

國立交通大學  
工業工程與管理學系

碩士論文

具順序相依整備時間與非固定序列特性之  
流程式製造單元排程

**A Genetic Algorithm of Non-permutation  
Flow Shop Manufacturing Cell Scheduling  
with Sequence Dependent Setups**

研究生：洪浩翰

指導教授：巫木誠 博士

中華民國九十九年六月

具順序相依整備時間與非固定序列特性之流程式製造

單元排程

**A Genetic Algorithm of Non-permutation Flow Shop  
Manufacturing Cell Scheduling with Sequence  
Dependent Setups**

研究生：洪浩翰

Student：Hau-Han Hung

指導教授：巫木誠 博士

Advisor：Dr. Muh-Cherng Wu



Submitted to Department of Industrial Engineering and Management

College of Management

National Chiao Tung University

in Partial Fulfillment of the Requirements

for the Degree of Master of Science

in

Industrial Engineering

June 2010

Hsin-chu, Taiwan, Republic of China

中華民國九十九年六月

# 具順序相依整備時間與非固定序列特性之

## 流程式製造單元排程

研究生：洪浩翰

指導教授：巫木誠博士

國立交通大學工業工程與管理研究所

### 中文摘要

本論文是對於單元製造(Cellular Manufacturing)系統中，各個單元內的流程式製造單元(Flow Shop Manufacturing Cell)的排程問題。而研究的重點在於提出有別於過去的染色體表達法(Chromosome Representation)，希望從染色體表達法的改善，以提高基因演算法的求解品質。探討的問題為靜態排程的架構，每個工件被劃分至不同的工件族(Family)，而同一種工件族其特性為在同一機台加工，其間隔不需要額外的整備時間(Setuptime)，而若是加工過程在同一部機台上，換成另一群的工件族來加工，而此時就需要整備時間，所以同一工件族的工件一起加工，加工完了才換至另一個工件族的工件加工，如此才可以達到最有效率的加工順序，而在流程式製造中，在機台與機台間如何排各工件族加工順序，以及各加工的工件族內，不同工件的加工順序，就是這篇論文所要探討的排程問題。

由於這個問題複雜度高，如果要使用整數規劃的方式求得最佳解，所需要花費的時間會而當高，所以本篇論文是以基因演算法來求解這個問題。

而本篇論文所提出新的染色體表達法以 GA-N 表示，而傳統的染色體表達法以 GA-T 表示。

關鍵詞：排程、流程式製造、基因演算法、染色體表達法、順序相依整備時間

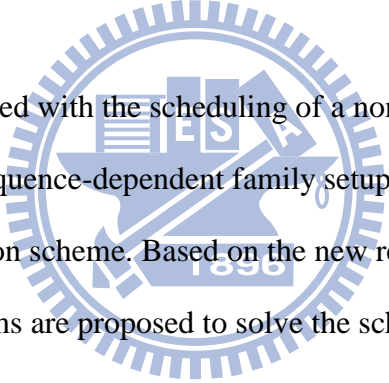
# A Genetic Algorithm of Non-permutation Flowline Manufacturing Cell Scheduling with Sequence Dependent Setups

Student : Hau-Han Hung

Advisor : Dr. Muh-Cherng Wu

Department of Industrial Engineering and Management  
National Chiao Tung University

## Abstract



This research is concerned with the scheduling of a non-permutation flowline manufacturing cell with sequence-dependent family setup times. We develop a new chromosome representation scheme. Based on the new representation and a prior one, two genetic algorithms are proposed to solve the scheduling problem. These two algorithms, respectively called GA-T (a prior algorithm) and GA-N (a proposed algorithm), adopt the same evolutionary mechanism but with different chromosome representation schemes. Numerical experiments indicate that GA-N outperforms GA-T in most test cases. This finding implies that developing new chromosome representation schemes might lead to better solutions quality for a particular meta-heuristic evolutionary mechanism. Choosing an appropriate chromosome representation is therefore very important to meta-heuristic algorithms.

**Keywords:** genetic algorithm, chromosome representation, scheduling, flowshop manufacturing cell, non-permutation, sequence-dependent family setups

## 誌謝

本論文能夠完成，最首要感謝的是我的指導教授巫木誠博士的細心指導。在整個研究的過程中，碰到瓶頸相當多次，好在巫老師的肯定與鼓勵，並為我找到對的方向，使我能完成這篇碩士論文。在巫老師的敦敦教誨下，使我在唸碩士這兩年來受益良多。而巫老師的教導總是從做人處事的方法、道理下手，這些對我來說，都是未來仍非常受用的。同時也感謝許錫美教授、彭德保教授在論文口試時，所給予的寶貴意見與指導，讓本論文更加完善。

在研究所的兩年中，要感謝同窗的陳曉嚴、凌金傳、戴邦豪以及謝佩芸，陪我一起修課，一起撰寫論文，彼此互相扶持，在我有困難的時候，都不吝惜給與幫助，使我在交大碩士的學習過程中非常順利。也並特別感謝邱志文學長的幫助，指導我 C++ 的撰寫，並且告訴我基因演算法的原理邏輯，使我在寫程式時能夠得心應手。我也非常感謝陳文旻，在碩二一整年，身邊發生太多事情的我，因為有她一直陪著我，分擔我的煩惱，讓我心情能夠有個抒發的管道，並且對於我的論文或多或少也給與了建議，使我得到相當多的幫助。

我也特別感謝我的家人，在研究所兩年內，我就發生了好多次困境、低潮期，家人總是以我的學業放第一優先考量，如果沒有他們的支持，相信我這兩年的學習過程會更困難許多。最後謹以此論文獻給我最敬愛的家人、師長與朋友。

洪浩翰 于 新竹交大

2010'7'1

## 目錄

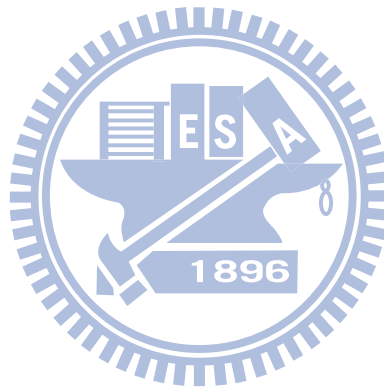
中文摘要.....	i
Abstract.....	ii
誌謝.....	ii
第一章 緒論.....	1
1.1 研究背景.....	1
1.2 研究議題.....	2
1.3 研究動機.....	3
1.4 研究方法與假設.....	4
1.5 論文組織.....	5
第二章 文獻探討.....	6
2.1 流程式製造單元排程(Flow Shop Manufacturing Cell Scheduling).....	6
2.2 家族整備時間(Family Setup Time).....	7
2.3 Permutation V.S. Non-permutation.....	8
2.4 基因演算法.....	9
2.4.1 染色體.....	10
2.4.2 適合度函數.....	10
2.4.3 交配.....	11
2.4.4 突變.....	12
2.4.5 交配率與突變率.....	12
2.4.6 選擇策略.....	12
第三章: 研究方法.....	13
3.1 研究問題.....	13
3.2 複雜度分析.....	14
3.3 基因演算法求解方法.....	14
3.3.1 染色體設計與解讀.....	15
3.3.2 適合度函數定義.....	16
3.3.3 交配.....	16

3.3.4 突變.....	17
3.3.5 傳統基因表達法實際執行方法.....	18
3.3.6 新式基因表達法搜尋 Family 加工順序的方法 .....	19
第四章 實驗情境與結果.....	25
4.1 實驗情境描述.....	25
4.2 實驗結果.....	25
4.3 小規模新式表達法輸的原因.....	30
第五章 結論與未來建議.....	32



## 表目錄

表 2.1 基因演算法求解應用於 FLOW SHOP 或單元製造系統 .....	6
表 2.2 NON-PERMUTATION V.S. PERMUTATION 的相關文獻 .....	9
表 4.1 結果 .....	26





