行政院國家科學委員會專題研究計畫成果報告

以限制條件為基礎之基因演算法在排程問題下的運作模式

計畫編號：NSC-89-2213-E-029-015
執行期限：88 **年** 8 **月** 1 **日至** 89 **年** 7 **月** 31 **日**

主持人：王偉華博士　　東海大學工業工程學系
Email: wangwh@ie.thu.edu.tw

一、中文摘要

基因演算法[5]**為一有效且穩健的搜尋方法。如今，此一方法已廣泛地應用於自然科學及人工智慧等領域。然而應用在解決工業界或是現實生活上常見的限制條件滿足問題**(Constraint Satisfaction Problems ; CSP)**，傳統的單點、雙點交配運算子或突運運算子，在運作過程上並沒有將限制條件納入考量，很容易在搜尋空間上流於亂走，作全面性的搜尋。造成求解過程搜尋時間上的浪費。有鑑於此，本研究將嘗試對基因演算法作改良，使其在面對一般性的限制條件滿足問題時，能夠在運作的過程中將限制條件納入考量，並結合基因演算法多點同步搜尋的特性，以更有效率的方式解決複雜的問題。此法乃以均勻交配運算子為基礎，並以表達限制條件滿足問題中的對偶圖作為決定基因交配之依據。使其在搜尋過程中產生的新解能落在符合部份限制條件的範圍之內，如此一來，對於限制條件滿足問題(如排程問題)可以在更短的時間內找到合理解，對於限制最佳化問題也可以很快地找到最佳解或近似最佳解。**

關鍵詞：基因演算法、限制條件滿足問題、對偶圖。

Abstract

Genetic Algorithm (GA) is an effective and sound searching method which has been widely used in artificial intelligence research and some industries applications. However, it is more difficult in finding out the right representation than the good answers. Constraint Satisfaction is a very power way in representing the problems. Thus, many real life industrial problems could be represented as a kind of Constraint Satisfaction Problem (CSP). Although GA is good in using multiple searchs to find out the "good" solution. However, it effectiveness and the efficiency is based on the randomness and greedy method. The merit of CSP is due to its constraint propagation. That is the solution space is truncated by the posted constraints. However, it is suffering by the poor search strategy.

In this research, we proposed a CSP based representation combined with the dual graph and uniform crossover method. In the preliminary study, it is very promising in limiting the search of GA in the partial-feasible solution space than the whole state space. The quality of the solutions could be enhanced and the time in finding them could be decreased effectively as well.

Keyword: Genetic Algorithm、Constraint Satisfaction Problem、Uniform Crossover、Dual Graph Representation

二、緣由與目的

利用基因演算法作為解決排程問題的工具時，我們會發現所要解決的問題常常伴隨著許多限制條件，使得求解的過程並不順利。主要原因乃是基因演算法在搜尋過程中並沒有處理限制條件的能力，必須加上額外的機制才可用來解決 CSP[8] **的問題。而常用的兩種方式有下列兩種：**

1.**懲罰函數法**(penalty function)[6][7]
這個作法是最常用的方法，它的概念來自於『如果我們直接解 CSP **是那麼的困難，那麼我們就將** CSP **的問題轉換成無限制式問題**(unconstraint problem)**再來解**』
2.**修正法**(decoders / repair)
修正法的概念來自於『如果我們所要解的問題是一個 CSP **的問題，那麼我們所要做的事就是將不合理的解修正成合理解』。**

然而上述兩種方式都是屬於事後補救的方法，以懲罰函數法而言其不能保證產生的解都落在合理解的範圍之內，還是有進入不合理解(infeasible solution)**範圍的情況。當不合理解產生時，我們才利用修正法將不合理解修正成為合理解。**

有鑑於此，本研究將嘗試對基因演算法作改良，使其在面對一般性的限制條件滿足問題時，能夠在運作的過程中將限制條件納入考量，並結合基因演算法多點同步搜尋的特性，以更有效率的方式解決複雜的問題。

本研究的目的主要是針對一般性的限制條件

滿足問題(但要求該問題的各個變數之值域為離散型)，重新設計基因演算法運算子(crossover、mutation)的運作方式，使其在搜尋的過程中考慮限制條件的因素，縮小搜尋範圍，以增加找到合理解或最佳解的機會。並使用限制推演(constraint propagation)有效縮減搜尋空間的能力，再結合基因演算法多點同步搜尋的優點，以縮短找到合理解及最佳解或近似最佳解的時間。

