\cdots$」, 再變成 01 照抄直到出現 1, 得「$010001\cdots$」; 再變成 01 互換直到出現 0, 得「$010001110_2$」, 所以 $f(x) = 010001110_2$。

所以

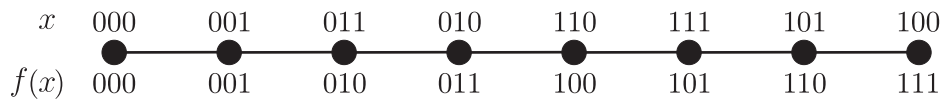
**定理 2.** 對於九連環的任意狀態 $x$, 完美解法如下:

(A) 若 $f(x)$ 爲奇數, 則以「$R \to S \to R \to \cdots$」解出。

(B) 若 $f(x)$ 爲偶數, 則以「$S \to R \to S \to \cdots$」解出。

我們將 $f(x)$ 與連環圖 $G_n$ 擺在一起來看:

**例 4.** 三連環的連環圖 $G_3$ 與 $f(x)$




將 $f(x)$ 與連環圖 $G_n$ 列出來比較的話, 馬上就可以看出, $f(x)$ 由左至右, 剛好是 0 到 $2^n-1$ 的 2 進位表示法。

**定理 3.** 對於任意狀態 $x$, 恰須要 $f(x)$ 步才能解出。

這個定理, 就留給讀者自行驗證了。

所以九連環的解法, $f(111111111_2) = 101010101_2$ 化成十進位爲 $2^0+2^2+\cdots+2^8 = \frac{4^5-1}{4-1} = 341$ 步。一般情況, $n$ 連環時, 當 $n$ 是奇數, $f(11\cdots1_2) = 1010\cdots01_2 = \frac{4^{\frac{n+1}{2}}-1}{4-1} = \frac{1}{3}(2^{n+1}-1)$; 當 $n$ 是偶數時, $f(11\cdots1_2) = 1010\cdots10_2 = \frac{2(4^{n/2}-1)}{4-1} = \frac{1}{3}(2^{n+1}-2)$。當然, 這樣的答案, 與差分方程「$a_n = a_{n-1}+2a_{n-2}+1$」解是一樣的:

**系理 1.** $n$ 連環解的步數爲 $\dfrac{1}{3}\Big(2^{n+1}+\dfrac{(-1)^{n+1}-1}{2}-1\Big)$。

有了定理 3 之後, 要做任意兩個狀態的轉換也就容易了:

**系理 2.** $n$ 連環, 狀態 $x$ 移動到狀態 $y$, 須要 $|f(x)-f(y)|$ 個步驟。

解法的話, 若 $f(x) > f(y)$ 的話, 則跟解開 $x$ 的方法一樣, 利用 $f(x)$ 的奇偶來判斷; 若 $f(x) < f(y)$ 的話, 則奇偶相反即可。

## 3. 格雷碼與相關研究

格雷碼 (Gray Code) 是貝爾實驗室的 Frank Gray 在 1940 年所提出的; 格雷碼是一個數列, 此數列的特性是「相鄰兩數間只有一個位元改數」。主要用於傳送訊號時, 防止訊號出錯的一種編碼方式。

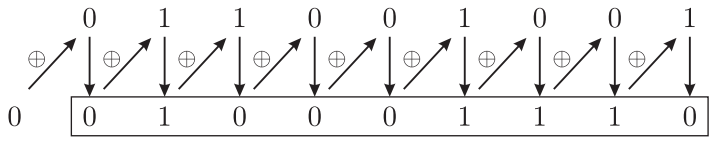
格雷碼並不是唯一的, 每種 $n$ 位元的格雷碼, 都相當於在 $n$ 位元的 Hypercube上的漢彌爾頓路徑 (Hamiltonian Path)。九連環之類的益智玩具, 因爲一次只能移動一個環, 所以很自然的, 畫出的連環圖就會是相鄰狀態只會差一個位元, 又因爲我們也證明了連環圖是個路徑, 所以它一定是一種格雷碼。然而最早提出的格雷碼, 與前面提到的連環圖是一模一樣的。所以有很多格雷碼上的性質都可以拿到九連環上使用。

