國 立 交 通 大 學

資訊學院　資訊學程

碩 士 論 文

數獨之最少提示數研究

The Study of Minimum Sudoku

研 究 生：黃裕龍

指導教授：吳毅成 教授

中 華 民 國 九 十 八 年 二 月

數獨之最少提示數研究

The Study of Minimum Sudoku

研 究 生：黃裕龍　　　Student：Yu-Long Huang

指導教授：吳毅成 教授　Advisor：Dr. I-Chen Wu

國 立 交 通 大 學
資訊學院　資訊學程
碩 士 論 文

A Thesis

Submitted to College of Computer Science

National Chiao Tung University

in partial Fulfillment of the Requirements

for the Degree of

Master of Science

in

Computer Science

February 2009

Hsinchu, Taiwan, Republic of China

中華民國九十八年二月

數獨之最少提示數研究

研究生：黃裕龍　　　　　　　　　　　指導教授：吳毅成 教授

國 立 交 通 大 學　　資 訊 學 院　　資 訊 學 程 碩 士 班

摘要

數獨遊戲的風行已經有幾年了，這段時間中陸續有各式解題法被開發出來，其中以候選數演算法最廣為大家所使用。近幾年盤面產生器成了新的研究風潮，其中林順喜教授指導開發了基因重整繁衍法，大量產生最少提示數盤面。

本篇論文研究候選數特性，以集合邏輯的觀念，開發新的解題法，簡化解題的架構及方法，減少解題所需時間；進而改良基因重整繁衍法，使盤面的產生更加快速。最後再根據盤面特徵，快速的比對新產生的盤面是否已重複。

The Study of Minimum Sudoku

Student: Yu-Long Huang Advisor: Dr. I-Chen Wu

Degree Program of Computer Science
National Chiao Tung University

ABSTRACT

Sudoku has been popular for several years around the world. There have been many algorithms built for solving Sudoku, and among them, candidates algorithms remain the most popular nowadays. Also, researchers have long been focusing on the generators of puzzles. For instance, Lin & Lin (2008) proposed the "genes restructuring" algorithm which was able to generate a large amount of Minimum Sudoku puzzles.

This paper aims to study the properties of Sudoku candidates. Then it suggests a new algorithm based on logic and sets principle as well as simplifies the process of solving puzzles and shortens the solving time. In addition, this paper improves the "genes restructuring" algorithm to make the puzzles generating faster. Finally, it checks the newly-generated puzzles to verify if they are duplicated ones.

誌謝

經過了長時間的努力，終於順利地將我的論文完成了。回想寫作論文的時光，有辛苦也有喜悅，最終看到自己完成了一本學術著作，心中感動真是筆墨難以形容。

首先要感謝我的指導教授，吳毅成博士，由於他不厭其煩地細心教導，在學術的這條路上時時指引我正確的方向，這篇論文才得以順利完成。另外，我也要特別感謝兩位口試委員，許舜欽教授以及顏士淨教授，謝謝兩位教授特地從台南及花蓮趕來，針對我的論文提出諸多建議，使得我的論文更加完善。

此外，我也要感謝學弟妹們對於論文內容所提出的各種建議，使得我的論文內容更為豐富。最後，謹以此論文，獻給我最摯愛的家人，感謝我的父母、弟弟、咕唧唧給予我的支持與鼓勵，以及協助我完成論文的北京狗。

目錄

圖目錄

表目錄

第一章、前言

本章第一節介紹數獨遊戲、第二節簡介數獨之最少提示數、第三節指出本論文之研究目的，並在第四節介紹本論文的內容大綱。

1.1. 遊戲介紹

數獨[5]是個與拉丁方陣(Latin Square)[6]有著極大相似度的遊戲，遊戲規則是在一個 9 格寬 x 9 格高的盤面上，填入數字 1~9 使每行、每列及 3 格寬 x 3 格高的小型九宮格，都不能出現相同的數字，遊戲的盤面如圖 1。

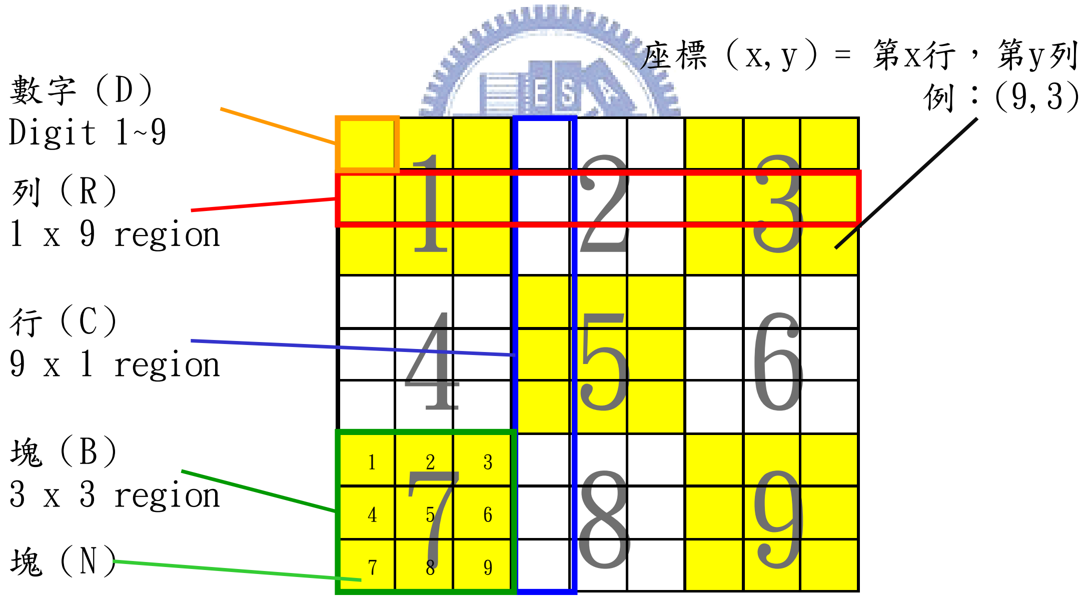


圖 1　數獨盤面

而遊戲的玩法則是在符合數獨規則[5]的前提下，盤面上填入數個數字（稱為提示數）作為題目，接著玩家必須在其他空格填上數字，直到全部填滿為止，如圖 2。相對於數獨題目，填滿的盤面就是這個題目的解。

若題目的解只有一個，稱為「唯一解」；若題目具有二個以上的解，稱

為「多重解」；若這個題目沒有解，稱為「無解」。而一個正式的數獨題目，必須是唯一解。

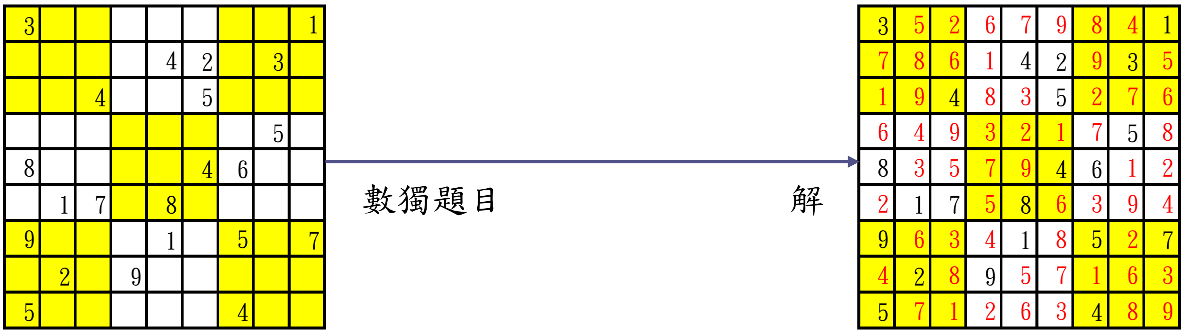


圖 2　數獨題目及其解

1.2.　最少提示數

數獨題目上所出現的數字就是提示數，而提示數的多少也可視為判定題目難易的依據，通常當提示數愈小，題目也相對變得比較難。由於多個數獨題目可以產生相同的解，所以在唯一解的前提下，一個解盤面可以產生多個不同的題目，找尋具有最少提示數的題目，是這幾年新興的研究方向。如圖 3，至今所能找到的最少提示數依舊是 17[4]。

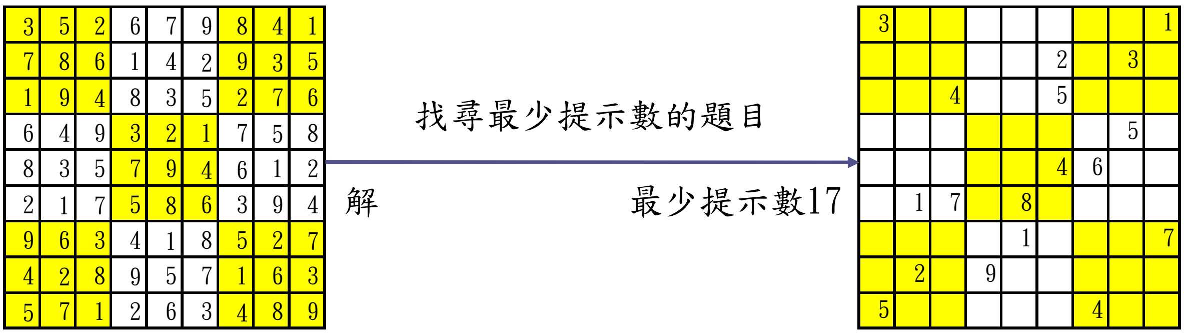


圖 3　最少提示數

鑑於目前所知之最少提示數為 17，且可由提示數 18 之盤面間接找尋提示數為 17 之盤面題目共 64 個[1]，如圖 4，因此推斷假設一個提示數 16 的題目，任選一個空格填入解盤面中相對的數字，就可以產生同樣是唯一