三、文獻探討

雖然基因演算法是一項可靠，穩健的廣域搜尋技術，但美中不足的是沒有處理限制條件的能力，因此必須加上額外的機制(mechanism)才可用來解決CSP問題。Michalewicz則針對一般基因演算法在處理限制條件之方法作探討，其中之一為懲罰函數法。懲罰函數法的內涵主要將CSP的問題轉換成無限制式問題(unconstraint problem)再來解，也就是說在計算的過程中，我們依據目標函數與限制式的差異，給定一些值來代替限制式，然後將這個值加入到目標函數中，使得目標函數能包含限制式的特性。依照上述的概念，假設有一個CSP的問題，它的目標函數為 $f(x_1,x_2,.....,x_n)$，有p個限制式，那麼可將目標函式轉換成

$$f(x_1, x_2,..., x_n) + \varepsilon \cdot \sum_{i=1}^{p} \delta_i \Phi_i$$

其中

ε =±1; 1:maximize; -1:minimize;

δ_i 為每一個限制式的penalty coefficient;

Φ 為penalty function;

依照這樣的轉換方式，一個限制條件滿足的問題就轉換成無限制條件的問題。但是這樣的做法有很大的一個問題在於懲罰函數的設計，而懲罰函數的設計往往就是找到最佳解的關鍵；另一個限制是，特定的問題有特定的懲罰函數，也就是說這樣的方式是依問題特性而變，不具有一般性。

而另一種在基因演算法中處理限制條件的方法為修正法(decoders/repair)，修正法主要的內涵是將不合理解(infeasible solution)修正成合理解，一般有兩個方法可以保證我們下一步所產生的解都一定符合限制式。一是設計一種資料結構，使得在這樣的結構之下所產生的解一定為合理解；另一方法是不去管資料結構，而是當不合理解產生時，如何修正此不合理解，使其成為合理解。以第一種方法而言，有他的困難存在，基本上要如何去找到一個資料結構，使他必定能產生合理解，這是相當困難的，如果條件少或許做的到，但當條件一多時，大概是很難找到這樣的資料結構。同樣的對第二種方法來說，我們也很難找到一個好的修正方法來修正不合理解，即使找到了，這也只對某一特定問題是用而已。

以上這兩種方法都是為了使演算法具有解決限制條件的能力所設計的。基本上，修正法的效率較高，但是不易設計；而懲罰函數法效率低、但使用上較具彈性且較為簡便。

簡介均勻交配(Uniform Crossover)運算子:

均勻交配[4]的方式是針對基因字串中的每一個基因，以類似擲銅板的方式來決定該基因是否互換。簡言之，對每一個基因而言，其互換的機率皆為 0.5。通常在採用均勻交配運算子之前，會對每一個基因隨機產生 0 或 1 的數，0 表示此基因不用互換、1 表示此基因要互換，而此一 0、1 的字串我們稱之為 Mask。其運作範例如下所示。

Parent 1=	1	0	1	1	0	0	1	1	0	1
Mask=	0	1	0	1	0	1	0	0	1	0
Parent 2=	0	0	1	0	0	1	1	0	1	1

圖 3-1 勻均 交配前之個體

Child 1=	1	0	1	0	0	1	1	1	1	1
Child 2=	0	0	1	1	0	0	1	0	0	1

圖 3-2 勻均交配後之個體

其餘有關文獻的詳細資料請參考[1]。

四、以限制為基礎的基因演算法運作方式

4.1 選擇 CSP 問題的表現法

根據 CSP 問題的定義，CSP 是由一組有限個數的變數所組成，每一個變數的值域皆是有限範圍的[8]。就僅考慮變數具離散型值域的情況下，我們選擇了對偶圖(dual graph representation)[2]的表現方法來表現 CSP 的結構。

那何謂 dual graph representation ? 現以下例作說明。假設一 CSP 之問題有 6 個變數(X_1~X_6)，及 4 個限制條件(C_1~C_4)，其對偶圖表現法如下所示。

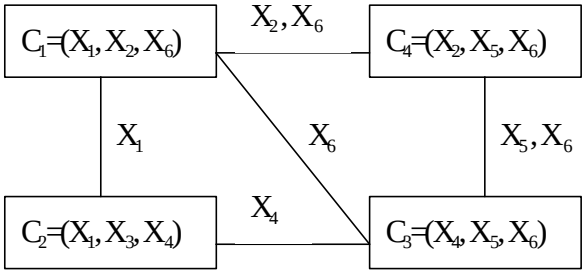


圖 4-1 CSP 問題之對偶圖表示法

上圖中 $C_1=(X_1,X_2,X_6)$的意思乃是指限制條件 C_1 是由變數 X_1、X_2、X_6 所構成(ex: $X_1+2X_2-3X_6>12$)，同理可知限制條件 C_2、C_3、C_4 分別是由為 (X_1,X_3,X_4)、(X_4,X_5,X_6)、(X_2,X_5,X_6)所構成。限制條件之間若有共用的變數，則將限制條件互相連線，並標註共用之變數為何。上圖中限制條件 C_1 及 C_4 即共用變數 X_2,X_6。利用 dual graph representation 來表示一限制條件滿足問題時，可以很容易了解其限制條件與變數之間的關係，並可根據共用變數的出現次數來決定此變數的重要程度，以作為基因演算法運作上的參考。