$n$ 位元的格雷碼可以利用 $n-1$ 位元的格雷碼前面補 0 加上逆序之後, 前面補 1 而成, 這個性質的證明, 與定理 1 類似。

**例 5.** 「10,00,01,11」是個 2 位元的格雷碼, 前面補 0 得「010,000,001,011」, 而逆序後前面補 1 得「111,101,100,110」, 兩者連起來「010,000,001,011,111,101,100,110」即爲一種 3 位元的格雷碼。但若一開始就利用此種方式來產生格雷碼的話, 就會剛好是連環圖: 「∅」→「0,1」→「00,01,11,10」→「000,001,011,010,110,111,101,100」。

機械中的資料處理, 不外乎都是位元運算 (Bits Operations): AND, OR, XOR, NOT 等等, 至於上一節提到的 $f(x)$ 函數, 在機械中, 用位元運算的型式呈現會非常快。假設狀態 $x = x_{n-1}x_{n-2}\cdots x_1x_0$, $f(x) = y_{n-1}y_{n-2}\cdots y_1y_0$, 另外令 $y_n = 0$; 則 $k$ 依「大到小」的順序, $y_k = y_{k+1} \oplus x_k$ ($\oplus$ 代表 XOR 位元運算)。

**例 6.** 同例 3。狀態 $x = 011001001$ 經過上述的位元運算, 可得 $f(x) = 010001110$, 示意圖如下:




$f(x)$ 可以知道 $n$ 連環的某個狀態 $x$ 距離解開還有幾步, 但事實上, 在格雷碼中, 連 $f^{-1}(x)$ 都也可以求出。

**定理 4.** $f^{-1}(x) = x \oplus \lfloor x/2 \rfloor$.

有了這個定理, 我們可以算出第 $k$ 步之後狀態。

**例 7.** 由 $000000000_2$ 開始, 操作 123 步之後的狀態爲 $f^{-1}(123)$

$$123 \oplus 61 = 001111011_2 \oplus 000111101_2 = 001000110_2$$

**例 8.** 狀態 $x = 101101100_2$, 先操作 $R$ 開始, 60 步後的狀態爲何? 因爲 $f(x) = 110110111_2 = 439$ 爲奇數, 操作 $R$ 的話, 是往數字小的方向走, 所以 $439 - 60 = 379$, 因此最後的狀態爲 $f^{-1}(379) = 379 \oplus 189 = 111000110_2$ 。

另一個有趣的現象, 就是 $n$ 連環, 我們把最外 (最右) 的環編號定爲 1, 然後依序爲 $2, 3, \ldots$。然後, 從 $000\cdots00$ 爲初始狀態開始移動, 然後把動到的每個環紀錄下來, 會得到下面的數列:

$$1, 2, 1, 3, 1, 2, 1, 4, 1, 2, 1, 3, 1, 2, 1, 5, 1, 2, 1, 3, 1, 2, 1, 4, 1, 2, 1, \ldots$$

此數列最早是由 Louis Gros 在 1872 年最先提出, 因此稱爲「Gros Sequence」, (OEIS, A001511), 利用九連環的解法特性, 可以列出 Gros Sequence 的遞迴式爲:

$$g_k = \begin{cases} 1, & k: \text{ odd}, \\ g_{k/2} + 1, & k: \text{ even}. \end{cases}$$

也就是說 $g_k$ 等於 $k$ 因式分解後 2 的冪次加 1。雖然寫不出一般式, 但與計算 $f(k)$ 一樣, 可以在 $O(\ln k) = O(n)$ 的時間計算出來。