## 圖目錄

圖 1.1 流線型工廠圖示.....	2
圖 1.2 基因演算法示意.....	4
圖 2.1 家族整備時間示意圖.....	7
圖 2.2 PERMUTATION V.S. NON-PERMUTATION .....	8
圖 2.3 基因演算法簡易流程.....	11
圖 3.1 基因演算法詳細流程.....	14
圖 3.2 傳統染色體的設計.....	15
圖 3.3 本篇論文所採用染色體的設計.....	15
圖 3.4 本篇論文解讀 FAMILY 加工順序.....	16
圖 3.5 為 PBX 的交配示意圖.....	16
圖 3.6 為 SWAP 示意圖.....	17
圖 3.7 為 INSERT 示意圖.....	18
圖 3.8 為傳統的染色體表達法作 SWAP 示意圖.....	19
圖 3.9 新式染色體.....	19
圖 3.10 新式染色體表達法找出機台 1 的 FAMILY 加工順序.....	20
圖 3.11 新式染色體表達法找出機台 1 的各工件加工順序.....	21
圖 3.12 新式染色體表達法派至機台 1 加工情況.....	21
圖 3.13 JOB1 在機台 1 加工完接著到機台 2 加工發生閒置的情況.....	21
圖 3.14 JOB6 在機台 1 加工完接著到機台 2 加工發生閒置的情況.....	21
圖 3.15 FAMILY2 和 FAMILY3 在機台 2 加工順序互換的情況.....	21
圖 3.16 把 FAMILY1 保留到機台 2 第一個加工(步驟一).....	22
圖 3.17 考慮 FAMILY3 與 FAMILY5 是否交換(步驟二).....	22
圖 3.18 考慮 FAMILY3 與 FAMILY2 是否交換(步驟三).....	23

圖 3.19 考慮 FAMILY2 與 FAMILY4 是否交換(步驟四).....	23
圖 3.20 確定機台 2 的 FAMILY 加工順序(步驟五) .....	23
圖 3.21 把 FAMILY1 保留到機台 3 第一個加工(步驟六).....	23
圖 3.22 決定全部機台的 FAMILY 加工順序 .....	24
圖 4.1 SSU 平均改善率 .....	27
圖 4.2 SSU 最大改善率 .....	28
圖 4.3 MSU 平均改善率 .....	28
圖 4.4 MSU 最大改善率 .....	29
圖 4.5 LSU 平均改善率 .....	29
圖 4.6 LSU 最大改善率 .....	30
圖 4.7 FAMILY 在各機台最佳排序(例子).....	31
圖 4.8 新式表達法所能求解的最佳 FAMILY 加工排序(例子).....	31



# 第一章 緒論

## 1.1 研究背景

在工廠內等待加工各個工件(Jobs)都帶有不同的屬性，例如加工時間、交期、數量…等，而因為這些屬性的不同，我們安排各個工件的加工順序不同時，可能就會有不同的績效，這就是為什麼我們要討論排程問題。

排程(Scheduling)就是決定每個工件的加工順序，一個好的排程可以讓生產的效率提高，一般衡量一個排程的好壞會有很多績效指標，舉最小化完工時間(Minimization Makespan)為例，好的排程能使完工時間短，也可減少機台的閒置(Idle)時間，因此也可以生產出較多的產品，所以排程問題在各個工廠中一直是很重要的一項議題。

排程大都是 NP-Hard (Garey & Johnson, 1979)的題目，意思是說當工件數及機台數增加時，其複雜度會上升很多，例如：3 個工件 3 台機台，其有  $(3!)^3$  種組合，而當工件 10 個機台 10 台時，其有  $(10!)^{10}$  種組合，看起來只多 7 個工件，多 7 台機台，但其組合卻突然暴增許多，這就是為什麼排程是 NP-Hard 的題目。

而由於排程問題有這樣子的特性，所以當規模比較小的時候，也許我們還可以用整數規劃(Integral Programming ; IP)來求解，但是一旦問題變大了，就使得整數規劃已經無法求解，或者是要花非常長的時間來求解，此時，我們就得依靠搜尋式演算法(Meta-Heuristic)來求解，例如基因演算法(Genetic Algorithm ; GA)、螞蟻演算法(Ant Colony Optimization ; ACO)、模擬退火法(Simulated Annealing ; SA)，這樣的演算法可以在相當短的時間內，求得一個不錯的解，甚至可以求得最佳解，而本篇論文所使用到的搜尋式演算法是基因演算法。

## 1.2 研究議題

本論文是流程式生產(Flow Shop)排程，生產加工模式(如圖 1.2)是各個加工機台皆為不同機台且排列在同一條生產線上，各個工件從機台一開始加工，每個工件都必須要經過每部機台加工，以圖 1.2 為例，工件從機台一進入加工系統，當工件從第一部機台加工完成，則轉送到機台二加工，最後三部機台都加工完成後，從機台三產出，因此該部機台的緩衝區內已無前一部機台的加工完的工件時，該部機台就必須閒置，直到工件到達。

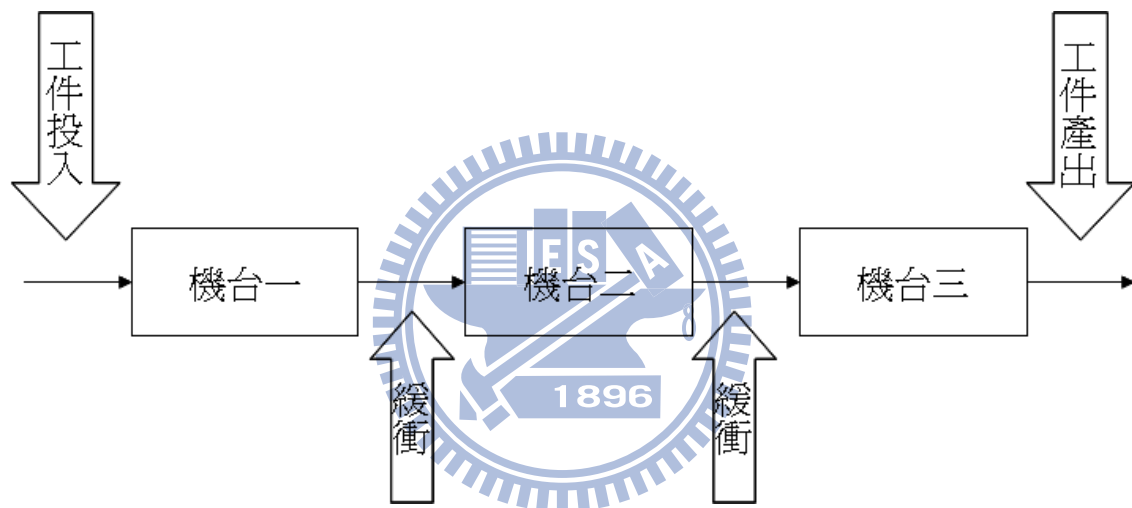


圖 1.1 流線型工廠圖示

工件又分為很多的工件族，同一個工件族(Family)的特性是當連續在同一部機台上加工時不用整備時間(Setuptime)，當同一部機台其加工的工件族更換時，這時就必須要花費整備時間，本論文就是希望節省這樣的整備時間，所以加工一定把同一個工件族的工件一起加工。

而本論文探討的是具有順序相依整備時間(Sequence Dependent Setups)性質的加工模式，且擁有非固定序列(Non-permutation)特性，所以，各個工件族的在各個機台上的加工順序可以不同。

而這樣的一個排程問題，套用在現實生產情境的例子可以是印刷電路板的插

件機製程的排程問題(Schaller et al. 2000)。每一塊電路板上都要插上非常多的電子零件，而這些印刷電路板就會經過一系列的插件機台加工，一塊電路板我們就視為一個工件，而一部插件機台就是一個工作站，每一個工件都必須要按照順序經過各個工作站，如此就形成了一個流程式生產的問題。當工件到達一個工作站時會插上一組電子零件群，而當兩塊不同的印刷電路板經過插件機台加工時，它們所使用上的電子零件群可能想當類似，甚至完全一樣，那麼這樣子我們就把這兩塊印刷電路板視為同一個工件族，因為它們所需使用上的零件雷同，那麼他們在同一部機台上加工時，需要的整備時間就很小或者甚至不需要整備時間。所以這樣子的生產模式，我們為了能使各機台在加工時，不把時間浪費在多餘的整備時間上，在派工時，我們就會把同一個工件族的工件排在一起加工，如此就形成家族式派工(Family-based)的一個排程問題。