解的提示數 17 題目共 65 個。所以找尋提示數 17 的題目，理論上即可間接找尋提示數 16 的題目。

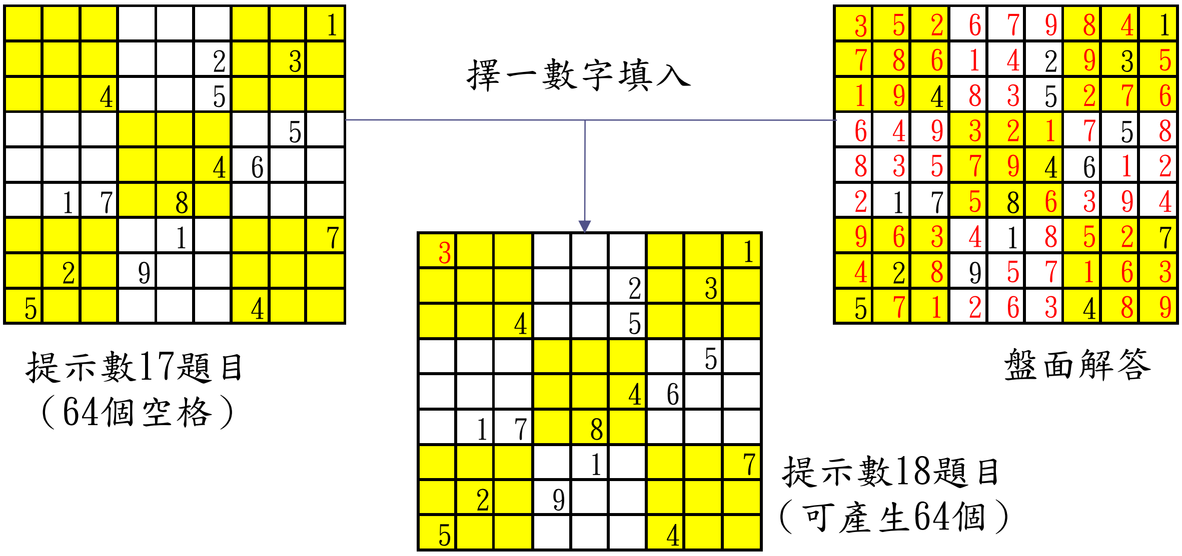


圖 4　由提示數 18 題目，間接找尋提示數 17 題目

1.3.　研究目的

為達成以更快的速度蒐集提示數 17 之數獨題目，本研究之研究目的有二：

a.　提出新的數獨解題法，使其能在更短時間內求解，並驗證是否具備唯一解。

b.　改善以前的基因重整繁衍法[1]，使有效且快速的產生提示數 17 的題目，並驗證是否重複。

整個研究上，「時間」一直是最大的考量重點，所以「簡化演算法、統一解題架構、去除不必要的組合、更快的驗證是否重複」，就成了整個研究的主軸。

1.4.　論文大綱

本篇論文首先於第一章的前言介紹數獨遊戲及確定研究問題。第二章中，

蒐集相關文獻，整理目前在數獨遊戲上的研究，包括一般的數獨解題法、重複數獨盤面的定義、提示數 17 題目產生法等。第三～五章，將依據整個研究的先後順序，一一介紹：第三章會提出一個新的解題法，從邏輯和集合的觀點進行解題，並簡化、改進解題方法；第四章，針對基因演算法提出改進的方式，說明如何去除過半數的排列組合，並完全且快速的檢查新產生盤面是否重複；第五章，會針對實作後的程式，列出實驗結果及分析。最後，第六章會提出結論、待解問題，以及未來可能的發展方向。本研究之流程如圖 5 所示。

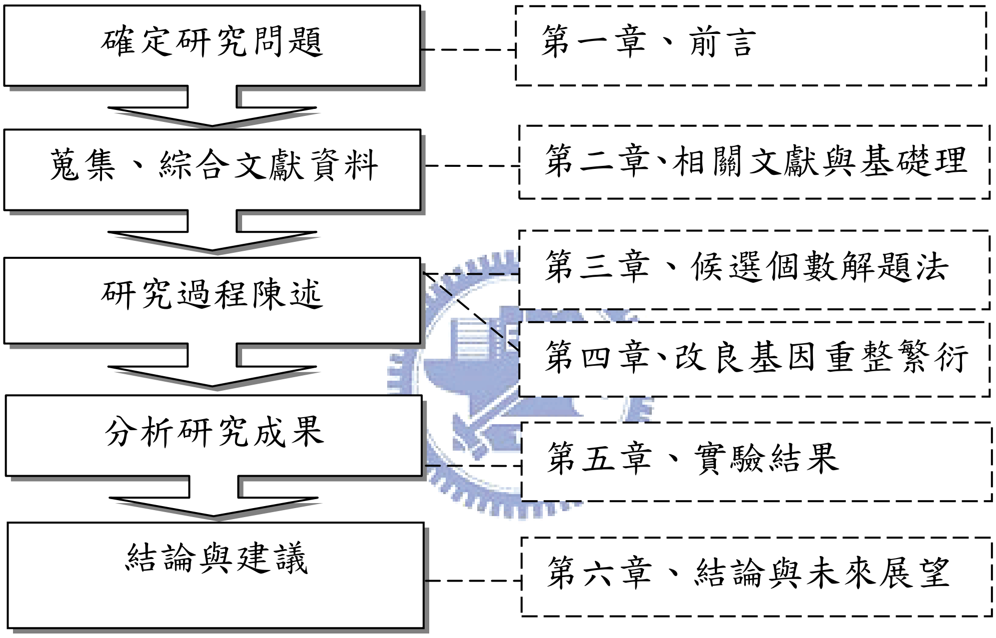


圖 5　研究步驟與章節配置圖

第二章、相關文獻與基礎理論

本章將介紹這篇論文所引用的幾種數獨解題法，並介紹重複盤面的定義。最後，整理目前已知之數獨盤面產生法，並提出需要解決的問題。第 2.1 節將介紹幾種常見的解題法：直觀解題法[8]、候選數解題法[2][5][8]。第 2.2 節將介紹重複盤面的定義[3]，以及所產生的問題。第 2.3 節將介紹幾種盤面產生法：包括傳統之亂數產生法、較有系統之必選集合產生法[7]、及基因重整繁衍法[1]。

2.1. 一般的數獨解題法

數獨的解題方式非常多樣，但大致分類為直觀解題法及候選數解題法兩類。底下將針對這兩類加以介紹。

2.1.1. 直觀解題法

這是人性化的解題法，直接分析盤面已經出現的數字，透過紙筆將答案直接寫下，或是將可能的答案列出，一一推敲而找出解答，如圖 6：

- 將數字 2 的行列劃去，會發現(6,3)只能填入數字 2。
- (2,5)刪去行、列、區塊上出現的數字，只能填 5。

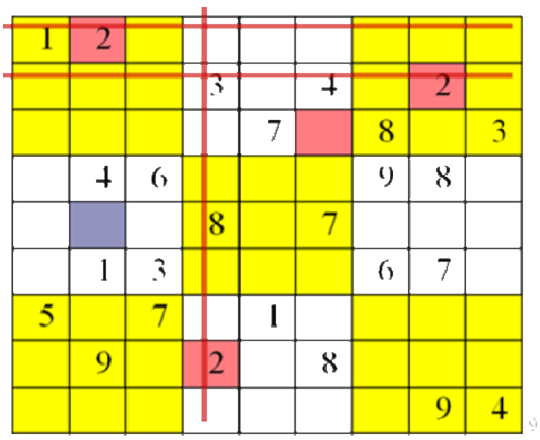


圖 6　直觀解題法

以上只是直觀解題法中簡單的例子，但是在實際上卻有著不容易實作的問題。

2.1.2. 候選數解題法

根據數獨的規則，在空格中填上可能是解答的數字-稱為候選數，再根據候選數做邏輯的分析，填上正確答案，或是刪除錯的候選數，這就是候選數解題法[2][5][8]。目前已知的解題法計有：唯一候選數法（Naked Singles）、數對刪除法（Naked Pair）等共九種。本研究將此九種解題法歸納為四大類，命名為(1) Naked Series、(2) Hidden Series、(3) Locked Series、以及(4) X-Wing Series。在本研究中，為了使用之方便，因此將每項解題法另外命名，如表 1 所示。

表 1 候選數解題法整理

分類	解題法名稱	本研究命名
Naked 系列	唯一候選數法（Naked Singles）	Naked 1
	數對刪除法（Naked Pair）	Naked 2
	三鏈數刪除法（Naked Triplets）	Naked 3
Hidden 系列	隱性唯一候選數（Hidden Singles）	Hidden 1
	隱性數對刪除法（Hidden Pair）	Hidden 2
	隱性三鏈數刪除法（Hidden Triplets）	Hidden 3
Locked 系列	區塊刪除法（Locked Candidates）	Locked
X-Wing 系列	矩形頂點刪除法（X-Wing）	X-Wing 1
	三鏈列刪除法（Swordfish）	X-Wing 2

以下將分類介紹此解題法中常見的方法。其中，本節下列之內容所指的 9 格區域泛指符合數獨規則的一行、一列、3x3 矩陣。

Naked 系列

- **唯一候選數法（Naked 1）**

  若某一格中僅有一個候選數，如圖 7 (a)中的(2,1)，直接填入該候選數 7，若填入其它候選數，會違反數獨的規則。

- **數對刪除法（Naked 2）**

  9 格區域中，若某二格中就只有二種候選數，如圖 7 (b)中的(2,1)及(6,1)，則這二種候選數 7、9 不可能出現在這區域中的其他格子，所以其他格子中的這兩種候選數應該刪除。