**例 9.** 由 $x = 101101100_2$ 狀態開始, 第 1 步先操作 $R$, 請問第 56 步, 是移動第幾個環? 同例 8, 因爲 $f(x) = 439$ 爲奇數, 操作 R 的話, 是往數字小的方向走, 所以 $439 - 56 + 1 = 384$, $g_{384} = g_{192} + 1 = g_{96} + 1 + 1 = g_{48} + 1 + 1 + 1 = \cdots = 8$, 或者直接由 $384 = 2^7 \cdot 3$, 所以是第 $7 + 1 = 8$ 個環。

Hinz [5] 證明了 Gros Sequence 有一個非常特別的性質。就是它是一個「貪心不重複數列 (Greedy Nonrepetitive Sequence)」。不重複數列在一些文獻上也稱作「Square-free Sequence」, 指的就是不存在兩個相同相鄰子數列。例如:「212342345678」就不是不重複數列, 因爲存在兩個相鄰的「234」; 而「12345678」是不重複數列。「貪心」指的就是部分和(Partial sum) 最小, 也就是說, 長度爲 5 的不重複數列中, 總和最小的就是 $1 + 2 + 1 + 3 + 1 = 8$。

另一方面, 若只用 1 與 2 兩個數字的話, 最長可以寫出長度爲 3 的不重複數列, 即「121」與「212」; 那只用 1, 2 與 3 三個數字的話, 最長可以寫出長度多少的不重複數列呢? 1906年, Thue [6] 用了一個很聰明的方法, 證明了這個答案是「無限大」。而目前圖論的研究中, 把不重複數列, 延伸出了「線上版 (online)」: 第一種型式就是由 Alice 與 Bob 兩人輪流由給定的字母集合中, 選一個字母出來填到數列上, Alice 的目的就是要讓此數列保持不重複, 但 Bob 的目的就是盡量讓此數列重複。2010年, Pegden [7] 證明了, 若只有 3 個字元的話, Bob 可以在 16 步內獲勝; 第二種型式, 也是目前非常熱門的討論問題, 由 Alice 從給定的字母集合中, 選一個字母出來填, 而 Bob 是指定 Alice 要填到哪個位置。2013 年, Grytczuk 等人 [8] 證明了, 若有 12 個字元的話, 玩再久 Bob 都無法獲勝, 若有 4 個字母的話, 則 Bob 定能獲勝。至於 5 到 11 之間, 誰勝誰負, 目前還未知。

Gros Sequence 與河內塔 (Hanoi Tower) 也有很大的關係。我們把河內塔的圓盤, 由小到大依序編號 $1, 2, 3, \ldots$, 三根柱子的河內塔最佳解是唯一的, 而把每個步驟動的圓盤紀錄下來, 剛好也是 Gros Sequence:

$$1, 2, 1, 3, 1, 2, 1, 4, 1, 2, 1, 3, 1, 2, 1, 5, 1, 2, 1, 3, 1, 2, 1, 4, 1, 2, 1, \ldots$$

也就是說, 奇數步都是動最小的 1 號盤, 偶數步是移動非 1 號盤。因此也是跟九連環一樣的方式, 利用 Gros Sequence 來算出第 $k$ 步是動幾號盤?

**例 10.** 在 $2^n - 1$ 步內, 解出 $n$ 個圓盤的河內塔, 無論 $n$ 是多少(當然數量要足夠), 第 60 步是動幾號盤? 因爲 $60 = 2^2 \cdot 15$, 所以 $g_{60} = 3$, 所以第 60 步是移動 3 號盤。

而解法方面, 因爲偶數步移動非 1 號盤, 就只能把編號第 2 小的盤移動到三者中最大的盤上去, 所以偶數步的移動方法唯一; 奇數步是移動 1 號盤, 有兩種移動方式, 只要能判斷的出來怎麼移, 那解法就容易了。我們可以從 $n = 1, 2, 3, \ldots$ 的流程中, 觀察出 1 號盤的移動, 是環循 (cyclic) 的。Hinz [9] 在 1992 年證明了這件事。由於我們知道河內塔最佳解是 $2^n - 1$ 步, 然後最後一步是把 1 號盤移動到 3 號柱子上, 所以倒推回來就可以知道第一步時, 1 號盤是要移動到第 2 或 3 的柱子上了。