### 1.3 研究動機

而近年來基因演算法相當盛行，很多的排程問題都會使用上，不過針對基因演算法的研究，大部分的 paper 都是著墨於演算法的進化方法，例如染色體突變的方法或是染色體交配的方法，還有染色體如何篩選或淘汰，這些方法的確都能夠對基因演算法的應用上有重大的貢獻，不過卻較少有論文把重點放在解讀法的改善部分，我們認為基因演算法應該包含有進化法與表達法的結合(如圖 1.1)，所以，想要改善基因演算法的求解品質，也可以從表達法下手，我們相信較好的染色體表達法的解讀，就算使用相同的進化方法，就能提升求解的品質，而改善染色體表達法的解讀概念，是延伸自呂佳玟(2009)的論文對於染色體表達法的概念，該篇論文就是從染色體表達法的解讀改善，就能比傳統的基因演算法求解，有明顯的改善，所以，我們希望能延讀這樣子的想法，應用到另外的生產類型，來研究，是否能同樣從染色體表達法的改善著手，使得整個基因演算法可以求得

較好的解。

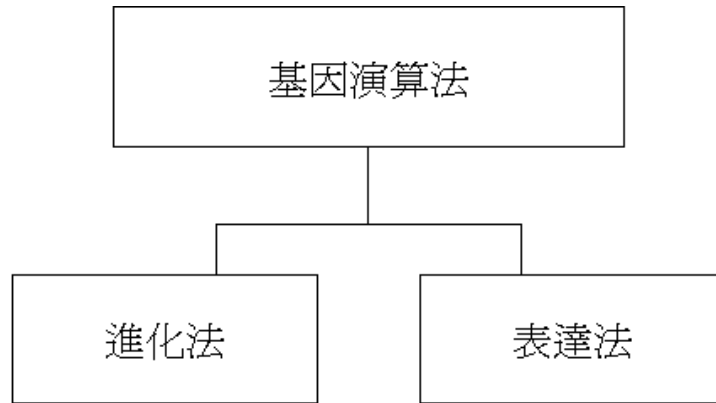


圖 1.2 基因演算法示意

#### 1.4 研究方法與假設

本論文研究的問題是非固定序列特性之流程式工廠排程，而在工件族間具有順序相依的整備時間，這類的排程問題，可以透過本篇論文的研究來決定如何作工件在各工件族內的排序以及各工件族在各機台的加工順序。

本研究的重點是從染色體表達法的改善，而研究的數據是依照 Lin et al. (2009) 的論文，該篇論文對整備時間分成長、中、短三種，而在各種整備時間底下，對機台數和工件族數目又做了十種組合，所以全部的情境一共有三十種，因為本篇的研究是希望從表達法的改善，來跟該篇論文比較，看基因演算法最後的求解品質是否真有提升，所以，本篇研究也使用這三十種情境，而基因演算法的進化機制則是符合該篇論文的方法。

本篇論文的問題是屬於靜態排程，也就是說在演算法開始前，已知總共有多少工件且各個工件其各屬性（如在各機台上的加工時間、交期）皆已確定。

而這篇論文使用的衡量績效指標是  $C_{\max}$  意即 Makespan，就是從一開始，到全部工件完成加工，其中所需要花費的時間。

本篇論文的假設有：

1. 在同工件族內，各工件間的整備時間忽略不計。
2. 各機台的最一開始的工件一到馬上就可以加工，即準備時間設為零。
3. 在機台與機台間的運輸時間等於零。
4. 機台前緩衝區內的在製品量沒有上限。
5. 一部機台一次只能加工一個工件。

## 1.5 論文組織

本論文內容共含有五章。第一章是說明研究背景、研究議題、研究動機和研究方法與假設介紹。而第二章是作相關文獻的介紹，首先是對流程式製造單元排程作介紹，再說明具順序相依整備時間與不具順序相依整備時間的差異，接著說明非固定序列特性，第二章最後對求解工具基因演算法作介紹。第三章是針對問題的定義，說明演算法採用的進化機制，以及不同染色體表達法在使用上造成解讀的差異。第四章為傳統染色體表達法與改良式的染色體表達法，其針對 Makespan 的結果差異。第五章則是提出未來可以延伸之研究方向。



## 第二章 文獻探討

本章首先簡介探討整備時間具有順序相依或獨立的特性，再介紹非固定序列性質，最後則針對本論文會使用到的演算法之相關研究進行探討。

### 2.1 流程式製造單元排程(Flow Shop Manufacturing Cell Scheduling)

單元製造(Cellular Manufacturing)是群組技術的重要的應用，單元製造可以同時擁有零工式生產(Job Shops)和流程式生產(Flow Shops)的優點，它可以彌補流程式生產無法達到的多樣化生產，卻又不會因此而讓產出速率降低太多，因此，對於現今客戶的需求朝向多樣化發展的時代，單元製造成為了現在相當重要的一種生產模式。而製造單元排程就是細胞製造中的一個重要部分。

群組技術是由 Mitrofanov(1966)提出，它是將很多的機台和工件先分群，讓各工件族內的各工件的製造特性相似，這些工件都必須要進到某一製造單元內的每一部機台作加工，而因為製造特性相似，所以該工件族內的工件與工件之間不需要太長的整備時間，而換製下一個工件族的工件群來加工時，才需要一段較長的整備時間，如此，各個製造單元內就成了流程式排程問題。而對於流程式排程問題的參考資料見表 2.1。

表 2.1 基因演算法求解應用於 Flow Shop 或單元製造系統

作者	年份	簡述研究內容
Lin, S. W, Ying, K. C, Lee, Z. J	2009	本篇論文投入數據的參考依據，該篇論文是以基因、塔布以及退火演算法，針對具順序相依整備時間和非序列式流線型排程問題。
Ruiz and Maroto	2006	基因演算法來求解具有順序相依整備時間的流程式排程問題。



## 2.2 家族整備時間(Family Setup Time)

Family Setup Time 是指對於不同工件族 (Parts Family) 間在同部機台上加工，必須要有整備時間(Setup Time)，雖然在同工件族內的各個工件間也是可能有一些小的整備時間，但是比起 Family 與 Family 間的整備時間，同 Family 內工件間的整備時間就小很多，在本篇論文內，工件間的整備時間不計算。

而整備時間有又分為順序相依家族整備時間 (Sequence-Dependent Family Setup Time) 與順序獨立家族整備時間 (Sequence-Independent Family Setup Time)，這樣子的分類由 Potts & Kovalyov (2000) 提出。

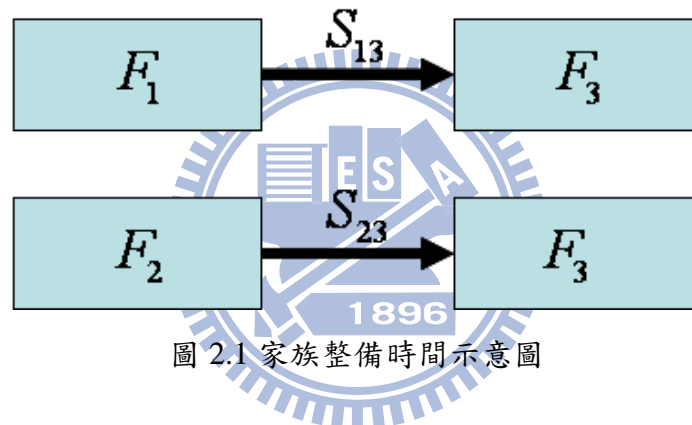


圖 2.1 家族整備時間示意圖

圖 2.1 是家族整備時間的的示意圖，其中  $F_1$ 、 $F_2$  和  $F_3$  是三個不同的工件族，而當在同一部機台上加工時， $F_1$  加工完後要換  $F_3$  加工時，必須要有  $S_{13}$  的整備時間，而同樣的當  $F_2$  加工完後要換  $F_3$  加工時，必須要有  $S_{23}$  的整備時間，而這  $S_{13}$  和  $S_{23}$  這兩值是否相等，就是順序相依家族整備時間和順序獨立家族整備時間兩類不同之處。