- **三鏈數刪除法（Naked 3）**

  9 格區域中，若某三格中共只有三種候選數，如圖 7 (c)中的(2,1)、(5,1)及(6,1)，則這三種候選數 3、7、9 不可能出現在這區域中的其他格子，所以其他格子中的這三種候選數應該刪除。

(a) 唯一候選數法

	1	2	3	4	5	6	7	8	9
1	14 68	7	5~~7~~ 89	12 45	123 456	35 89	12 489	23 489	6~~7~~ 89
2									
3									
4									
5									
6									
7									
8									
9									

(b) 數對刪除法

	1	2	3	4	5	6	7	8	9
1	14 68	79	124 5~~9~~	123 45	12 3~~9~~	79	124 568	234 58	126 ~~7~~8
2									
3									
4									
5									
6									
7									
8									
9									

(c) 三鏈數刪除法

	1	2	3	4	5	6	7	8	9
1	14 68	79	34 5~~9~~	12 458	37	39	124 568	12 ~~3~~8	126 ~~7~~8
2									
3									
4									
5									
6									
7									
8									
9									

圖 7 Naked 系列的範例

Hidden 系列

- **隱性唯一候選數 (Hidden 1)**
  9 格區域中，若某個候選數只出現過一次，如圖 8 (a)中(3,1)的候選數 5，則這個格子中不可能有其他候選數，所以其他候選數應該刪除。
- **隱性數對刪除法 (Hidden 2)**
  9 格區域中，若某二種候選數就只有出現在某二格中，如圖 8(b)中(5,1)及(9,1)的候選數 1、9，則這二格中不可能有其他候選數，所以其他候選數應該刪除。
- **隱性三鏈數刪除法 (Hidden 3)**
  9 格區域中，若某三種數字就只有出現在某三格中，如圖 8(c)中(1,1)、(6,1)及(8,1)的候選數 4、6、8，則這三格中不可能有其他候選數，所以其他候選數應該刪除。

	1	2	3	4	5	6	7	8	9
1	246 789	34 67	~~23~~ 5~~7~~	23 89	134 689	123 68	23 789	34 67	12 789
2									
3									
4									
5									
6									
7									
8									
9									

(a) 隱性唯一候選數

	1	2	3	4	5	6	7	8	9
1	246 78	34 67	23 57	235 67	1~~3~~ 49	23 678	234 78	34 67	1~~2~~ ~~78~~9
2									
3									
4									
5									
6									
7									
8									
9									

(b) 隱性數對刪除法

	1	2	3	4	5	6	7	8	9
1	~~2~~4 ~~7~~8	13 79	125 79	23 57	123 59	~~12~~ 68	23 57	~~12~~ 46	12 39
2									
3									
4									
5									
6									
7									
8									
9									

(c) 隱性三鏈數刪除法

圖 8 Hidden 系列的範例

Locked 系列

- **區塊刪除法（Locked）**
  當任兩個 9 格區域發生交集時，若其中一個區域中的某個候選數，只出現在交集的小區域上，如圖 9 中左方縱行的候選數 7，只出現在左上方 1x3 的交集區域中，因此上方 3x3 區域中的其他格子內，不可能有這個候選數，應該要刪除。

	1	2	3
1	12 48	124 6~~7~~	124 68
2	12 47	9	23 6~~7~~
3	124 78	345 6~~7~~	345 68
4	14 89		
5	125 89		
6	123 89		
7	6		
8	145 89		
9	14 89		

圖 9　Locked 系列的範例

X-Wing 系列

- **矩形頂點刪除法（X-Wing 1）**
  若某個候選數，出現在某兩橫列中相同的縱行上，如圖 10 (a)頂邊及底邊的候選數 8，則在這二縱行上的其他格子中，不可能有這個候選數，應該要刪除。

- **三鏈列刪除法（X-Wing 2）**
  若某個候選數，出現在某三橫列中相同的縱行上，如圖 10 (b)頂邊及底邊的候選數 8，則在這三縱行上的其他格子中，不可能有這個候選數，應該要刪除。

	1	2	3	4	5	6	7	8	9
1	4	136	56	2	136 8	7	13 56	1356 8	9
2	13 59	8	256	13 49	13 469	34 69	123 456	7	13 46
3	13 79	123 69	267	13 489	134 689	5	123 46	123 468	13 468
4	6	34 79	457	134 579	2	349	8	13 45	13 47
5	35 789	234 79	245 78	134 5789	1345 6789	346 89	134 569	134 56	13 467
6	35 789	34 79	1	345 789	3456 789	346 89	345 69	34 56	2
7	18	146	3	45 89	45 89	24 89	7	124 68	146 8
8	178	5	46 78	34 78	34 78	23 48	123 46	9	134 68
9	2	47	9	6	34 78	1	34	348	5

(a) 矩形頂點刪除法

	1	2	3	4	5	6	7	8	9
1	234	1	5	246	246 8	7	346	346 8	9
2	49	489	6	14 59	145 89	3	145	7	2
3	234 79	234 789	234 78	124 569	124 5689	245 689	134 56	134 568	134 68
4	1234 5679	234 679	123 47	1234 5679	124 5679	245 69	8	123 456	134 67
5	1234 5679	234 679	123 47	8	124 5679	245 69	1234 5679	123 456	134 67
6	8	234 679	123 47	1234 5679	124 5679	245 69	1234 5679	123 456	134 67
7	123 467	234 678	245 679	245 679	245 6789	245 689	123 467	123 468	134 678
8	123 467	5	123 478	24 67	246 78	24 68	123 467	9	134 678
9	24 67	2467 8	9	24 67	3	1	24 67	246 8	5

(b) 三鏈列刪除法

圖 10　X-Wing 系列的範例

這些解題法在實作上一個比一個更加複雜，而且不同的解題法可能會應用到不同的架構，所以整個盤面的資料還必須在不同的解題法間轉換或同步，這些動作都會增加解題所需的時間以及困難度。

2.2.　重複數獨盤面的定義

重複數獨盤面是指將行、列、數字，予以對調轉換而產生的盤面。透過這個定義，可以將數獨的解盤面共 6,670,903,752,021,072,936,960 個，扣除不需要重複考慮的數獨盤面，減少至 5,472,730,538 個[3]。同理，在產生盤面並檢查是否重複時，透過這層定義，可以縮小需要考慮的重複盤面數量。

只要透過表 2 所指定的六個方式，將數獨盤面的行、列、數字，予以對調轉換，所能產生的盤面都是重複盤面，文後之附錄 A 列出重複盤面的範例。

表 2　重複盤面之對調方式

對調方式	組合個數
行列對調	2
列 1/2/3 相互對調 列 4/5/6 相互對調 列 7/8/9 相互對調	$(3!)^3$
行 1/2/3 相互對調 行 4/5/6 相互對調 行 7/8/9 相互對調	$(3!)^3$
三大區域相互對調 列 1~3、列 4~6、列 7~9	3!
三大區域相互對調 行 1~3、行 4~6、行 7~9	3!
數字 1~9 相互對調	9!

總結以上所有的方式，總計可以產生 $9! \times (3!)^6 \times 2$ 共 1218998108160 種組合。如果要比對某個新產生的盤面是否重複，就必須考慮這麼多可能的盤面，若以每千分之 1 秒的速度來比對其中一種可能，全部比對完畢約需要 38 年的時間。

2.3.　提示數 17 題目產生法

自數獨解題法引起研究的熱潮後，緊接著就是題目產生法，其中也有方法是利用重複盤面的定義，產生不同的盤面。最古老的方式為亂數產生法。此產生法首先以現有的提示數 17 盤面，亂數隨機產生新盤面，隨後驗證盤面是否符合數獨規則，以及驗證是否此盤面有唯一解。這種方法若用在提示數 17 問題的產生上，總是產生牴觸規則或是多重解的盤面，效率非常差。因此，本節將介紹幾款成功率較高之題目產生法，說明如何較有效率地產生全新的提示數 17 盤面，並驗證是否符合數獨規則。

2.3.1. 必選集合產生法

所謂的必選集合(Unavoidable Set)[7]，就是在解的盤面上之具有連鎖反應的組合。如果這個組合中所有的數字都不是提示數字，一定會產生多重解，如圖 11。若標示的六個位置都沒有提示數字，這個題目就至少有兩個解。

7	6	4	3	2	1	9	8	5
3	2	1	9	8	5	7	6	4
9	8	5	7	6	4	2	3	1
6	4	7	2	1	3	8	5	9
2	1	3	8	5	9	6	4	7
8	5	9	6	4	7	1	2	3
4	7	6	1	3	2	5	9	8
5	9	2	4	7	8	3	1	6
1	3	8	5	9	6	4	7	2

7	6	4	3	2	1	9	8	5
3	2	1	9	8	5	7	6	4
9	8	5	7	6	4	3	2	1
6	4	7	2	1	3	8	5	9
2	1	3	8	5	9	6	4	7
8	5	9	6	4	7	2	1	3
4	7	6	1	3	2	5	9	8
5	9	2	4	7	8	1	3	6
1	3	8	5	9	6	4	7	2

圖 11 必選集合

一個填滿數字的數獨解盤面，都可以找到很多這樣的必選集合，為了形成唯一解，就要設法解除這種必選集合。方法是從這些必選集合中，選擇一個數字使成為提示數字。如圖 12，將必選集合中的其中一個候選數 3，改為提示數字，其他五個空格就只剩下一種可能。