**定理 5.** *$n$ 圓盤的河內塔, 第一步是把 1 號盤移動到第 $2 + (n \mod 2)$ 柱子上。*

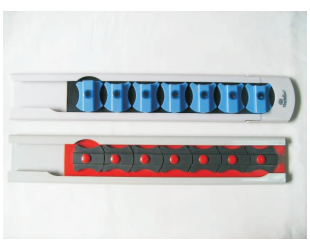
事實上, 在自己試玩了幾次後, 應該也可以整理出這個規則。而且當 $n$ 是偶數後, 1 號盤會依 $2, 3, 1, 2, 3, 1, 2, 3, 1, \ldots$ 順序移動, 當 $n$ 是奇數, 1 號盤是依 $3, 2, 1, 3, 2, 1, 3, 2, 1, \ldots$ 來移動。

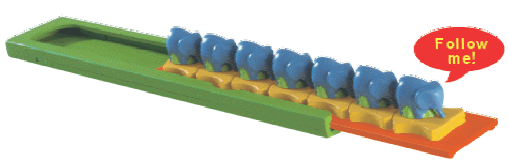
另外 $p$ 根柱子河內塔的研究到現在, 都還非常熱門, 到目前爲止, 有約四百篇的論文在討論它; 但即使如此, 當 $p \geq 4$ 時, 卻還一直無法有個最佳的解法。Klavzar 等人 [10], 證明了目前文獻中提出的五種 $p$ 根柱子河內塔移動策略的方法數, 都是相同的。這些策略中, 最有名的就是 Frame-Stewart algorithm。Korf 與 Felner [11] 利用電腦證明了當 $p = 4$, $n \leq 30$ 時, Frame-Stewart algorithm 是最佳解。Chappelon [12] 則是將 Frame-Stewart algorithm 推廣到更一般的型式。

## 4. 變形與推廣

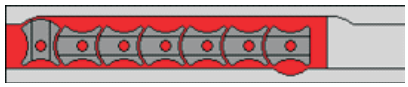
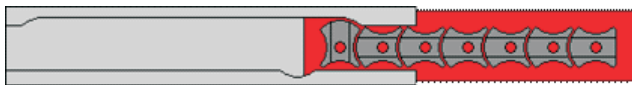
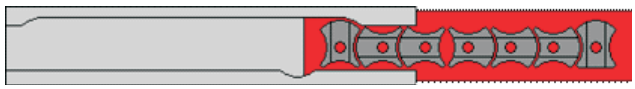
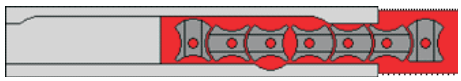
與九連環類似的益智玩具有很多, The Brain (左), 還有中國環 (右), 都與九連環同構。後者在科博館, 有一個非常大型的模型可供操作。

另一個與九連環類似的益智玩具, 叫「大象轉彎」:





「大象轉彎」英文叫「Spin-Out」, 九章出版社翻譯爲「大象扭出來」。其解法相當於 7 連環。每隻大象的狀態  即爲 1, 逆時針轉 90 度時  爲 0, 若順時針 90 度的狀態  會讓整個卡住, 所以基本上是沒有這個狀態的, 詳細的情況需要讀者拿到實物操作會比較了解。大象轉彎與 7 連環一樣, 需要 85 步完成。但值得一提的是, 因爲結構的關係, 事實上可以有個「捷徑」來完成它, 方法如下:(圖形取自[13])

| | |
|---|---|
| 先用正常的方式將前六隻大象轉 90 度。 |  |
| 然後將底座移到最右 |  |
| 將右邊四隻大象, 在外面轉90度, 如圖。 |  |
| 然後再將左邊三隻大象, 在裡面轉90度。 |  |
| 即可讓所有大象抽出。 |  |