順序相依家族整備時間：

即家族整備時間會依照前面一個 Family 不同而有所不同，意思是圖 2.1 當  $F_3$  前一個加工是  $F_1$  時，與  $F_3$  前一個加工是  $F_2$ ，產生的整備時間分別為  $S_{13}$  和  $S_{23}$ ，這兩個使是不相等的。

順序獨立家族整備時間：

相反的順序獨立家族整備時間正說家族整備時間並不會因為前一個 Family 的不同而有所不同，意即  $S_{13}$  和  $S_{23}$  兩值相等。

### 2.3 Permutation V.S. Non-permutation

圖 2.2 是固定序列式問題(Permutation)與非固定序列式(Non-permutation)問題的差異，現在工廠內總共有  $m$  部機台，待加工的 Family 數總共有三個，分別是  $F_1$ 、 $F_2$  和  $F_3$ ，而每個 Family 都必須要經過各個加工機台作加工，而圖 2.2 上半部是 Permutation 的例子，在 Family 加工順序那一行中一小段，指的就是三個 Family 加工順序，從圖中可以得知，三個 Family 的加工順序不管在第一台或在之後接下來各個機台，其加工順序都是先做  $F_3$ ，做完換  $F_1$  最後再做  $F_2$ 。而相反的下半部 Non-Permutation 的特性，就是在不同機台上加工，各個加工的 Family 順序是可以改變的，如圖 2.2 在機台一的加工順序是先做  $F_3$  再做  $F_1$  最後做  $F_2$ ，而到了機台二時，加工順序變為先作  $F_2$  再做  $F_1$  最後做  $F_3$ 。

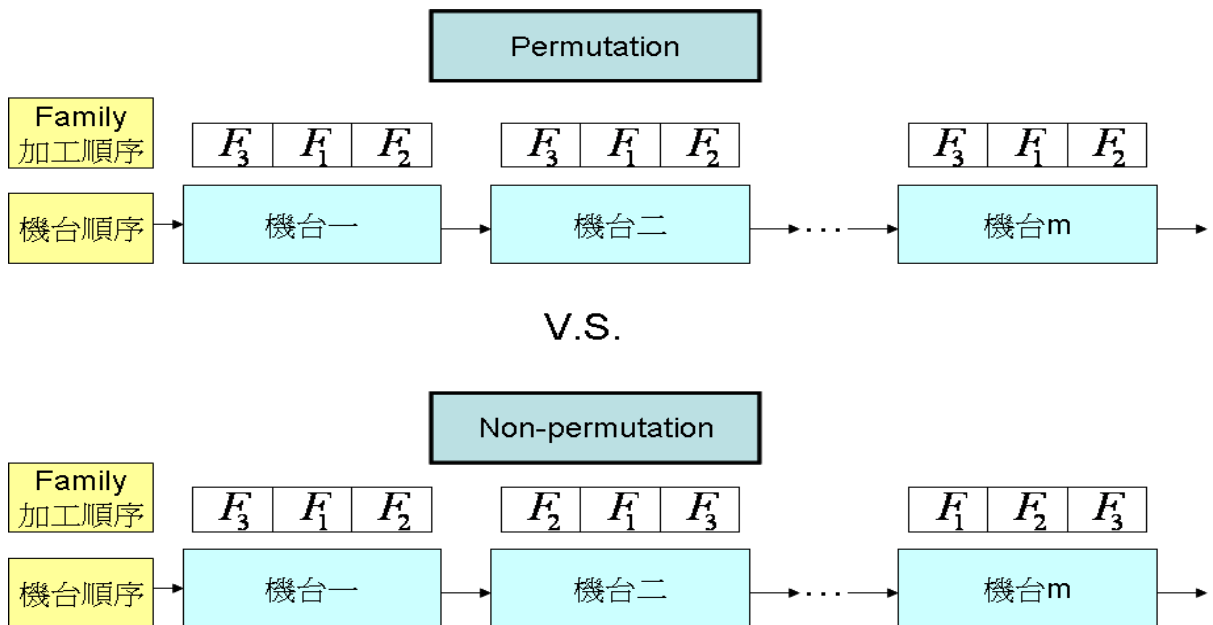


圖 2.2 Permutation V.S. Non-permutation

經過上述的例子說明，我們可以知道 Non-permutation 的複雜度會比 Permutation 的複雜度來得高，假設 Family1、Family2 和 Family3 中所包含的工件數分別是  $n_1$ 、 $n_2$  和  $n_3$ ，那麼在 Permutation 中工件加工順序的組合總共是  $(n_1!) * (n_2!) * (n_3!)$  種，而在 Non-permutation 的情況，全部工件加工順序的組合有  $(n_1!) * (n_2!) * (n_3!)^m$  種。

有關於 Permutation 和 Non-permutation 的參考文獻有：

表 2.2 Non-permutation V.S. Permutation 的相關文獻

作者	年份	簡述研究內容
Liao, C. J.; Liao, L. M.; Tseng, C. T.	2006	對 flow shop 的排程問題，目標值以總完工時間、總延遲時間、含權重的總延遲時間三項來比較 Non-permutation 和 Permutation。
S. Pugazhendhi, S. Thiagarajan, C Rajendran and N. Anantharaman	2004	在 flowline，且具有順序相依整備時間的排程問題，使用 heuristic or near-optimal methods 來求解證明 Non-permutation 可以比 Permutation 得到更好的績效

#### 2.4 基因演算法

基因演算法是由 Holland(1975)提出，其搜尋求解概念是來自於遺傳學的染色體(DNA)的遺傳或是突變機制。基因演算法其概念是把我想們要求解的相關資訊轉換成基因編碼(Gene encoding)，當我們決定好基因編碼方式以及解讀方法後，就可以開始我們的基因演算法的整個流程。

首先，要隨機產生一群初解，我們稱作母代(Population)，再依據我們想要比較的目標值，把這些母代的基因透過目標函數(Objective Function)轉換成目標值，再把這些目標值透過適應值函數轉換成適應值(Fitness)，當適應值越高表示

這條染色體所代表的解是比較好的，所以之後我們在作突變(Mutation)或交配(Crossover)時，會產生新的染色體，我們稱作子代，而這些子代也是要透過目標函數轉成目標值，再透過適應值函數轉成適應值，當子代適應值比交配或突變的母代好時，就會替換掉母代的染色體，這個過程的目的就是讓原先的一群初解，透過這個機制反覆替換掉適應值較差的函數，最後，我們可以得到較好的解，這解可能是最佳解也可能是很接近最佳解的近似最佳解。

### 2.4.1 染色體

染色體就是一條可以代表我們問題中其一個解的字串，這條染色體可以再透過目標值函數轉換成目標值，即表示，這條染色體所代表的組合，其可成的目標值就是從目標值函數得出的結果。



### 2.4.2 適合度函數

我們在作基因演算法時，一定會先決定我們的目標是什麼(像本篇論文採用Makespan 為目標值)，決定好後，我們就能把染色體解碼，利用目標值函數得到的目標值，再透過這適合度函數求得這染色體所代表的適合度。這適合度從名詞即可以知道是適合的程度，也就是說當適合度越高，代表這組解越好，所以才可知我們在做淘汰篩選機制時，是否要淘汰掉母代的組合。