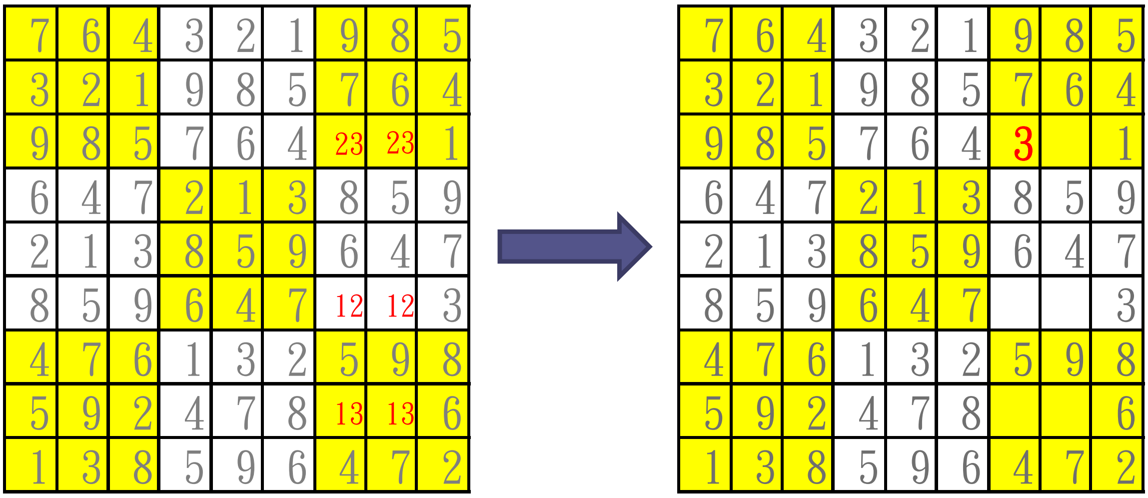


圖 12　解除必選集合

應用這個方式，用以下幾個程序，試圖產生最少提示數盤面，如圖 13。由於可能的組合不只一個，所以反覆進行這個程序，直至所有的組合都試完為止。

a.　從最大的必選集合選一個數字，或選最多必選集合交集的數字。
b.　選擇最少的數字 (例：17 個)，以解除最多的必選集合。
c.　最後以 Sudoku Solver[7]驗證是否唯一。

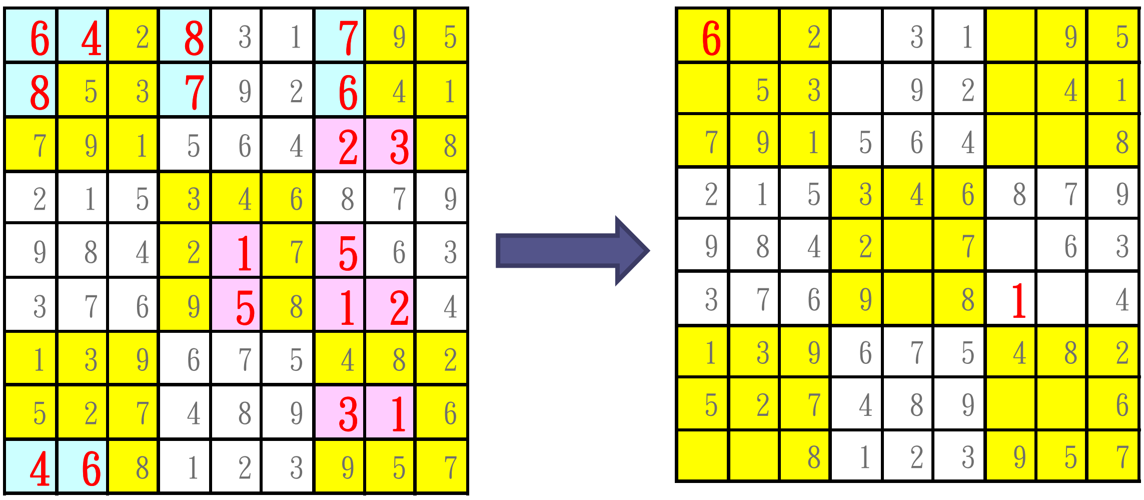


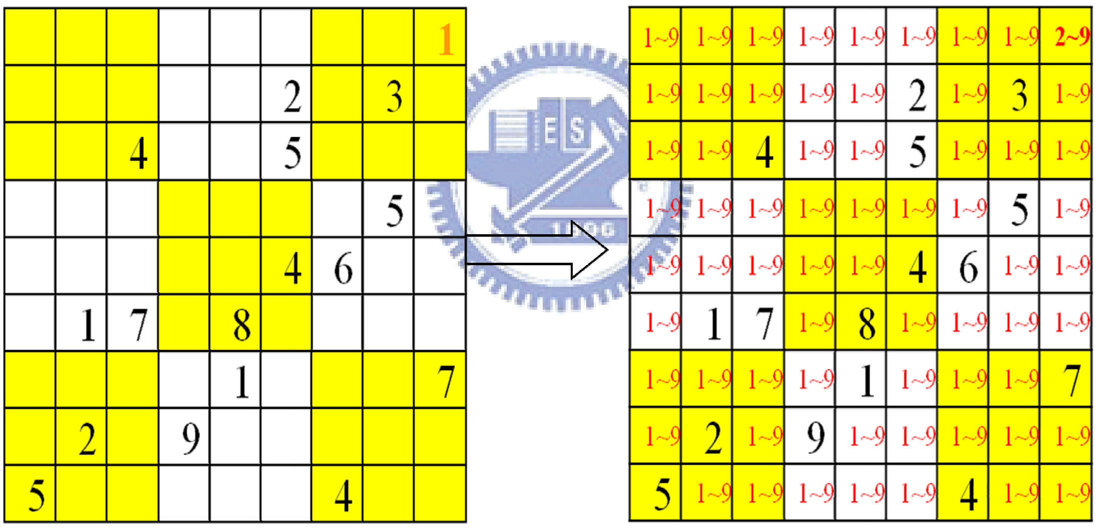
圖 13　必選集合產生法

2.3.2.　基因重整繁衍法

此演算法[1]利用基因繁衍的特性，經由本研究整理、綜合演算過程，可歸納為三個主要步驟：(1)繁衍(以現有的提示數 17 盤面為母盤)、(2)突變(改變一小部份提示數字)、以及(3)適者生存(驗證是否唯一解、是否重複)。

其中，突變是此演算法的核心，相較於亂數產生法，它僅更改母盤的一小部份提示數字，所以比較不容牴觸數獨規則，也比較可能產生具有唯一解的題目。以下分成 2 個動作解釋突變的方式（假定改變的提示數字個數為 1），以此方式共可以產生 $C_1^{17}(C_1^{64}9^1+C_1^{1}8^1) = 9928$ 種組合。

a. 從現有的 17 個提示數字中，選擇 1 個予以移除，如圖 14 (a)。
b. 在剩下的 65 個空格中，填入新的提示數字，如圖 14 (b)。



(a) (b)

圖 14 基因重整繁衍法

最後動作，驗證是否唯一解，及是否不重複之步驟，該篇論文以候選數解題法檢驗是否唯一解。至於盤面是否重複，因為重複盤面的數量相當龐大，所以只根據「數字 1~9 相互對調」產生對應的重複盤面，再與資料庫中的盤面一一進行比對。經由一個月的時間繁衍出八代新生子，總計獲得超過 20 萬筆新盤面。然而，本研究卻認為此演算法有兩個問題不容忽視：

- 經實驗發現，此演算法每次產生的 9928 個待驗棋盤中，約 2/3 牴觸規則，1/3 有多重解，產生的唯一解盤面個數平均只有 2 個。
- 根據重複盤面[3]之定義，任一盤面都有 1218998108160 種可能，但「數字 1～9 相互對調」僅檢查出 9!，共 362880 種重複盤面，由此可發現，此方式無法確實刪除重複盤面。

綜上所述，本章首先整理數獨常用之候選數解題法、並定義重複盤面、最後介紹基因重整繁衍法及提出其問題。本研究將綜合這些文獻，以候選數解題法為基礎，提出新的解題法；進一步改良基因重整繁衍法，改善新生盤面的產生效率，並提出一個可以嚴謹且快速地檢查新生盤面是否重複的方法。

第三章、候選個數解題法

本研究綜合文獻，提出新的數獨解題方法，命名為「候選個數解題法」。本章第一節先定義所使用到之數學符號及算式；第二節介紹演算法內容；第三節提出一些實作的技巧用以提昇效率。

3.1. 定義

候選個數解題法主要是建構在集合與邏輯分析上，所以先定義所有使用到的元素。以候選數出現的位置及候選數本身作為元素，參考圖 1 定義如下：

- C 為**行**的集合，集合元素有 c1~c9。
- R 為**列**的集合，集合元素有 r1~r9。
- D 為**候選數**的集合，集合元素有 d1~d9。
- B 為 **3x3 區塊**的集合，集合元素有 b1~b9。
- N 為 **3x3 區塊中每一小格位置**的集合，集合元素有 n1~n9。

以圖 15 為例，第一列上的每一行上都有候選數，所以可以得到行的集合 C = {c1, c2, c3, c4, c5, c6, c7, c8, c9}及列的集合 R = {r1}，而出現的候選數從 1 到 9 都有出現，因此可以得到候選數的集合 D = {d1, d2, d3, d4, d5, d6, d7, d8, d9}。

	1	2	3	4	5	6	7	8	9
1	14 68	7	5~~7~~ 89	12 45	123 456	35 89	12 489	23 489	6~~7~~ 89
2									
3									
4									
5									
6									
7									
8									
9									

圖 15 Naked 1 範例

在數獨棋盤上，每個元素之間有關係存在，以下定義三個基本關係式：

- D(c, r)={d1, d2, …} 代表 行 c，列 r，對應到候選數 d 的集合。
- C(r, d)={c1, c2, …} 代表 列 r，候選數 d，對應到行 c 的集合。
- R(c, d)={r1, r2, …} 代表 行 c，候選數 d，對應到列 r 的集合。

以圖 15 為例，第 3 行第 1 列上有候選數 5、7、8、9，這句話可以表示為 D(c3, r1) = {d5, d7, d8, d9}。第 1 列上候選數 7 出現在第 2、3、9 行，這句話可以表示為 C(r1, d7) = {c2, c3, c9}。第 3 行上候選數 7 出現在第 1 列，這句話可以表示為 R(c3, d7) = {r1}。