4.2 Dual Graph 廿 VC Table

利用對偶圖表示法來表現一限制條件滿足問題時，所表現的乃是問題變數與限制條件之間結構(struct)上的關係。可是相當一 CSP 的問題變得很大時，也就是變數與限制條件相對地增加許多，此時的對偶圖會變得很複雜，易造成視覺上混淆。有鑑於此，我們將對偶圖轉換成 VC Table(Variables vs. Constraints Table)。今以上例之對偶圖轉換成下表 4-1 的 VC Table。

表 4-1 CSP 問題之 VC Table

\ Constraints Variables \	C_1	C_2	C_3	C_4	Row Sum
X_1	1	1			2
X_2	1			1	2
X_3		1			1
X_4		1	1		2
X_5			1	1	2
X_6	1		1	1	3
Column Sum	3	3	3	3	

表中有標示 "1" 者表示此變數是構成限制條件的成員.

上表中「Column Sum」說明了相關限制條件的複雜度，數目越大，表示牽扯的變數越多，相形之下也較難控制。而「Row Sum」說明了各個變數的重要性，數目越大的，表示所影響的 constraints 越多，亦即，效益越強。

4.3 演算法的設計

利用基因演算法來搜尋解答的過程中，主要是運用一些擬生物化的人工運算過程，如複製、交配、突變等運算子進行運作。這些運算子中以交配運算子最為重要，它是使整個基因族群演化的重要關鍵。然而目前大部份基因演算法在交配運算子的運作上都是採用單點交配(one-point crossover)或是雙點交配(two-point crossover)，綜觀這些傳統的運算子在運算時鮮少將限制條件考慮進去，所以其搜尋路徑會遍及整個狀態空間(state space)，做全面性的搜尋。如此不但造成利用基因演算法求解的過程浪費太多的額外處理工作(例如修正法)，及搜尋時間上的浪費。如下圖 4-2 所示，假設 A、B、C、D 為眾多啟始點中的四個啟始點，每一啟始點的搜尋過程並不會將自己的搜尋範圍限制在某一個區域，而是在不合理解及部份合理解之間作跳躍式的搜尋，直到所產生的解剛好落到合理解的範圍之內，整個搜尋過程才結束。從圖 4-2 中看來，這樣的搜尋過程是很沒有效率。

有鑑於此，本研究認為若將搜尋範圍縮小在部份合理解(partial feasible solution)的區域，如圖 4-3 所示。對於搜尋到合理解或近似最佳解的效率，將會有很大的幫助。以下演算法之設計，將是針對這個概念而來，使基因演算法在搜尋的過程中，能因考慮限制條件的因素，將搜尋範圍限定在一固定區域內，以增加找到解答的效率。

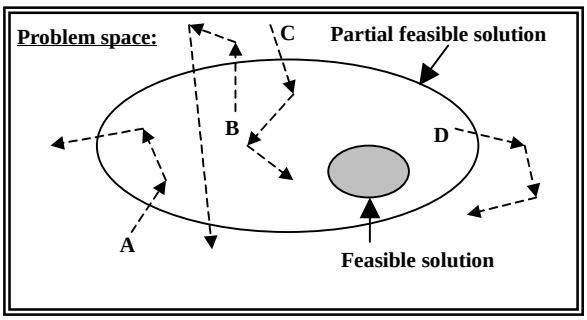


圖 4-2 傳統基因演算法搜尋過程示意圖

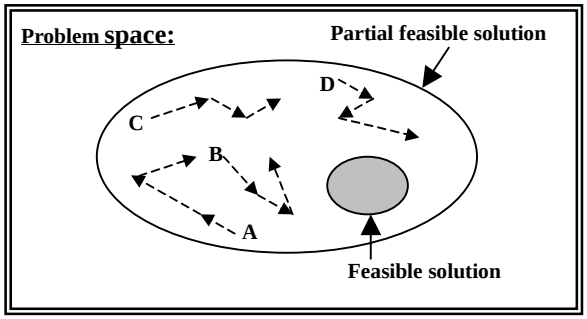


圖 4-3 考慮限制條件之基因演算法搜尋過程示意圖

I.初始族群的產生

為了符合如圖 4-3 的設計精神，在初始族群的產生上，本研究要求所產生的個體至少要符合一個以上的限制條件。若產生的個體沒有符合任何限制條件則採 backtrack 的方式重新選取目前所在變數的基因值或是全部重新取樣，如此將能確保初始族群中的每個個體都是 partial feasible solution。而要使個體至少符合一條以上的限制條件並不是一件很困難的工作，只要加入些許確認(checking)機制即