所有的大象, 原本都只能在裡面缺口的地方轉動, 但此「捷徑」, 是利用在外面也能轉動的偷吃步, 減少了最後一輪的遞迴, 只用了 $42 + 7 = 49$ 步, 即可解出。

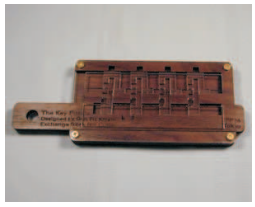
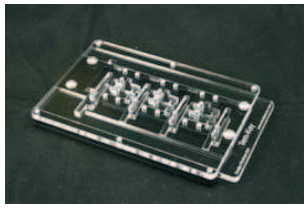
2005年, 德國的 Markus Gotz [14] 設計了「瘋狂象舞 (Crazy Elephant Dance)」, 在第 25 屆國際益智玩具大會 (IPP25) 中, 獲得了「最佳榮譽獎(Honorable Mention)」, 如下圖:

他改進了原始大象轉彎的結構, 讓原本只有兩種狀態的大象, 現在變成三種狀態, 向上、向下與向右; 解的過程不再只有 $R$ 或 $S$ 這兩種狀態而已, 而且若無法利用「三進位格雷碼 (Base-3 Gray Code)」來解的話, 常常會走入死路而出不來。而且 Markus Gotz 設計的結構非常簡單, 就能達到這麼大的改變, 真的很厲害。對結構有興趣, 或是想在線上操作的話, 可以參考 [14] 的連結。

Bill Cutler 在 2003 年設計了名爲 Binary Burr (下圖左) 的神秘盒, 須要用格雷碼的方式才能打開。Goh Pit Khiam [4], 是世界有名的組木 (Burr) 設計專家, 他對格雷碼類型的組木情有獨鍾, 並在 2007 年設計了名爲 Ternary Burr 或稱 Base-3 Gray Code Burr 的神秘盒, 2009 年被 Brian Young 改進 22 塊版本(下圖中), 若不了解三進位格雷碼的話, 解法會非常非常難, 其拆開後如下圖右。

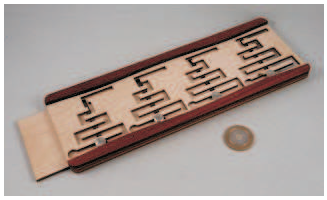


Goh Pit Khiam [4] 另外還設計了格雷碼類型的鎖 (key), 2004 年的 Binary Key (下圖左), 還有 2011 年的 Tern Key (下圖右):

2011年, Jean Claude Constantin 設計了五進位的鎖 Kugellager (下圖左), 需要 1250 步才能解出。並更進一步, 把平面改成立體的型式, 做成鎖的樣子, 變成一把不用鑰鎖的鎖, 名爲 Schloss Dick (下圖右)。

2012年, Jean Claude Constantin 做出了七進位的鎖 Kugellager 7 (如下圖左), 需要 4802 步才能解出, 即使是 Constantin 本人, 也需要 45 分鐘才能解出來。2013 年 Constantin 破紀錄的做出了 15 進位的鎖, 稱爲 Generation Lock (下圖右), 需要約 2.1 億步才能解出, 不眠不休的每秒解 1 步, 也要 6.7 年才解的出來。




## 5. 結語

我們將目前所有格雷碼型式的益智玩具, 整理如下:

| Base | Puzzles |
|---|---|
| 2 | 九連環、大象轉彎、The Brain、Binary Burr、Binary Key、Cubi Barcode Burr、Hexadecimal Puzzle |
| 3 | 瘋狂大象跳舞、Tern Key、Ternary Burr、Super Cubi、K-323、Loony Loop、Sliding Block Puzzle |
| 4 | K-419、MMMDXLVI |
| 5 | Kugellager, Schloss Dick, Mysterians, Cross and Crown |
| 6 | Sliding Lock |
| 7 | Kugellager 7 |
| 11 | Seestern |
| 15 | Generation Lock |