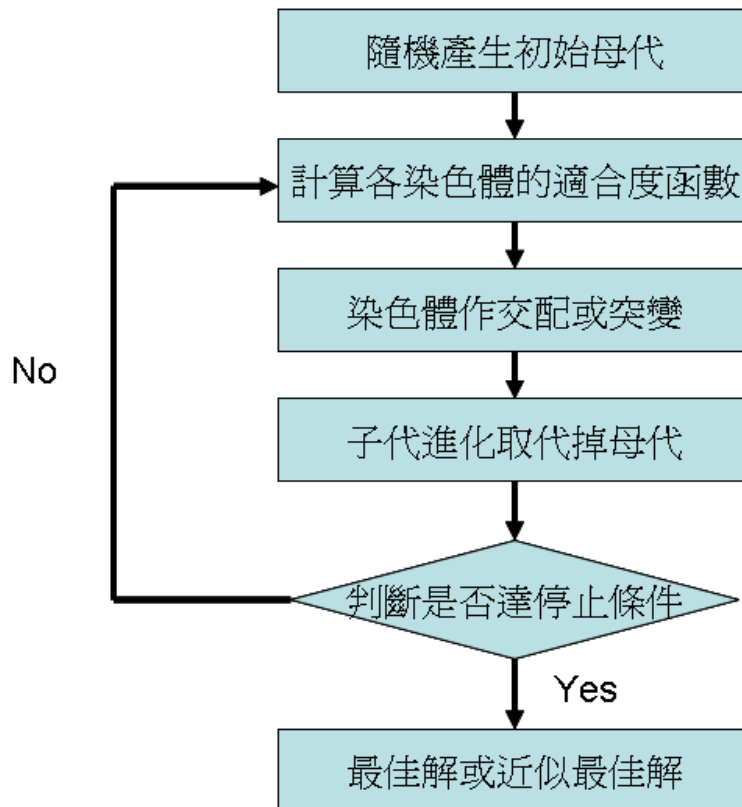


圖 2.3 基因演算法簡易流程

### 2.4.3 交配

交配是從母體中隨機抽出兩條染色體，而這兩條染色體交換交換它們彼此的基因，利用這樣子的方法，可能可以保留到母體中較好的解的片段，因此而得到更好的子代來。交配有單點交配 (One-point Crossover)、線性順序交配(LOX: Linear Order Crossover Operator)、部分相應交配(PMX: Partially Matched Crossover Operator)、 NABEL Operator 等。關於各種交配的方式在 Talip Kellego'z ,\*, Bilal Toklu , John Wilson(2008)這篇論文中有作介紹。而本篇論文所採用的交配方式是 Position Based Crossover (PBX)。

#### 2.4.4 突變

突變通常會配合交配一同使用，其目的是避免基因演算法陷入局部最佳化 (local optimum)。一般突變的種類有交換(Swap)、反轉(Inverse)及插入(Insert) Wang and Zheng(2003)中有作說明。

#### 2.4.5 交配率與突變率

我們在作基因演算法時，會給予交配率和突變率兩項設定值，這兩個值是用來決定整個基因演算法，交配和突變兩者的比重是多少，一般我們會把交配率的值設高一點，而突變率的值設低一點。

#### 2.4.6 選擇策略

在基因演算法的求解過程中，染色體作交配與突變後，交配池中會有舊有的母代以及新產生子代的染色體，這時要依據各個染色體的適應函數值來選擇哪些染色體要被淘汰掉。而選擇的方式有很多種，其中較常見的有：俄羅斯輪盤法 (Roulette wheel selection)、競賽法(Tournament selection)。




## 第三章：研究方法

在本章首先會先介紹一般傳統的基因演算法是怎麼運用來解 Flow Shop 的排程問題，之後再說明本篇論文所採用的染色體表達法跟傳統的基因表達方法有何不同。

### 3.1 研究問題

如同前面 1.3 有提到的，本論文研究的問題是具順序相依整備時間與非固定序列特性之流程式製造單元排程，而由於本論文是希望發展出一套比傳統的染色體表達法更好的一套表達法，所以比較的對象是 Lin et al(2009)這一篇 paper，因此實驗數據全部是依照 Lin et al(2009)中所提到的數據，下方為研究問題描述：

- 
- (1) Family 間的 setup time 有三種，分別為：  
Small setups(SSU) :  $U[1,20]$   
Medium setups(MSU) :  $U[1,50]$   
Large setups(LSU) :  $U[1,100]$
  - (2) Family 數介於 3~10 之間，機台數介於 3~10 之間。
  - (3) Family 和機台數的組合情境有  $\{(3, 3), (3, 4), (4, 4), (5, 5), (5, 6), (6, 5), (6, 6), (8, 8), (10, 8), (10, 10)\}$  共十種，其中括號內的表示為(Family 數，機台數)。
  - (4) 每個 Family 內的工件個數為 2~10 個。
  - (5) 每道加工都僅有一部機台，且各個工件需依序經過每一部機台加工。
  - (6) 各個工件的加工時間為  $U[1,10]$

### 3.2 複雜度分析

若現在是三個機台三個 Family 的情境，假設 Family1、Family2 和 Family3 中所包含的工件數分別是  $n_1$ 、 $n_2$  和  $n_3$ ，那麼全部的組合就有  $(n_1!) * (n_2!) * (n_3!)^3$  種，而如果用通式表達，那麼這個問題的組合就有  $[(n_1!) * (n_2!) * (n_3!) \cdot \dots \cdot (n_F!)]^m$ ，其中  $m$  是機台數， $F$  是 Family 數。

### 3.3 基因演算法求解方法

由圖 3.1 可以看到基因演算法的詳細流程，先隨機產生 1000 組初始解作母代，然後對這 1000 組初始解計算染色體的目標值，接著計算各染色體的適合度函數，判斷是否達停止條件，如果沒有，則計算  $p_c$  判斷是否小於交配率  $P_c$ ，如果是就作交配，如果不是則再計算  $p_M$ ，判斷是否小於突變率  $P_M$ ，如果是則作突變，如果不是則迴圈重覆，整個迴圈一直到判斷是否達停止條件時，整個基因演算法才算結束，而停止條件比照 Lin et al.(2009)，執行上限設定為當執行代數達到  $(m * F + \sum n_i) * 1000 * 44$  時，停止演算法。

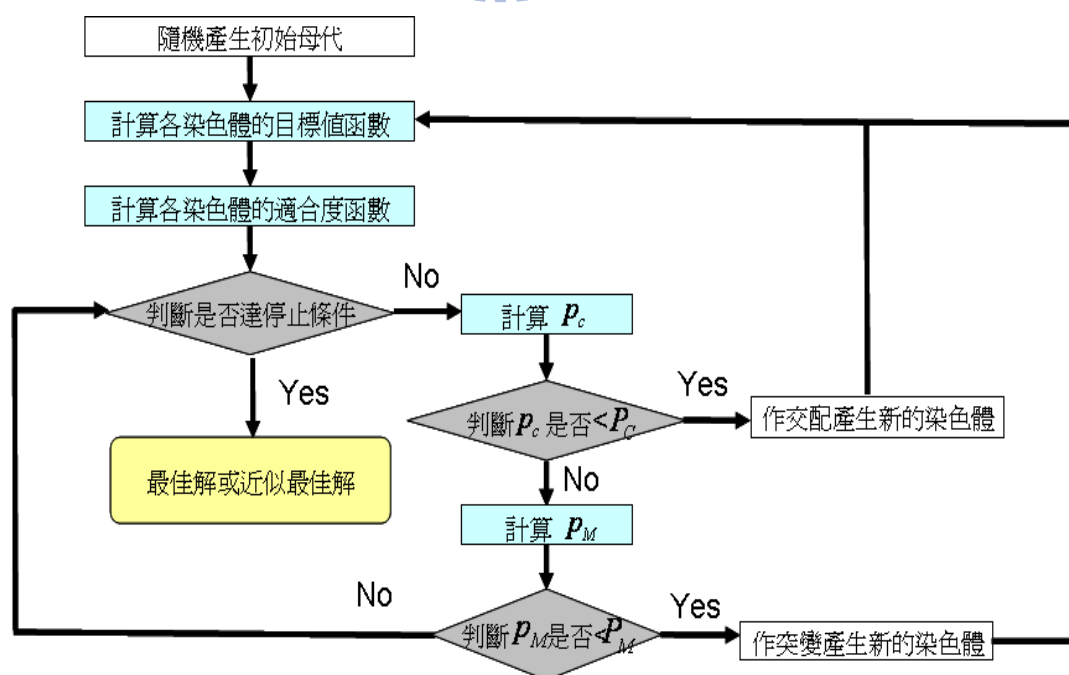


圖 3.1 基因演算法詳細流程



### 3.3.1 染色體設計與解讀

傳統的染色體表達方式如同圖3.2所表示，這個例子是有三部機台三個Family的例子，傳統的染色體解讀是把一條染色體切成六段，前三段是說明各個Family在各個機台的加工順序，而後三段則是分別說明各Family內，各個工件的加工順序。