可。此確認過程除不會嚴重影響初始族群產生的速度外，若從全部都是符合部份限制條件的族群中再進行基因演算法的搜尋工作，我們相信會增加找到最佳解或近似最佳解的機會。

II.複製運算子的運作方式

在複製方面，本研究以進化策略(Evolution Strategy)中的選擇運算子(μ+λ)-ES[1][3]為依據。其中μ為母代的個數，λ為子代的個數，且μ≧λ≧1。其意義為將母代中最差的λ個個體，由子代全數取代，成為新的母代。通常在一般的實驗中設定μ>λ，因此在這樣的選擇機制運作下，保證每一世代的個體平均適應值將會呈現單調遞增(monotonic increasing)或單調遞減(monotonic decreasing)的情況[3]；也就是保證搜尋的方向將會往適應性最好的方向搜尋，且在搜尋時不會有跳開的情況發生。

III.交配運算子的運作方式

本研究將以對偶圖或 VC Table 作為均勻交配(uniform crossover)決定基因交配的參考根據，從限制條件的複雜度及變數的重要性兩個構面提出基因交配的方式。

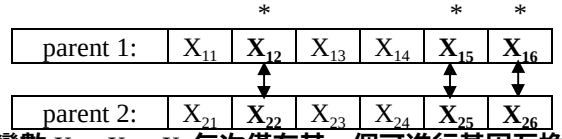
以下提出在考慮限制條件之下，基因進行交配的方式 **Constraint View Method**:

(1)從限制條件著手:

根據該 CSP 之 dual graph representation，隨機選取(with equal probability)所有限制條件中任一個，將構成此一限制條件的所有變數，設定為可以進行交換的變數(mask 設為 1)。其餘非構成此一限制條件的變數，其 mask 值為 0。

(2)改良 uniform crossover 的運作內涵:

以圖 4-1 為例，假設選取到限制條件 $C_4=(X_2,X_5,X_6)$，此時可進行交換的變數共有三個，若依照 uniform crossover 所定義的方式運作，就是將變數 X_2,X_5,X_6 一起進行交換的動作。不過在此並不打算採用此運作方式，稍微做了一點修改。在 mask 中標示為 1 的變數，是可進行交換的變數。但在此並不是考慮 $C_3^3=1$ 的情況，而是考慮 $C_1^3=3$ 的情況，也就是說每次僅有一基因可進行互換的動作。如下圖 4-4 所示。



變數 X_2、X_5、X_6 每次僅有其一個可進行基因互換

圖 4-4 基因交換示意圖

(3)交配的過程，產生多子代:

針對每一個可進行交換的變數進行基因互換的動作，在基因互換後，確認該交換變數符合相關限制條件[2]的程度。以變數 X_6 為例，在基因互換後，它就必須確認是否符合限制條件 C_1,C_3 及 C_4 其中之一。而以變數 X_2 為例，它僅須確認是否符合限制條件 C_1 及 C_4 其中之一。在分別對可交換的變數進行基因互換之後，唯有至少符合一相關限制條件的子代才會被視為合格的子代。以限制條件四，$C_4=(X_2,X_5,X_6)$為例，因其是由三個變數(X_2,X_5,X_6)所構成的，所以依據上述交配原則在其交配後會產生六個子代(交配後產生子代的情形如下表 4-2 所示)，這個六個子代會根據存活的條件來判斷該子代是否為新族群的成員。

表 4-2 基因交配後產生多子代之情形

交換 X_2:會產生的子代為 子代 1: X_{11}, **X_{21}**, X_{13}, X_{14}, X_{15}, X_{16} 子代 2: X_{21}, **X_{12}**, X_{23}, X_{24}, X_{25}, X_{26} 存活條件:符合限制條件(1)、(4)之一	交換 X_5:會產生的子代為 子代 3: X_{11}, X_{12}, X_{13}, X_{14}, **X_{25}**, X_{16} 子代 4: X_{21}, X_{22}, X_{23}, X_{24}, **X_{15}**, X_{26} 存活條件:符合限制條件(3)、(4)之一
交換 X_6:會產生的子代為 子代 5: X_{11}, X_{12}, X_{13}, X_{14}, X_{15}, **X_{26}** 子代 6: X_{21}, X_{22}, X_{23}, X_{24}, X_{25}, **X_{16}** 存活條件:符合限制條件(1)、(3)、(4)之一	ps:存活條件,請參照表 4-1

交配後產生的子代數要視所選擇到之限制條件其組成變數多寡來決定，若有二個變數則會產生四個子代，四個變數會產生八個子代。產生子代的個數是可交換變數個數的兩倍。因此不再侷限於傳統基因演算法一次交配僅能產生兩個子代的概念。

IV.突變運算子的運作方式

突變運算子的方式，與傳統基因演算法的突變方式並無差異。在突變的過程中並不加入任何限制，乃是隨機地從該變數的值域中任選一值作為突變值。也由於有突變運算子的關係，我們才能確保搜尋演算法的可達性(reachability)。