由於數獨規則都是定義一個 9 格區域中所出現的候選數種類，可以視為多個位置的候選數聯集關係，因此本研究根據集合關係式，延伸定義以下聯集關係式：

- $CollectD(C,R)=\bigcup_{\forall c\in C}\bigcup_{\forall r\in R} D(c,r)$
- $CollectC(R,D)=\bigcup_{\forall r\in R}\bigcup_{\forall d\in D} C(r,d)$
- $CollectR(C,D)=\bigcup_{\forall c\in C}\bigcup_{\forall d\in D} C(c,d)$

以圖 15 為例，第一列的第 1~3 行，共出現候選數 1、4～9，這樣的結果也可以由公式推導出來，例如：CollectD({c1, c2, c3}), {r1}) = D(c1, r1) ∪ D(c2, r1) ∪ D(c3, r1) = {d1, d3, d6, d8} ∪ {d7} ∪ {d5, d7, d8, d9} = {d1, d4, d5, d6, d7, d9}

本研究中定義 |F(x)| 代表 F(x)這個集合內的元素個數。以圖 15 為例，第 3 行第 1 列有候選數 5、7、8、9 共 4 個，這句話可表示為|D(c3, r1)| = |{d5, d7, d8, d9}| = 4。依此類推，

- |C(r1, d7)| = |{c2, c3, c9}| = 3
- |R(c3, d7)| = |{r1}| = 1
- |CollectD({c1, c2, c3}, {r1})| = |{d1, d4, d5, d6, d7, d8, d9}| = 7

3.2. 解題法

這個章節將藉上一節所做的定義，依先前三大分類依序介紹，分別為：Naked 系列、Hidden 系列、X-Wing 系列。

3.2.1. Naked 系列

根據先前集合的定義，如果 CollectD(C,R)=D 且 |C|=1，|R|=|D|=w ，這就是 Naked w。同理，如果 CollectD(C,R)=D 且 |R|=1，|C|=|D|=w ；CollectD(B,N)=D 且 |B|=1，|N|=|D|=w ，均為 Naked w。以下一一舉例驗證。

- **Naked 1 範例 (w=1)**
  以第 7 頁的圖 7(a)為例，在第 1 列第 2 行的位置上，僅有一個候選數 7，符合條件 CollectD({c2},{r1}) = {d7} 且 |{r1}| = 1，|{c2}| = |{d7}| = 1。

- **Naked 2 範例 (w=2)**
  以第 7 頁的圖 7 (b)為例，第 1 列第 2、6 行的位置上，僅有二個候選數 7、9，符合條件 CollectD({c2,c6},{r1}) = {d7,d9} 且 |{r1}| = 1，|{c2,c6}| = |{d7,d9}| = 2。

- **Naked 3 範例 (w=3)**
  以第 7 頁的圖 7(c)為例，第 1 列第 2、5、6 行的位置上，僅有三個候選數 3、7、9，符合條件 CollectD({c2,c5,c6},{r1}) = {d3,d7,d9} 且 |{r1}|= 1，|{c2,c5,c6}| = |{d3,d7,d9}| = 3。

3.2.2. Hidden 系列

同理，根據集合定義，如果 CollectC(R,D)=C 且 |R|=1，|D|=|C|=w ；CollectR(C,D)=R 且 |C|=1，|D|=|R|=w ； CollectN(B,D)=N 且 |B|=1，|D|=|N|=w ，均為 Hidden w。以下一一舉例驗證。

- **Hidden 1 範例 (w=1)**

  以第 8 頁的圖 8 (a)為例，第 1 列上的候選數 5，僅出現在第 3 行的位置上，符合條件 CollectC({r1},{d5}) = {c3} 且 |{r1}| = 1，|{d5}| = |{c3}| = 1。

- **Hidden 2 範例 (w=2)**

  以第 8 頁的圖 8 (b)為例，第 1 列上的候選數 1、9，僅出現在第 5、9 行的位置上，符合條件 CollectC({r1},{d1,d9}) = {c5,c9} 且 |{r1}| = 1，|{d1,d9}| = |{c5,c9}| = 2。

- **Hidden 3 範例 (w=3)**

  以第 8 頁的圖 8(c)為例，第 1 列上的候選數 4、6、8，僅出現在第 1、6、8 行的位置上，符合條件 CollectC({r1},{d4,d6,d8}) = {c1,c6,c8} 且 |{r1}| = 1，|{d4,d6,d8}| = |{c1,c6,c8}| = 3。

3.2.3. X-Wing 系列

同理，根據集合定義，如果 CollectC(R,D)=C 且 |D|= 1，|R|=|C|=w ；CollectR(C,D)=R 且 |D|= 1，|C|=|R|=w ，均為 X-Wing w。其中，當 w=1 時可直接套用 Naked 1 或是 Hidden 1，除此之外，以下一一舉例驗證。

- **X-Wing 2 範例 (w=2)**

  以第 10 頁的圖 10 (a)為例，第 1、9 列上的候選數 8，僅出現在第 5、8 行的位置上，符合條件 CollectC({r1,r9},{d8}) = {c5,c8} 且 |{d8}| = 1，|{r1,r9}| = |{c5,c8}| = 2。

- **X-Wing 3 範例 (w=3)**

  以第 10 頁的圖 10 (b)為例，第 1、2、9 列上的候選數 8，僅出現在第 2、5、8 行的位置上，符合條件 CollectC({r1,r2,r9},{d8}) = {c2,c5,c8} 且 |{d8}| = 1，|{r1,r2,r9}| = |{c2,c5,c8}| = 3。

3.3. 實作方法

解說完候選個數解題法後，下一個問題就是該如何實作它？藉由集合的定義，以位元（bits mask）代表集合的元素，集合的聯集可以透過位元的 OR 運算來達成，以下介紹實作的方法及技巧。

- 以一個 16 位元代表一格中的候選數，位元 m 對應候選數(m+1)，當計算多個格子的聯集時，只需要將這些變數做 OR 運算即可。例如候選數 2、7、9 聯集候選數 1、2，對應到位元運算，可以表示為 0x142 OR 0x3 = 0x143，可以再對應回候選數 1、2、7、9。

- 9 個候選數有 512 種組合，以陣列存放這些組合的候選個數，當需要計算候選個數時，就可以直接查表得知，如圖 16 (a)。

- 搜尋 Naked N 時，只需要搜尋候選個數小於等於 N 的格子，如圖 16 (b)：當 N=2 時，需要考慮的對象只有第 1、3、4 格。

- OR 運算的過程中，若個數已經超過 N，則中斷放棄這個組合，如圖 16 (c)：當 N=3 時，考慮對象為第 3、6、9 時，當計算到第 6 格時，候選個數早已超過 3，這時就可以停止這個組合的運算。

候選數	1	2	1,2	3	1,3	2,3	1,2,3	4	1,4
16位元值	0x1	0x2	0x3	0x4	0x5	0x6	0x7	0x8	0x9
候選個數	1	1	2	1	2	2	3	1	2

(a)

1	1 7 8 9	2 3	2 3	2 3 7 8 9	4 5 6	4 5 6	4 5 6	4 5 6 7 8 9

(b)

1	1 7 8 9	2 3	2 3	2 3 7 8 9	4 5 6	4 5 6	4 5 6	4 5 6 7 8 9

(c)

圖 16 Naked N 小技巧

接下來以兩個例子，說明如何透過位元運算的觀點，快速解題。以圖 17 (a)上方的 Hidden 3 為例，將第 1 列上候選數 4、6、7 所在位置的行，分別以二進制的 1 來表示，所得到的三組 Bits Mask 再透過 OR 運算，就可以獲得同樣具有三個 1 的 Bits Mask，以這樣的方式就可以找出 Hidden 3。

同理，以圖 17 (b)上方的 X-Wing 3 為例，將第 1、2、9 列候選數 8 所在位置的行，分別以二進制的 1 來表示，所得到的三組 Bits Mask 再透過 OR 運算，就可以獲得同樣具有三個 1 的 Bits Mask，以這樣的方式就可以找出 X-Wing 3。

	1	2	3	4	5	6	7	8	9
1	~~2~~4 ~~7~~8	13 79	125 79	23 57	123 59	~~12~~ 68	23 57	~~12~~ 46	12 39

	c1	c2	c3	c4	c5	c6	c7	c8	c9
d4	1	0	0	0	0	0	0	1	0
d6	0	0	0	0	0	1	0	1	0
d8	1	0	0	0	0	1	0	0	0
OR	1	0	0	0	0	1	0	1	0

(a) Hidden 3

	1	2	3	4	5	6	7	8	9
1	234	1	5	246	246 8	7	346	346 8	9
2	49	489	6	14 59	145 89	3	145	7	2
3									
4									
5									
6									
7									
8									
9	24 67	2467 8	9	24 67	3	1	24 67	246 8	5

	c1	c2	c3	c4	c5	c6	c7	c8	c9
r1	0	0	0	0	1	0	0	1	0
r2	0	1	0	0	1	0	0	0	0
r9	0	1	0	0	0	0	0	1	0
OR	0	1	0	0	1	0	0	1	0

(b) X-Wing 3

圖 17 Bits Mask 實作範例

第四章、改良基因重整繁衍法

為了改善之前基因重整繁衍法[1]的缺點，避免產生過多與規則衝突、無解、多重解的盤面，本章將利用候選數提出改良方式，步驟有(1)排除規則抵觸盤面、(2)排除多重解盤面、(3)藉由尋找盤面的特徵，找出重複盤面的代表，加速重複盤面的檢查。