F1	F2	F3	F2	F3	F1	F1	F2	F3	J1	J3	J2	J6	J4	J5	J7	J8	J9	J10
Family在 M1機台的排序			Family在 M2機台的排序			Family在 M3機台的排序			Job在 F1內的加工排序			Job在 F2內的加工排序			Job在 F3內的加工排序			

圖 3.2 傳統染色體的設計

而本篇論文所採用的染色體是如圖3.3，想法是不希望染色體被切成多條，此概念其實是延續呂佳玟(2009)學姊的論文，該篇論文是僅用一條染色體的表示法，來解讀步進機台加工的排程問題。

J1	J3	J6	J7	J2	J4	J8	J5	J9	J10
----	----	----	----	----	----	----	----	----	-----

圖 3.3 本篇論文所採用染色體的設計

在這個例子中，J1、J2、J3三個是屬於Family 1，而J4、J5、J6是屬於Family2，J7、J8、J9、J10四個則是屬於Family3，而Family在機台1的加工順序就是先做Family1再做Family2最後做Family3，但是這個方法只能找出第一部機台各Family的加工順序，所以本篇論文就在篩選過程多加了一個找出該組合下，Family在各機台下最佳的加工順序的函式，這Family加工順序的找尋方法留待後面說明介紹。

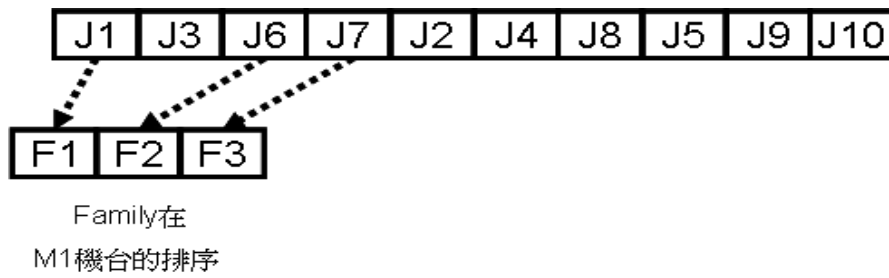


圖 3.4 本篇論文解讀 Family 加工順序

### 3.3.2 適合度函數定義

適合度函數是從目標值作轉換，而本篇論文所採用的目標值函數為：

$$f_j = \frac{1}{1+v_i}$$

， $v_i$ 即是目標值，代入可算出適合度 $f_j$ 。

而本篇論文所採用的目標值是 $C_{\max}$ 而事實上我們知道 Makespan 是越小越好，所以代入適合度函數，可以很清楚地發現， $f_j$  值是越大越好。

### 3.3.3 交配

本篇採用的交配方式是 PBX，而 PBX 的交配示意如圖 3.5 所示。而交配的發生是在 $p_M$  的值小於交配率 $P_M$ ，本篇論文使用的 $P_M = 0.9$ ，而 $p_M$  為服從 U[0.1] 的一個隨機值。

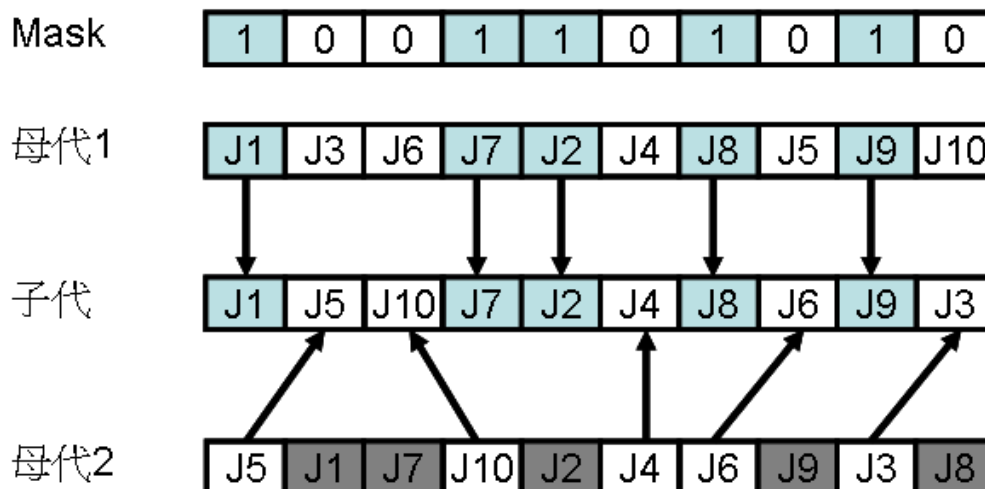


圖 3.5 為 PBX 的交配示意圖

PBX 的交配方式是先從交配池隨機抓出兩條染色體作母代，然後對母代一跑一個 Mask，簡單說就是在各個位子跑 0 或 1，當基因位置為 1 時，母代 1 該位置的基因直接複製到子代上，而剩下的格就對照母代 2，把母代 2 剩下沒有出現在子代的基因由左至右依序填入即可。

### 3.3.4 突變

本篇文所採用到的突變方法總共有兩個，分別是 SWAP 和 Insert。突變的發生是在  $p_c$  的值小於交配率  $P_c$ ，本篇論文使用的  $P_c=0.1$ ，而  $p_c$  為服從  $U[0.1]$  的一個隨機值。

SWAP：圖 3.6 是 SWAP 示意圖，首先先從交配池中隨機選出一條作母代，然後再 random 兩個值，圖例 random 的結果是 3 和 7，所以 SWAP 就會對第三個和第七個位置的這兩個基因作交換，所以原本的上面的這組母代排序就變成了下方的這條子代排序。

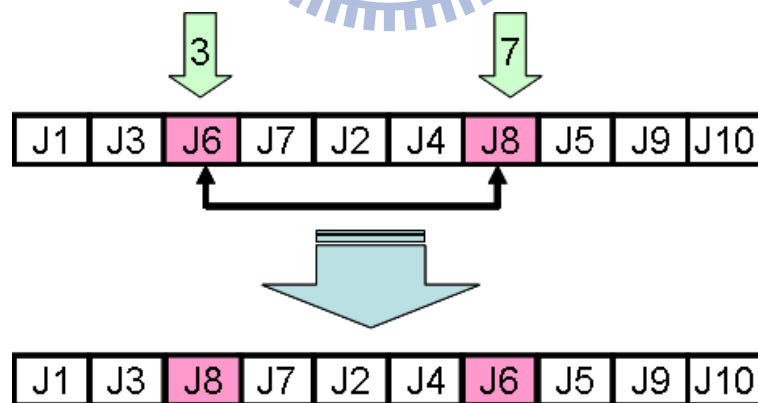


圖 3.6 為 SWAP 示意圖

Insert：圖 3.7 是 insert 示意圖，一開始如同 SWAP 先從交配池中隨機抽出一條染色體當母代，然後再 random 兩個數，圖中的範例是 random 的兩個值是 4 和 8，所以就把第八個位值的基因插入到第四個基因位置的前面，形成下方的子代。

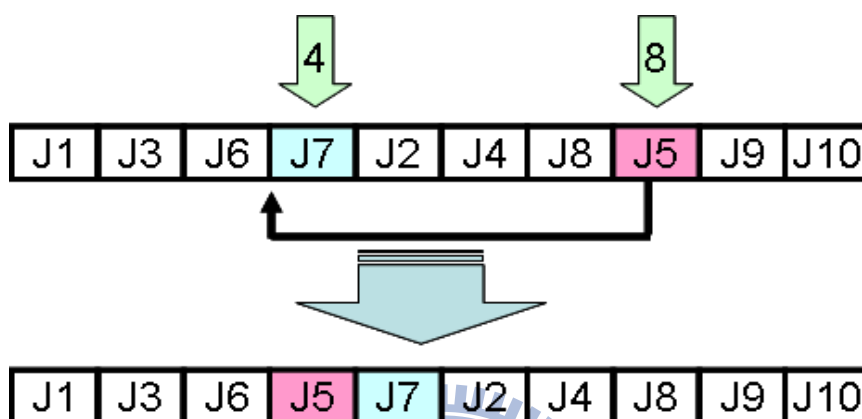


圖 3.7 為 Insert 示意圖

### 3.3.5 傳統基因表達法實際執行方法

在 3.3.1 有提到，傳統的染色體表示法是把一條染色體分成了好多條來做，如圖 3.2 的例子就是把一條染色體切成了六條，如果不管是在作 SWAP、Insert 或是 PBX 都要個別作六次，如此每次不管是作突變或是交配，都會變動非常大，那我們可以合理的認為採用這樣子的方法不僅效率會較差，而且每次的變化太大，可能造成最佳解較難以收斂，以至於較難得到最佳解。