V.基因演算法運作流程

由於上述突變的過程中並不考慮突變後的個體是否符合任何限制條件，因此將不能保證所有經突變後的個體皆能落在部份合理解的範圍之內。此時若為了確保整個族群中的個體都是部份合理解，而將突變後的子代也進行確認，則有些基因型式將會被限制產生，如此不但會影響整個族群的多樣性(diversity)，也不能再確保搜尋空間上的可達性。基於上述理由，一方面為了使突變運算子的運用能保有擴展搜尋空間之特性，另一方面也要使族群中的個體在進行一連串的運算子之後，都還能落

[1] ES: Evolution Strategy

[2] 相關限制條件:共用該交換變數之所有限制條件

在部份合理解的範圍之內。因此將基因演算法的突變運算子置於交配運算子之前，此時的突變運算子可以在不考慮任何限制條件下進行突變，突變完之後的個體雖然不保證會落在部份合理解的範圍之內，但在進行基因交配之後所產生每一個子代皆保證是部份合理解。

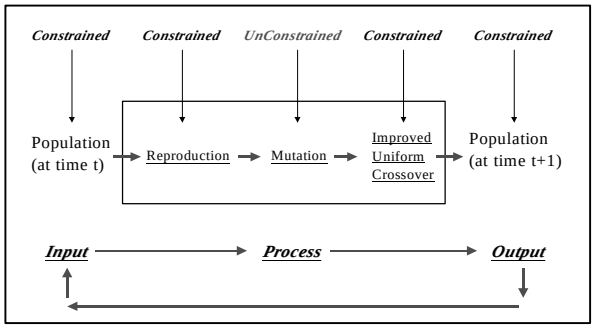


圖 4-5 考慮限制條件之基因演算法運作流程圖

VI.演算法之收斂性說明

根據 Hoffmesiter[3]所述，若某一演算法則符合下面兩個條件，那麼利用該演算法進行搜尋一定會收斂至整體最佳解。

1.經由此演算法則所產生的族群(population) 之適應函數值會呈現單調遞增或單調遞減的情況。
2.此演算法經由各種運算子的組合，可以到達(reachable)狀態空間中的任何一點。

由於在所設計的以限制條件為基礎之基因演算法運作模式中，其複製的過程是採用(μ+λ)-ES 運算子，所以適應函數值會呈現單調遞增或單調遞減的情況。再加上突變運算子的運用可以確保搜尋過程的可達性，因此完全符合上述收斂的兩個條件。因此所設計的演算法最終將收斂至最佳解。

五、台鐵動力機車排程問題實証

5.1 問題描述

根據台鐵列車時刻表，每天有 N 班車次，由 M 個動力機車來行駛。不同車次需搭配特定的動力機車。例如，自強號為 E1000、EMU100、EMU200 或 EMU300 型動力機車，莒光及復興號則為 E200 或 E400 型動力機車。數個使用相同型式動力機車之車次集結成一群組，稱為「機車運用」，並安排適當機務段之動力機車行駛之。

每一個機車運用的時間長度不得超過三天，且動力機車駛滿特定公里數或結束一個運用時需由原屬機務段執行保養工作，稱之日檢。例如，E1000 型動力機車每行駛 1500 公里必須執行保養，E200 及 E400 型之保養里程則為 1200 公里。安排動力機車運用除需考慮運用內每一個車次間應間隔適當時間以提供動力機車連結廂並執行基本檢查工作外，對於相連車次，前一車次迄站與次一車站起站間之距離應儘量接近，避免增加軌道使用量、動力機車行駛里程及乘務員之工作量。

若有一個以上之機務段持有相同型式的動力機車，則應儘量依照動力機車持有比率分派機車，使得各機務段維修人力及乘務員之工作量能夠平均分配。根據上述排班原則，安排動力機車運用，儘可能動用最少的動力機車數並使所有動力機車運用的總行駛里程數最短。

5.2 動力機車排班問題限制條件說明

根據台鐵機務處提供之資料，目前僅考慮西部幹線莒光號及復興號動力機車排班(共計 53 個車次，其詳細的車次資料如表 A-1 所示)，其它車次(自強號、平快車、普通車…)不在這次的問題之內，其相關之限制條件如下所述。

(1)莒光或復興號使用之動力機車型別有 E200 及 E400 型兩種。

(2)僅有南調及高雄機務段分別持有 E200 或 E400 型動力機車，所以莒光或復興號之機車運用的發車站為南調或高雄。

(3)南調機務段 E200 型動力機車有 12 輛，E400 型動力機車有 18 輛，合計 30 輛。高雄機務段 E200 型有 29 輛，無 E400 型機車。由於需考慮機車維修、保養及預備車之需求，所以各機務段每日可以使用的機車數不可超過該機務段機車數的 70%，亦即南調機務段每日最多有 21 輛機車、高雄機務段每日最多有 20 輛機車可供使用，每日最多僅有 41 輛動力機車可供運用。