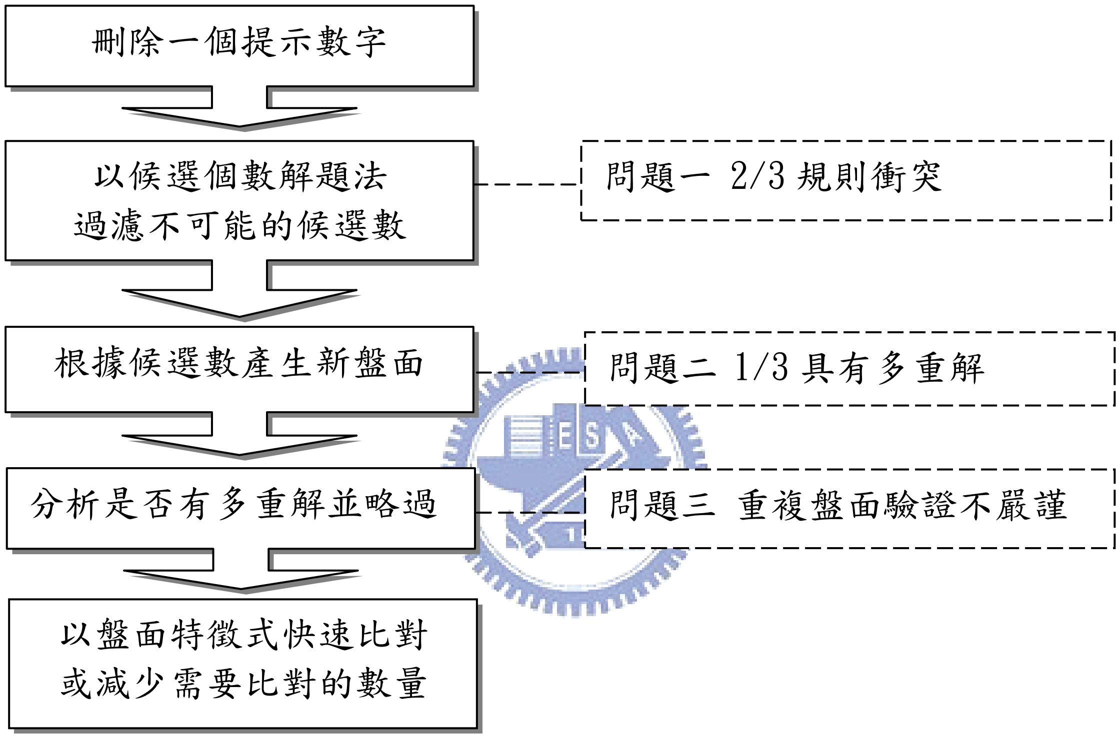


圖 18 改良基因重整繁衍法

4.1. 排除規則抵觸盤面

本研究先以候選個數解題法排除不可能的候選數，僅留下可能的候選數，如圖 19。經過解題法的分析後，大約會剩下 1/3 的候選數，再使用迴圈逐一測試每種可能的組合。

1~9	1~9	1~9	1~9	1~9	1~9	1~9	1~9	2~9
1~9	1~9	1~9	1~9	1~9	2	1~9	3	1~9
1~9	1~9	4	1~9	1~9	5	1~9	1~9	1~9
1~9	1~9	1~9	1~9	1~9	1~9	1~9	5	1~9
1~9	1~9	1~9	1~9	1~9	4	6	1~9	1~9
1~9	1	7	1~9	8	1~9	1~9	1~9	1~9
1~9	1~9	1~9	1~9	1	1~9	1~9	1~9	7
1~9	2	1~9	9	1~9	1~9	1~9	1~9	1~9
5	1~9	1~9	1~9	1~9	1~9	4	1~9	1~9

123 6789	356 789	123 5689	134 678	34 679	136 789	125 789	124 6789	245 689
16 789	56 789	15 689	14 678	46 79	2	15 789	3	145 689
123 6789	36 789	4	13 678	36 79	5	12 789	126 789	12 689
234 689	34 689	23 689	12 367	23 679	13 679	123 789	5	123 789
23 89	35 89	23 589	12 357	23 579	4	6	12 789	12 389
23 469	1	7	23 56	8	369	239	249	23 49
34 689	34 689	36 89	234 568	1	368	23 589	26 89	7
134 678	2	13 68	9	34 567	36 78	13 58	168	13 568
5	36 789	13 689	23 678	23 67	36 78	4	12 689	123 689

圖 19　排除規則牴觸盤面

4.2.　排除多重解盤面

經過解題法刪除不可能的候選數後，還有 1/3 的可能候選數，但幾乎都是多重解。此步驟主要目的是略過在解題過程中出現多重解的組合。以下會列出幾個多重解的範例，一旦發現這樣的盤面就直接略過，省去驗證的時間。

範例 1

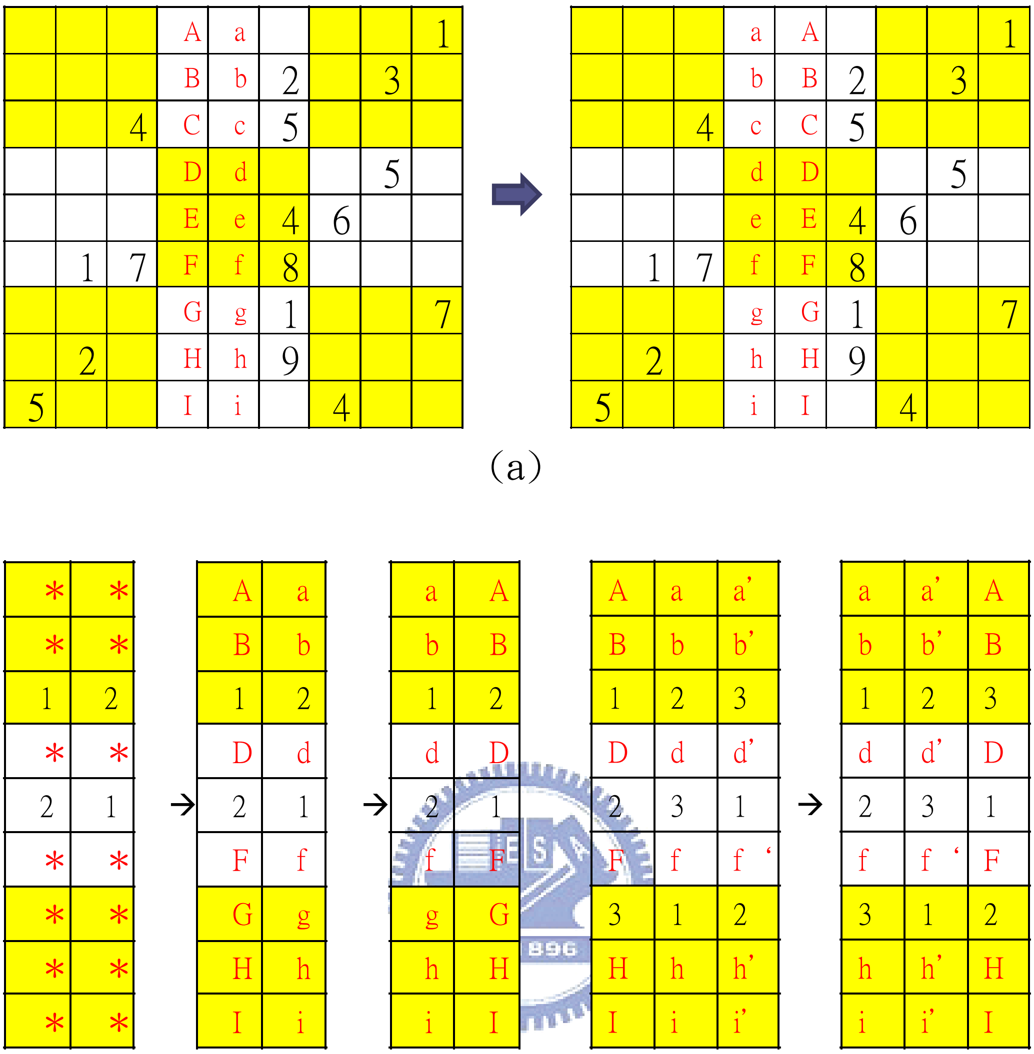
在 3x9 的行列區域中，有兩個以上的 1x9 區域，完全沒有數字，如圖 20 (a)。由於範例中的這兩行完全沒有提示數字，又可以相互對調，所以不論解為何，只要屆時將這兩行對調，又是另一個新的解。

範例 2

在 3x9 的行列區域中，兩個 1x9 區域中候選數完全對稱，如圖 20 (b)。這個例子與範例 1 很相似，雖然不是兩行完全對調，但只要將這兩行中，扣除提示數字的部份對調，也可以得到另一個解。

範例 3

在 3x9 的行列區域中，三個 1x9 區域中候選數完全對稱，如圖 20 (c)。這個例子與範例 2 相同，只要任兩行的候選數完全對稱，就有多重解存在。



(a)　　(b)　　(c)

圖 20　多重解範例

4.3. 驗證重複盤面

由於任一數獨盤面擁有 1218998108160 個重複盤面[3]，本章節會提出盤面特徵式來代表這個盤面的特徵，即使是重複但不完全相同的兩個盤面，也必須獲得相同的特徵式。以下依據取得特徵式的步驟順序，一一加以介紹。

盤面特徵式

對於一個提示數 17 的盤面，以每種數字的出現個數，及每行、每列上

的數字個數，共計 3x9 = 27 個數字作為特徵式，如圖 21 上方的數字列，共分為 Digit Pattern、Column Pattern、Row Pattern。由於提示數字有 17 個，所以這三個 Pattern 內的數字和都是 17。