事實上, 依 Constantin 的製作方式, 可以無限延伸, 但解法上, 其實都是類似的, 先找出其狀態圖, 並得知是如何編碼後, 就可以知道怎麼解, 當然也可以算出幾步後是什麼狀態。因此, 解法大致上都是同樣的幾個動作一再的重覆進行, 解出來只是時間上的問題而已; 若設計多幾個位數的話, 解的步數是指數成長, 就算知道怎麼解, 可能花一輩子也解不出來。

能把益智玩具轉換成數學模型, 不但可以減少很多嘗試錯誤的時間, 也可以得到很多漂亮的性質; 但真正厲害的, 是還能把數學性質轉換成實體的物件來操作。Khiam 和 Constantin 等的這些大師, 不但設計了格雷碼型式的益智玩具, 還能推廣到更一般的形式, 非常讓人驚嘆。

國際益智玩具大會 (International Puzzle Party) 每年都會在暑假舉辦, 今年 (2014) 已是第 34 屆, 到底還會有什麼驚豔發明呢? 就讓我們拭目以待吧!

## 參考資料


1. Andreas M. Hinz, Klavzar Sandi, Milutinovic Uros and Petr Ciril, *The Tower of Hanoi - Myths and Maths*, Springer Basel, 2013.
2. Wei Zhang and Peter Rasmussen, *Chinese Puzzles*, Chinese Culture Center of San Francisco, (2008), 26-31.
3. Brian Hayes, Group Theory in the Bedroom, *Hill and Wang*, (2008), 179-200.
4. Goh Pit Khiam, The Design of Mechanical Puzzles Based on the Gray Code, *Cubism For Fun*, 91(2013), 10-15.
5. Andreas M. Hinz, Square-free Tower of Hanoi Sequences, *L'Enseignement Mathématique*,

(2) 42(1996), 257-264.
6. A. Thue, Uber unendliche Zeichenreichen, *Norske Vid. Selsk. Skr., I Mat. Nat. Kl. Christiania* 7 (1906), 1-22.
7. Wesley Pegden, Highly nonrepetitive sequences: winning strategies from the Local Lemma, *Random Structures & Algorithms*, 38(2010), 140-161.
8. Jaroslaw Grytczuk, Piotr Szafruga and Michal Zmarz, Online version of the theorem of Thue, *Information Processing Letters*, Vol. 113, Issues 5-6 (2013), 193-195.
9. Andreas M. Hinz, Pascal's Triangle and the Tower of Hanoi, *The American Mathematical Montyly*, 99(1992), 538-544.
10. Sandi Klavzar, Uros Milutinovic and Ciril Petr, On the Frame-Stewart algorithm for the multi-peg Tower of Hanoi problem, *Discrete Applied Mathematics*, 120(2002), 141-157.
11. Richard E. Korf and Ariel Felner, Recent Progress in Heuristic Search: A Case Study of the Four-Peg Towers of Hanoi Problem, *Proceedings of International Joint Conference on Artificial Intelligence*, 2324-2329.
12. Jonathan Chappelon and Akihiro Matsuura, On generalized Frame-Stewart numbers, *Discrete Mathematics*, 312(2012), 830-836.
13. Jaap's Puzzle Page, Spinout/The Brain/Chinese Ring puzzle, `http://www.jaapsch.net/puzzles/spinout.htm`.
14. Markus Gotz, Crazy Elephant Dance, `http://www.markus-goetz.de/puzzle/0019.html`.

—本文作者任教國立臺灣師範大學數學系—

## 2014 許振榮講座

主　講　人：Nigel Hitchin 教授 (University of Oxford)

日　　期：2014 年 10 月 15 日 (星期三) ～ 2014 年 10 月 17 日 (星期五)

地　　點：台北市大安區羅斯福路四段1號　天文數學館6樓演講廳

詳見中研院數學所網頁 http://www.math.sinica.edu.tw