(4)每一莒光或復興號僅需一輛 E200 或 E400 型動力車。

(5)每一機車運用的發車站及終點站應為同一機務段。

(6)E200 及 E400 型機車每行駛 1200 公里或完成一個運用必須保養，稱為日檢。日檢執行後，該機車之行駛里程重新累計。機車之日檢應由所屬機務段執行之，日檢至少需 3 小時。

(7)動力機車與第一個列車班次之距離不可大於 50 公里。

(8)動力機車連結車廂及基本檢查之工作時間平均為 45 分鐘，且相連車次上一車次之迄站與下一車次之起站間的距離不可以大於 50 公里。

(9)復興/莒光號之動力機車單機[3]迴送之平均時速為每小時 75 公里。

(10)過夜動力機車[4]之發車時間必須在 14:00 之前。

(11)過夜動力機車中相連車次最大間隔時間為 840

[3] 單機表示該動力機車沒有拉任何列車班次

[4] 超過晚上 24:00 點未回到原屬機務段者，視為過夜動力機車。

分鐘(14 小時)。

上述限制條件(10)(11)僅適用於過夜動力機車，非過夜動力機車並不適用。

5.3 台鐵動力機車排程問題編碼方式

台鐵動力機車排程問題的表示法乃是利用一個1×53的一維向量來表現動力機車排班情形。這 53 個元素分別代表 53 個列車班次。而每一個元素中所記錄的乃是動力機車的 ID[5]。這樣的編碼方式可以保證每一班列車一次僅能指派給一輛動力機車，而動力機車可以在不違背限制條件的情況下重複指派給不同的列車班次。其表示法如下所示:

項目:	班次 1	班次 2	班次 3	…	班次 53
內容:	1~41	1~41	1~41	…	1~41

圖 5-1 台鐵動力機車基因編碼方式

5.4 動力機車排程問題交配運算子運作方式

在台鐵動力機車排程問題中，交配運算子的運作方式乃是隨機地從母代選擇一台動力機車，今假設所選到的動力機車的 ID 為 18(以母代 1為主, 如圖 5-2 所示)，而該動力機車 18 所拉的列車班次有班次 2 及班次 52。此時班次 2 及班次 52 即是我們認為基因要交換的對象。

項目	班次 1	***班次 2***	班次 3	…	***班次 52***	班次 53
內容	17	***18***	23	…	***18***	15

項目	班次 1	班次 2	班次 3	…	班次 52	班次 53
內容	21	17	18	…	23	18

圖 5-2 交配前之母體

針對班次 2 而言, 基因交配後可產生代 1 及子代 2，如圖 5-3 所示:

項目	***班次 1***	***班次 2***	班次 3	…	班次 52	班次 53
內容	***17***	***17***	23	…	18	15

項目	班次 1	***班次 2***	***班次 3***	…	班次 52	***班次 53***
內容	21	***18***	***18***	…	23	***18***

圖 5-3 交配後之子代 1 及子代 2

在子代 1 中因為換上去的值為 17, 所以此時動力機車 17 拉了班次 1 及班次 2 的列車。而子代 1 存活的條件為符合該動力機車所屬的限制條件，以確保該動力機車 17 目前的狀態是處於 feasible solution。同理在子代 2中因為所換下去的值為 18, 此時動力機車 18 拉了班次 2、班次 3 及班次 53 的列車，其存活條件如上所述。而針對班次 52 而言，可產生子代 3 及子代 4，如圖 5-4 所示:

項目	班次 1	班次 2	***班次 3***	…	***班次 52***	班次 53
內容	17	18	***23***	…	***23***	15

項目	班次 1	班次 2	***班次 3***	…	***班次 52***	***班次 53***
內容	21	17	***18***	…	***18***	***18***

圖 5-4 交配後之子代 3 及子代 4

上述的基因交配運作方式，同時滿足所提演算法中的從限制條件中決定欲交換的基因變數及一次交配產生多子代的概念。

5.5 實驗結果:

在初步實驗中，利用所提方法求解的結果如下所示:

表 5-1 實驗結果

動力機車頭 ID	列車班次內容	行駛距離
1	25，1，18	913.9
2	49，27	456.4
3	44，35	818.1
22	15，20，46	873.3
23	14，22，9	805.9
24	51，52	216.4
25	50，28	216.4
26	48，30	235
27	16，5	768.9
28	12，7	768.9
29	45，32	812.2
30	37，23	805.9
31	24，41	804
32	43，40	864.4
33	39，2	808.4
34	29，47	859.5
35	33，17	815.2
36	3，13	767
37	31，19	767
38	26，42，38	903.7
39	6，21，53	853.2
40	8，34	820
41	4，10，36，11	954.8
	總距離	**16908.5**

ps:列車班次內容中的數字乃是表 A-1 中的編號。

六、參考文獻

[1] 廖祐笙,「整合基因演算法與限制條件滿足問題之研究」,東海大學工業工程研究所碩士論文, 民國八十八年.