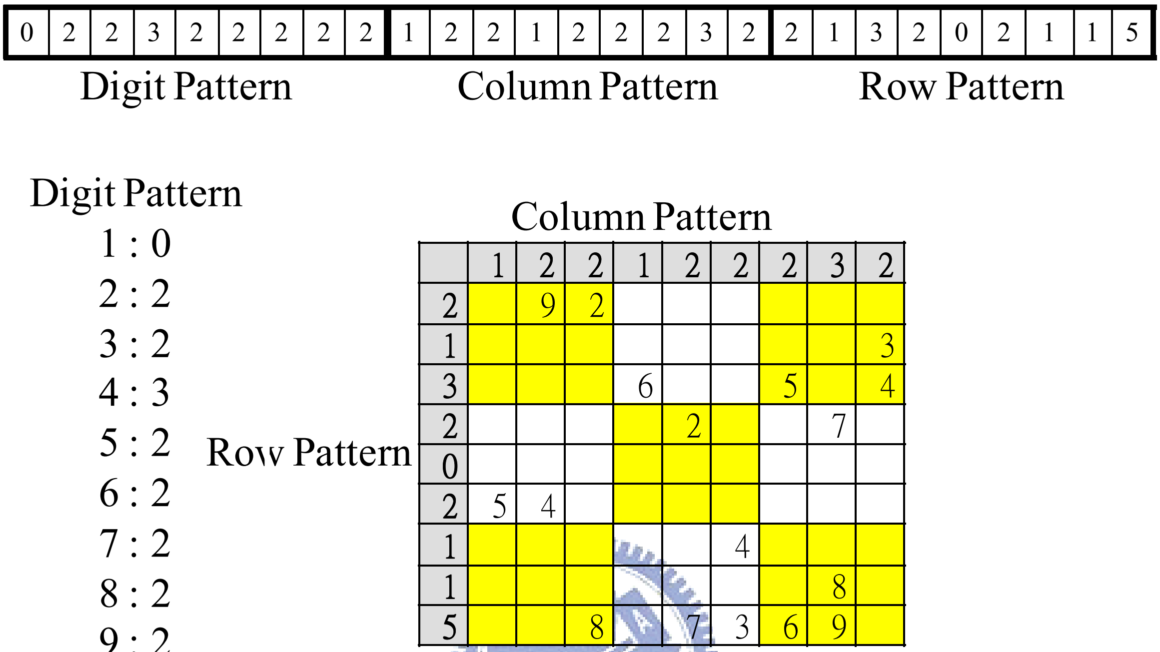


圖 21 盤面特徵式

根據重複盤面的定義，當發生區域對調時，對特徵式的影響僅順序上發生改變，如圖 22。利用這個特性排序特徵式（由左至右，從大到小），作為這個盤面的代表。先以這個特徵式與其他盤面的特徵式進行比對，如果不相同，就表示這個盤面是新盤面。

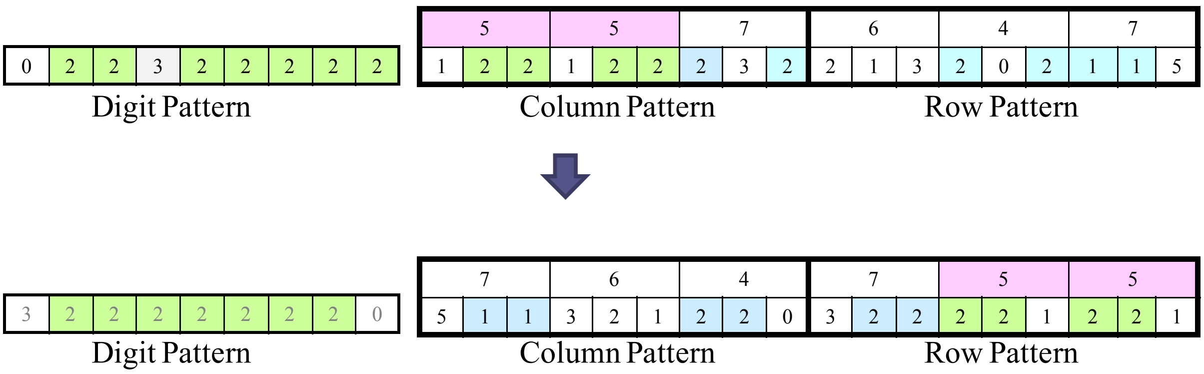


圖 22 具有代表性的特徵式

如果特徵式相同，就根據內容找出所有符合特徵式的盤面，再進行比對。以圖 23 為例，在不影響特徵式的前提下，建立如表 3 的巢狀迴圈來執行相互對調，進而產生 7! x $(2!)^6$ ，共 322560 個重複盤面。

5040 = 1! * (1 * 5040 * 1)								
1	5040 = 7!							1
3	2	2	2	2	2	2	2	0

Digit Pattern

64 = 1! * (4 * 16)																	
4 = 1! * (2 * 1 * 2)									16 = 2! * (2 * 2 * 2)								
2 = 2!			1 = 1!			2 = 2!			2 = 2!			2 = 2!			2 = 2!		
5	1	1	3	2	1	2	2	0	3	2	2	2	2	1	2	2	1

Column Pattern　　Row Pattern

圖 23　以特徵式產生迴圈

表 3　迴圈產生範例

迴圈	相互對調動作	組合個數
Loop 1	數字 2~8 相互對調	7!
Loop 2	行 2 / 3 相互對調	2!
Loop 3	行 7 / 8 相互對調	2!
Loop 4	列 2 / 3 相互對調	2!
Loop 5	列 4 / 5 相互對調	2!
Loop 6	列 7 / 8 相互對調	2!
Loop 7	列 456 / 列 789 相互對調	2!

第五章、實驗結果

本研究之實驗環境軟、硬體規格如下：

表 4　系統規格

項次	規格名稱	規格描述
1	程式開發語言	Visual C++ 6.0
2	作業系統	Windows XP Professional (Service Pack 2)
3	主機 CPU	Pentium-4 (3GHz)
4	記憶體 RAM	512 MB RAM
5	硬碟容量	80 GB
6	測試盤面	36628個提示數17盤面
7	取得來源	2006年，取自於Gordon的Minimum Sudoku網頁

以 36628 個提示數 17 盤面為測試對象，及根據一般候選數解題法所建立的解題器，全部解題後產生如表 5(a)的數據。再根據候選個數解題法，產生相對的集合架構，並搭配位元運算的技巧，最後產生如表 5 (b)的數據。其中。大部分的解題法都集中在 Naked 1 及 Hidden 1，使用率上差異不大，但受到新的架構及位元運算的影響，整體的計算速度仍有提昇。由於其中 Naked 3 出現比率很低且耗時，所以如果略過這個解題程序，平均解題時間會從原本的 0.767 ms 增快至 0.576 ms。

表 5　候選數解題法 比較表

項目名稱	項目描述
1. 總共時間	38.5692 秒
2. 平均時間	1.053 毫秒
3. 解題法平均使用率	
3.1　Naked 1	51.617 %
3.2　Hidden 1	45.376 %
3.3　Locked	2.569 %
3.4　Naked 2	0.192 %
3.5　Hidden 2	0.119 %
3.6　Naked 3	0.005 %
3.7　Hidden 3	0.002 %
3.8　X-Wing 2	0.003 %
3.9　X-Wing 3	0.003 %
3.10　Guess	0.113 %

(a)　候選數解題法

項目名稱	項目描述
1. 總共時間	28.0945 秒
2. 平均時間	0.767 毫秒
3. 解題法平均使用率	
3.1　Naked 1	96.297 %
3.2　Naked 2	2.097 %
3.3　Naked 3	0.105 %
3.4　Locked	1.389 %
3.5　Guess	0.112 %

(b)　候選個數解題法

在實驗改良基因重整繁衍法時，可以利用分散式運算的方式，增加整體處理的速度，例如多核 PC 或是多部 PC，系統架構如圖 24。以下利用多部 PC，分別針對改變提示數字個數 1 及 2 進行實驗，結果如表 6、表 7。

當改變數字個數為1時，共花了3.5天的時間處理36628個盤面，雖然產生了第一代新生子74271個，但經過重複盤面的檢驗後，僅9個是不重複的新生提示數17盤面。如果以2008年取得的47386個盤面重新測試，產生的新生盤面都是重複的，若要增加新生盤面的差異性，就要增加改變數字個數，產生更大量且差異更大的盤面。

然而，當改變數字個數為2時，因為大量的多重解發生，使檢驗所需時間大幅增加，用兩週的時間也僅能完成前26個盤面的統計數據，雖然單一母體產生的唯一解盤面數量大增，但都是重複盤面。

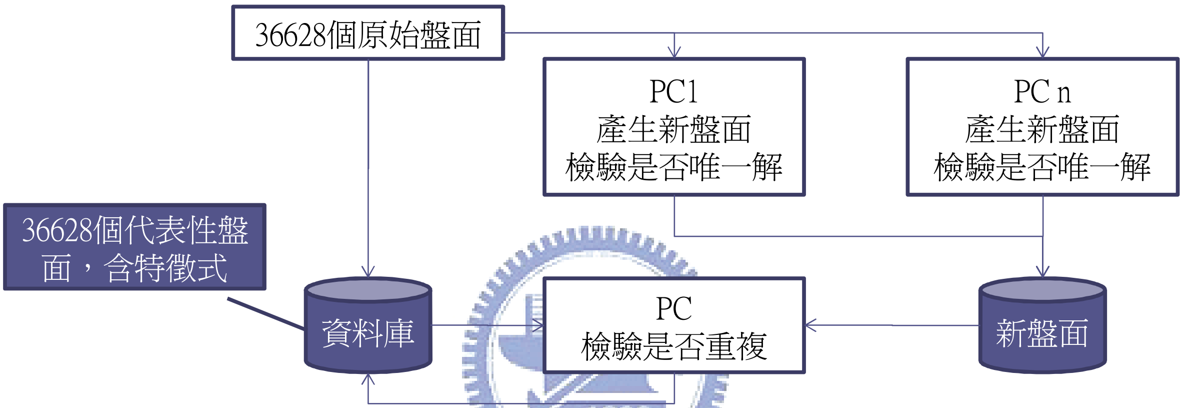
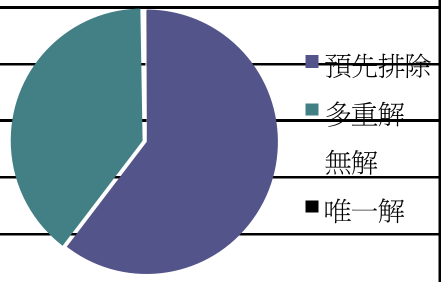