如圖 3.8，不但作法變得繁複，而且們可以看到因為染色體是被切成了六段，尤其在各機台內 Family 的加工順序的改變，作一次交配或突變，就可能如上面的例子，整個加工順序就作了大副變化，如此實際在跑基因演算法時，該解可能會較不易收斂。

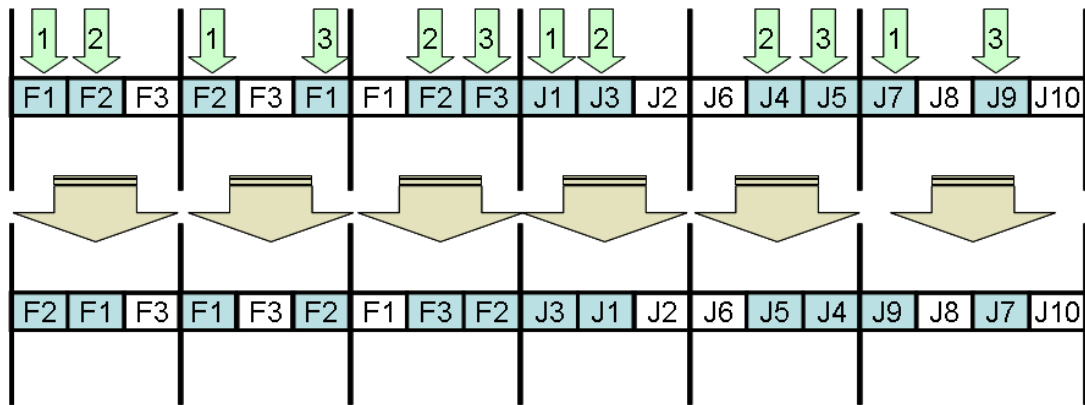


圖 3.8 為傳統的染色體表達法作 SWAP 示意圖

### 3.3.6 新式基因表達法搜尋 Family 加工順序的方法

不過在 3.3.1 也提到，本篇論文的染色體表達法其缺點是只能確定第一部機台的 Family 加工順序，不過我們認為，當第一部機台的 Family 加工順序確定後，之後的每一部機台上加工順序應該僅會依據其上一部機台的排序作一些更動，因為 Family 加工順序的大副度的改變，可能會造成下一部機台的閒置時間被拉長，所以，根據上述觀點，本論文所提出找尋從第二部機台開始至第  $m$  部機台 Family 的派工順序邏輯如下：

我們延續 3.3.1 的例子當 Family 數為 3，機台數為 3 的例子。我們訂各個 Family 中有的工件分別為，Family1 有 Job1、Job2、Job3，Family2 有 Job4、Job5、Job6 以及 Family3 有 Job7、Job8、Job9、Job10，那麼如果我們得到一條染色體如圖 3.9 所示

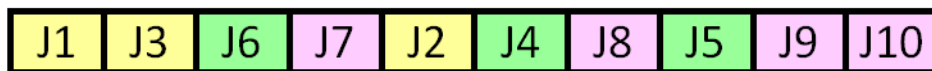


圖 3.9 新式染色體

透過解讀，我們可以得各 Family 在機台 1 上的加工順序為 Family1 => Family2 => Family3 (如圖 3.10)，那經過整理後我們可以得到全部的工件在機台

1 上的加工順序會成為如圖 3.11 所示，把這樣的加工順序派到機台 1 上加工，那在交換 Family 時，中間則會有  $S_{12}^1$  和  $S_{23}^1$  的整備時間(如圖 3.12)而當工件 1 加工完要到機台 2 上加工前，會發生閒置時間(如圖 3.13)，而如果我們還要更換 Family 的加工順序時，將會有更長的閒置時間(如圖 3.14)，而如果產生更大的變動，例如在機台 2 時，Family3 的工件先加工，又將會產生更長的閒置，所以，新式染色體表達法在找尋第二部機台至第  $m$  部機台的 Family 加工順序時，第一個加工的 Family 一定會延續上一部機台的第一個加工的 Family，接著才再找尋各個機台的其它的 Family 加工順序，而由圖 3.15 可以看到，當機台 2 加工完 Family1 的所有工件時，當他考量下一個待加工的 Family 時，其實可以選擇 Family2 或是 Family3，如此就有可能會發生 Family 加工順序互換，而可以達到更好的績效的情況發生，但類似在考量第一個 Family 的加工情況，其 Family 的加工選擇應該也是從鄰近的 Family 來選擇是否交換，所以，Family 加工順序的變化管理應從上一機台的 Family 加工順序作參考，因此各個機台的其它 Family 加工順序則由上一部機台的排序來加以尋找。

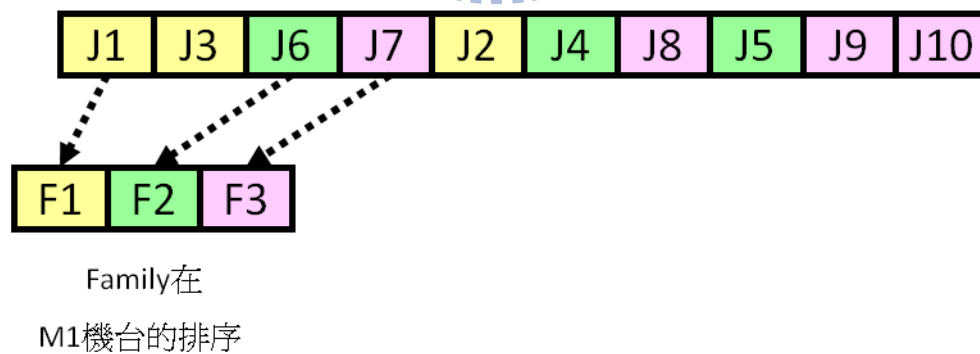


圖 3.10 新式染色體表達法找出機台 1 的 Family 加工順序

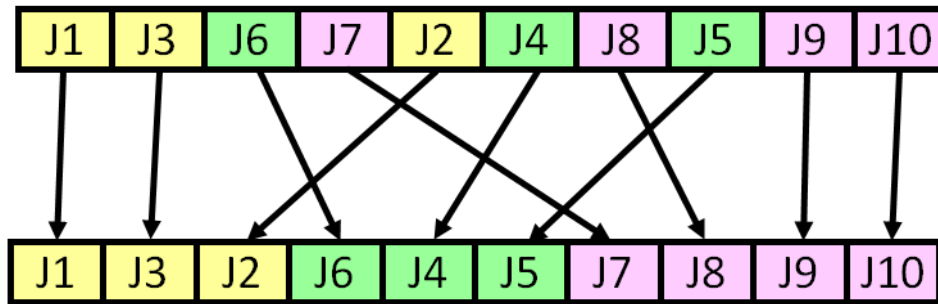


圖 3.11 新式染色體表達法找出機台 1 的各工件加工順序

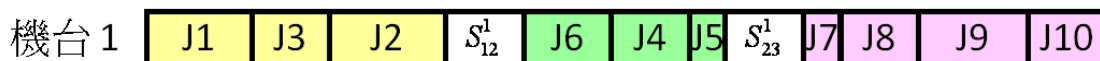


圖 3.12 新式染色體表達法派至機台 1 加工情況

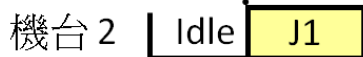
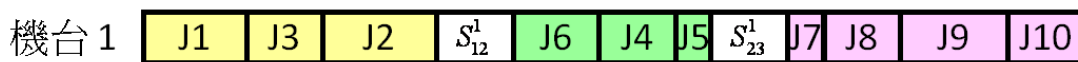