[2] Bacchus, F. and P. Van Beek, "On The Conversion Between Non-Binary And Binary Constraint Satisfaction Problems", *Proceedings*

[5] ID:1~21 屬於南調機務段之動力機車;22~41 屬於高雄機務段之動力機車

of the National Conference on Artificial Intelligence Proceedings of the 1998 15th National Conference on Artificial Intelligence, AAAI Jul 26-30.

[3] Frank, H. and B. Thomas, "Genetic Algorithms and Evolution Strategies: Similarities and Differences", *System Analysis Research Group Technical Report* No.Sys-1/92, February 1992.

[4] Gilbert, S., "Uniform Crossover in Genetic Algorithm". *Proceedings of the 3th ICGA*, pp.2-9,1989.

[5] Goldberg, D. E., "*Genetic Algorithms in Search , Optimization , and Machine Learning*", Addison-Wesley, Reading, MA, 1989.

[6] Michalewicz, Z., " Genetic algorithm, numerical optimization and constraint", *Proceedings of the 6th ICGA*, L. J. Eshelman(ed.) (San Mateo, CA: Morgan Kufman), pp.151-158, 1995.

[7] Michalewicz, Z., " *Genetic Algorithm + Data Structures = Evolution Programs*, (Berlin:Springer), 1994.

[8] Tsang, E., " *Foundations of Constraint Satisfaction*", Academic Press, 1993

附錄 A: **表** A-1 **台鐵 復興/莒光 列車班次資料表**

編號:	車次代碼	起站代碼	迄站代碼	起站時間	迄站時間	經由路線	行駛距離
01	401	**豐原**(07)	**高雄**(14)	06:15	09:26	**縱海線**(11)	221.0
02	405	**基隆**(01)	**高雄**(14)	08:50	15:27	**縱海線**(11)	404.2
03	406	**高雄**(14)	**南調**(03)	09:45	15:53	**縱海線**(11)	383.5
04	407	**高雄**(14)	**屏東**(15)	10:15	10:38	**縱北線**(10)	24.6
05	409	**南調**(03)	**高雄**(14)	10:14	16:22	**縱海線**(11)	383.5
06	410	**高雄**(14)	**南調**(03)	10:30	16:39	**縱海線**(11)	383.5
07	413	**南調**(03)	**高雄**(14)	11:14	17:32	**縱南線**(12)	385.4
08	414	**屏東**(15)	**南調**(03)	11:00	17:50	**縱南線**(12)	410.0
09	415	**南調**(03)	**高雄**(14)	12:25	19:05	**縱南線**(12)	385.4
10	416	**高雄**(14)	**基隆**(01)	12:15	19:25	**縱南線**(12)	406.1
11	418	**高雄**(14)	**台南**(13)	17:05	17:45	**縱北線**(10)	46.7
12	420	**高雄**(14)	**南調**(03)	17:30	23:38	**縱海線**(11)	383.5
13	421	**南調**(03)	**高雄**(14)	18:14	23:26	**縱海線**(11)	383.5
14	422	**高雄**(14)	**台中**(08)	19:20	20:30	**縱海線**(11)	206.8
15	424	**高雄**(14)	**豐原**(07)	20:10	23:28	**縱海線**(11)	221.0
16	426	**高雄**(14)	**南調**(03)	23:00	29:27	**縱南線**(12)	385.4
17	427	**基隆**(01)	**高雄**(14)	22:15	29:07	**縱南線**(12)	406.1
18	428	**高雄**(14)	**基隆**(01)	23:30	30:31	**縱海線**(11)	404.2
19	429	**南調**(03)	**高雄**(14)	23:14	30:07	**縱海線**(11)	383.5
20	470	**彰化**(09)	**八堵**(02)	05:50	09:25	**縱南線**(12)	213.2
21	471	**八堵**(02)	**高雄**(14)	04:33	10:55	**縱海線**(11)	400.5
22	472	**台中**(08)	**八堵**(02)	07:05	10:34	**縱海線**(11)	193.7
23	473	**八堵**(02)	**高雄**(14)	18:26	25:15	**縱海線**(11)	400.5
24	474	**高雄**(14)	**八堵**(02)	15:15	21:48	**縱海線**(11)	400.5
25	475	**八堵**(02)	**彰化**(09)	20:41	24:25	**縱海線**(11)	211.3
26	483	**台南**(13)	**高雄**(14)	08:10	08:50	**縱北線**(10)	46.7
27	502	**二水**(10)	**南調**(03)	06:28	10:31	**縱南線**(12)	228.2
28	503	**嘉義**(12)	**高雄**(14)	06:00	07:41	**縱北線**(10)	108.2
29	504	**屏東**(15)	**基隆**(01)	06:02	13:45	**縱南線**(12)	430.7