圖 24 分散式系統架構

表 6　改良基因重整繁衍法之實驗結果 1

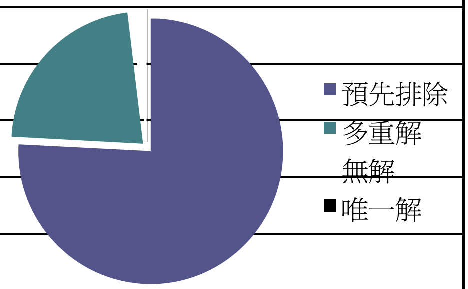
項目名稱	項目描述　（針對單一盤面）
1.　改變的數字個數	1
2.　處理所需時間	8.346 秒
3.　新生盤面 個數	9928
3.1　預先排除 個數	5993
3.2　多重解 個數	3909
3.3　無解 個數	24
3.4　唯一解 個數	2



項目名稱	項目描述　（針對36628個盤面）
1.　總共所需時間	3.5 天
2.　唯一解盤面 總數	74271
3.　新生盤面 總數	9（以2008年的47386個盤面重測，新生數 0）

表 7　改良基因重整繁衍法之實驗結果 2

項目名稱	項目描述　（針對單一盤面）
1.　改變的數字個數	2
2.　處理所需時間	40261.71 秒　(11.18 小時)
3.　新生盤面 個數	23470336
3.1　預先排除 個數	17799881
3.2　多重解 個數	5221630
3.3　無解 個數	448814
3.4　唯一解 個數	11



項目名稱	項目描述　（針對26個盤面）
1.　總共所需時間	12.1 天 (36628個盤面約需17068天 ≒ 47年)
2.　唯一解盤面 總數	286
3.　新生盤面 總數	0

第六章、結論與未來展望

本章綜合研究結果，針對兩個研究問題：(1)提出新的數獨解題法，使其能在更短時間內求解，並驗證是否具備唯一解；(2)改善以前的基因演算法，使有效且快速的產生提示數 17 的題目，並驗證是否重複，研究結果整理如下。

新的候選個數解題法幾乎僅用 Naked 1 就完成解題，解題所需時間較傳統的候選數解題法更少。實作上也只需要製作一套解題法、一個統一的架構，避開發生錯誤時，必須在不同的解題法間相互偵錯的困難。

改良後的基因重整繁衍法，除了應用候選個數解題法，預先過濾掉大量規則抵觸的盤面，也可以濾掉部分具有多重解的盤面，大幅減少需要考慮的盤面個數。在重複盤面的驗證上，也有了快速且完整的檢驗。

實驗的最後結果的確有產生新的提示數 17 盤面，所以證明改良基因重整繁衍法的確可以產生新的盤面，但數量有限。而新產生的盤面中，依舊沒有找到提示數 16 的盤面。而且當繼續下一步研究時，增加改變的提示數字個數為 2 時，因為多重解的情況大量出現，使唯一解的檢查變得相當耗時。所以，這實驗衍生了一個新的研究方向，那就是「如何快速辨識具有多重解的盤面?」。

此外，若改以證明提示數 16 盤面不存在為目的，改良基因重整繁衍法並沒有辦法證明是否能找出全部的提示數 17 盤面，進而證明這些盤面中沒有提示數 16 盤面的存在。反之，如果先找出所有的數獨的解盤面，進而找出每個盤面的最少提示數及題目盤面，這是證明最少提示數的另一個研究方向。

參考文獻

[1] 林育臣，「現今最少提示數之數獨盤面產生器研究」，國立臺灣師範大學，碩士論文，民國 96 年。

[2] Tom Davis, The Mathematics of Sudoku. Retrieved October 2008, from www.geometer.org/mathcircles/sudoku.pdf

[3] Bertram Felgenhauer, Frazer Jarvis, Enumerating possible Sudoku grids. Retrieved June 20, 2005, from http://www.afjarvis.staff.shef.ac.uk/sudoku/sudoku.pdf

[4] Gordon Royle, Minimum Sudoku. Retrieved June 1, 2006, from http://www.csse.uwa.edu.au/~gordon/sudokumin.php

[5] Sudoku. (n. d.). Retrieved June 1, 2006, from http://en.wikipedia.org/wiki/Sudoku

[6] Latin square Wikipedia. Retrieved June 1, 2006, from http://en.wikipedia.org/wiki/Latin_square

[7] Gary McGuire 's Minimum Sudoku Page, Sudoku Checker http://math.ie/checker.html

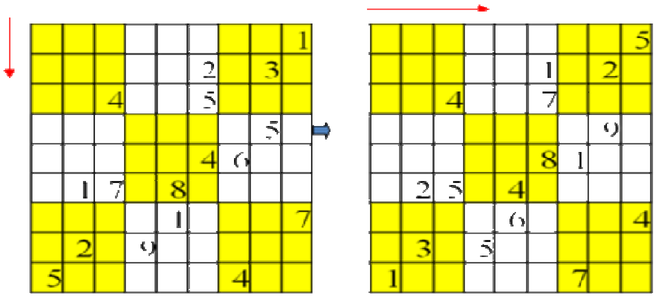
[8] 尤怪之家. Retrieved June 1, 2006, from http://oddest.nc.hcc.edu.tw/

附錄 A　重複盤面之範例說明

本附錄將以範例說明重複盤面之定義，以下介紹幾種透過相互對調的方式，可以得到不同盤面的方法。雖然盤面不相同，但由於盤面的特性都是相同的，所以將透過以下相互對調方式所產生的盤面，都視為重複盤面。

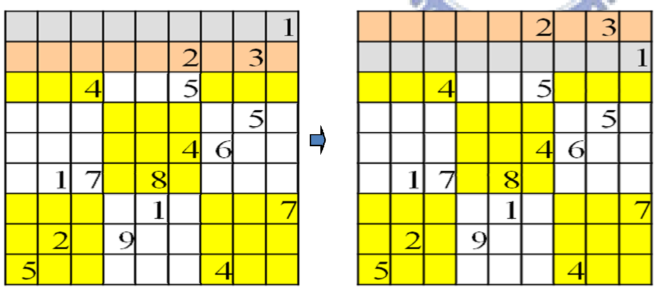
- **行列對調**

  座標位置上的行與列相互對調，相當於沿著左上右下的斜線，做盤面的翻轉。

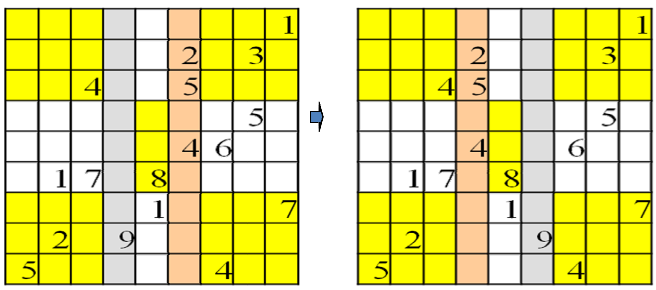


- **1 x 9 區域間，彼此相互對調**

  第 1 列、第 2 列、第 3 列等 1x9 區域，彼此間相互對調。同理，第 4 列、第 5 列、第 6 列，彼此間可以相互對調；及第 7 列、第 8 列、第 9 列，彼此間也可以相互對調。

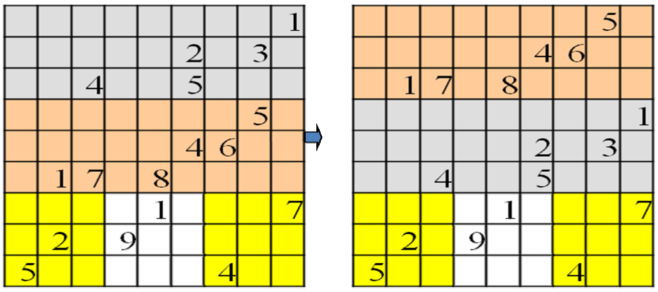


行與列相同，所以第 1 行、第 2 行、第 3 行，彼此間可以相互對調；第 4 行、第 5 行、第 6 行，彼此間可以相互對調；第 7 行、第 8 行、第 9 行等 1x9 區域，彼此間可以相互對調。

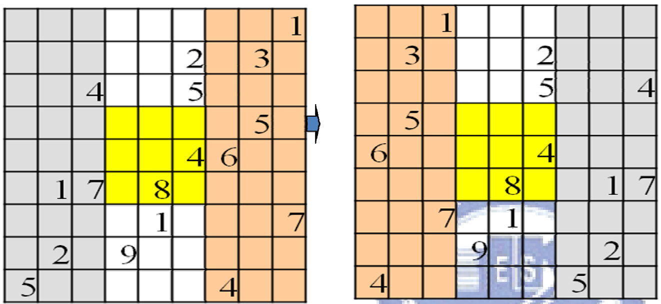


- **3 x 9 區域間，彼此相互對調**

  第 1~3 列、第 4~6 列、第 7~9 列等 3x9 的大區域，彼此間可以相互對調。



與列相同，所以第 1~3 行、第 4~6 行、第 7~9 行等 3x9 的大區域，彼此也可以相互對調。



- **數字 1~9 相互對調**

  針對盤面上出現的數字 1~9，任選兩種數字，將他們完全相互對調。