30	507	**民雄**(11)	**高雄**(14)	06:36	08:22	**縱北線**(10)	117.5
31	508	**高雄**(14)	**南調**(03)	07:30	14:15	**縱海線**(11)	383.5
32	511	**基隆**(01)	**高雄**(14)	05:00	12:08	**縱南線**(12)	406.1
33	512	**高雄**(14)	**南調**(03)	08:10	14:52	**縱南線**(12)	385.4
34	513	**松山**(04)	**高雄**(14)	07:05	14:05	**縱南線**(12)	383.9
35	516	**高雄**(14)	**南調**(03)	09:15	15:39	**縱南線**(12)	385.4
36	517	**基隆**(01)	**高雄**(14)	07:25	14:56	**縱南線**(12)	406.1
37	520	**高雄**(14)	**南調**(03)	10:15	16:52	**縱南線**(12)	385.4
38	523	**基隆**(01)	**高雄**(14)	08:35	15:42	**縱南線**(12)	406.1
39	524	**高雄**(14)	**松山**(04)	11:45	19:05	**縱海線**(11)	382.0
40	527	**南調**(03)	**高雄**(14)	09:59	16:35	**縱南線**(12)	385.4
41	531	**南調**(03)	**高雄**(14)	11:29	18:42	**縱海線**(11)	383.5
42	532	**高雄**(14)	**基隆**(01)	13:45	21:50	**縱海線**(11)	404.2
43	536	**屏東**(15)	**基隆**(01)	13:47	21:30	**縱南線**(12)	430.7
44	539	**南調**(03)	**屏東**(15)	13:28	20:48	**縱海線**(11)	408.1
45	540	**高雄**(14)	**基隆**(01)	16:10	23:14	**縱南線**(12)	406.1
46	543	**南調**(03)	**高雄**(14)	13:29	21:15	**縱南線**(12)	385.4
47	547	**基隆**(01)	**高雄**(14)	15:00	22:25	**縱海線**(11)	404.2
48	548	**高雄**(14)	**民雄**(11)	18:15	20:02	**縱北線**(10)	117.5
49	549	**南調**(03)	**二水**(10)	16:46	20:56	**縱南線**(12)	228.2
50	552	**高雄**(14)	**嘉義**(12)	21:40	23:19	**縱北線**(10)	108.2
51	F544	**高雄**(14)	**嘉義**(12)	17:15	19:02	**縱北線**(10)	108.2
52	F551	**嘉義**(12)	**高雄**(14)	21:00	22:40	**縱北線**(10)	108.2
53	F511	**高雄**(14)	**屏東**(15)	12:20	12:45	**縱北線**(10)	24.6
					總距離 ：		16385.3

附錄 B　　**表** B-1 **台鐵各站的距離資料表**

	(1) **基隆**	(2) **八堵**	(3) **南調**	(4) **松山**	(5) **樹林**	(6) **中壢**	(7) **豐原**	(8) **台中**	(9) **彰化**	(10) **二水**	(11) **民雄**	(12) **嘉義**	(13) **台南**	(14) **高雄**	(15) **屏東**
(1)**基隆**	0.0	3.7	23.7	22.2	41.0	67.4	183.2	197.4	216.9	248.9	288.6	297.9	359.4	406.1	430.7
(2)**八堵**		0.0	20.0	18.5	37.3	63.7	179.5	193.7	213.2	245.2	284.9	294.2	355.7	402.4	427.0
(3)**南調**			0.0	1.5	20.3	46.7	162.5	176.7	196.2	228.2	267.9	277.2	338.7	385.4	410.0
(4)**松山**				0.0	18.8	45.2	161.0	175.2	194.7	226.7	266.4	275.7	337.2	383.9	408.5
(5)**樹林**					0.0	26.4	142.2	156.4	175.9	207.9	247.6	256.9	318.4	365.1	389.7
(6)**中壢**						0.0	115.8	130.0	149.5	181.5	221.2	230.5	292.0	338.7	363.3
(7)**豐原**							0.0	14.2	31.8	63.8	103.5	112.8	174.3	221.0	245.6
(8)**台中**								0.0	17.6	49.6	89.3	98.6	160.1	206.8	231.4
(9)**彰化**									0.0	32.0	71.7	81.0	142.5	189.2	213.8
(10)**二水**										0.0	39.7	49.0	110.5	157.2	181.8
(11)**民雄**											0.0	9.3	70.8	117.5	142.1
(12)**嘉義**												0.0	61.5	108.2	132.8
(13)**台南**													0.0	46.7	71.3
(14)**高雄**														0.0	24.6
(15)**屏東**															0.0

單位:公里