圖 3.13 Job1 在機台 1 加工完接著到機台 2 加工發生閒置的情況

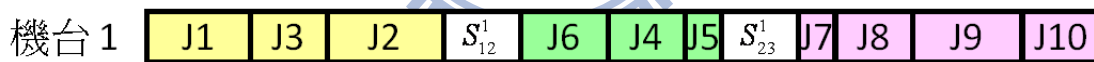


圖 3.14 Job6 在機台 1 加工完接著到機台 2 加工發生閒置的情況

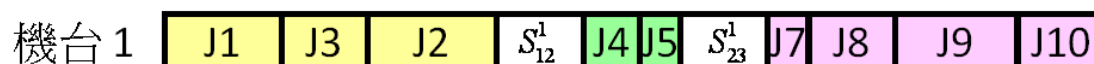


圖 3.15 Family2 和 Family3 在機台 2 加工順序互換的情況

根據以上的想法，再建立新式染色體表達法找尋第二部機台以後各個 Family 的加工順序，以下舉 Family 為 5 個，機台數為 5 個為例：

步驟一：

因為 Family1 是在機台 1 第一個加工的 Family，所以把它保留至機台 2 第一個加工。



圖 3.16 把 Family1 保留到機台 2 第一個加工(步驟一)

步驟二：

圖 3.17 是考慮 Family3 和 Family5 互換後，在機台 2 把所有工件做完時，總時間會不會縮短，假使縮短了，在機台 2 上的加工順序，Family5 就調到 Family3 之前加工。



圖 3.17 考慮 Family3 與 Family5 是否交換(步驟二)

步驟三：

圖 3.18 是考慮，我們已經把 Family5 和 Family3 交換加工順序後，那在加工 Family5 的過程，等我們在決定第三加工順位時，就可以考慮要先加工 Family2 還是 Family3，假使換完後並沒有縮短機台 2 把所有工件完工所花費的時間，則不考慮交換，意即機台 2 作完 Family5 後，接著做 Family3。





圖 3.18 考慮 Family3 與 Family2 是否交換(步驟三)

步驟四：

概念同上，圖 3.19 是在考慮 Family2 和 Family4 加工順序是否需要調換，如果完工時間縮短，則交換加工順序。



圖 3.19 考慮 Family2 與 Family4 是否交換(步驟四)

步驟五：

如此確定全部 Family 在機台 2 上的加工順序(見圖 3.20)。



圖 3.20 確定機台 2 的 Family 加工順序(步驟五)

步驟六：

反覆上述的動作，應用在機台三。

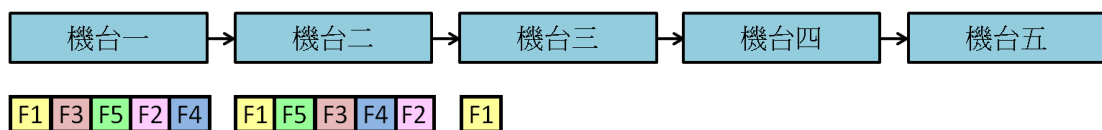


圖 3.21 把 Family1 保留到機台 3 第一個加工(步驟六)

依照上面的步驟，一階段一階段就可以求得各機台上，Family 的最佳加工順序(見圖 3.15)。

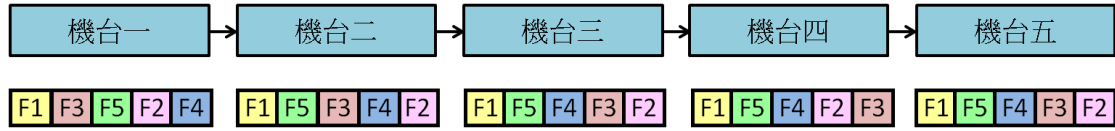
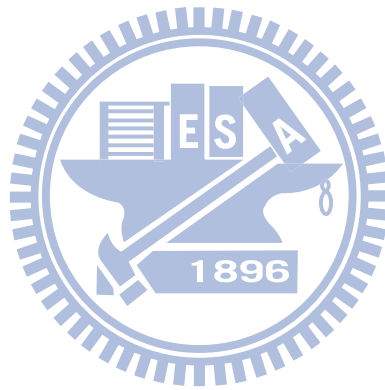


圖 3.22 決定全部機台的 Family 加工順序



## 第四章 實驗情境與結果

### 4.1 實驗情境描述

根據 Lin et al. (2009) 的論文，該篇論文對整備時間分成長(LSU)、中(MSU)、短(SSU)三種，而在各種整備時間底下，對 Family 數目和機台數又作了十種組合，這十種組合分別是{(3, 3), (3, 4), (4, 4), (5, 5), (5, 6), (6, 5), (6, 6), (8, 8), (10, 8), (10, 10)}，針對這 30 種情境，每個情境又給與 30 個問題，即不同的工件數、加工時間長短及整備時間長短的資料，每個問題又各跑 15 個 Seed，對這所有的問題使用基因演算法，比較新式與傳統的染色體表達法，對於  $C_{max}$  也就是 Makespan 的求解品質及求解時間的比較。

以下是實驗數據的 Input 資料隨機產生的分配：

(1) Family 間的 Setup Time 有三種，分別為：

Small setups(SSU) : U[1,20]

Medium setups(MSU) : U[1,50]

Large setups(LSU) : U[1,100]

(2) 每個 Family 內的工件個數為 2~10 個。

(3) 各個工件的加工時間為 U[1,10]

### 4.2 實驗結果

表格中 Nw 是指各 Scenario 中新式表達法比傳統表達法，Makespan 比較小的數目，意即贏的數目，而 Ne 是指各 Scenario 中新式表達法與傳統表達法，其 Makespan 相等的數目，意即平手的數目。Ave.(%)指得是 30 個 Scenario 平均後，新式表達法對傳統表達法改善率，而 Max(%)指得是在 30 個 Scenario 中，贏最多的那一個問題，其改善率。Tn 指得是新式表達法電腦計算時間，而 Tt 指得是傳統表達 電腦計算時間，%Gap 是指新式表達法與傳統表達法，其計算時間差距的百分比。

表 4.1 結果

ID	Scenario	F	M	J	$C_{max}$						Computation Time		
					$N_e$	$N_w+N_e$	$N_e$	$N_w$	Ave. (%)	Max. (%)	$T_n$	$T_t$	%Gap
1	SSU33	3	3	18.33	30	30	30	0	0	0	68.2861	65.55264	-4.17
2	SSU34	3	4	19.43	30	28	24	4	-0.099226	0.268199	99.4919	92.99	-7.00
3	SSU44	4	4	24.47	30	28	10	18	0.302429	1.410416	159.684	149.02084	-7.16
4	SSU55	5	5	30.87	30	30	0	30	2.474348	6.914272	354.28	321.37773	-10.24
5	SSU56	5	6	31.00	30	30	0	30	3.975327	7.41154	447.73	399.77427	-12.00
6	SSU65	6	5	36.87	30	28	0	28	4.086946	7.991397	510.35	448.4578	-13.80
7	SSU66	6	6	36.97	30	30	0	30	5.969102	9.587345	682.067	624.81023	-9.16
8	SSU88	8	8	48.50	30	30	0	30	16.21915	21.18832	1826.15	1538.3972	-18.70
9	SSU108	10	8	60.13	30	30	0	30	19.22867	23.9849	2965.27	2564.8247	-15.61
10	SSU1010	10	10	61.90	30	30	0	30	28.25966	31.11226	4452.48	3769.599	-18.12
11	MSU33	3	3	17.00	30	28	28	0	-0.207902	0	60.8808	58.810767	-3.52
12	MSU34	3	4	18.17	30	26	26	0	-0.333959	0	91.2937	86.211523	-5.89
13	MSU44	4	4	24.17	30	26	6	20	-0.709069	1.098901	159.846	148.27243	-7.81
14	MSU55	5	5	30.80	30	26	0	26	2.197908	6.989958	345.806	310.71593	-11.29
15	MSU56	5	6	29.80	30	29	0	29	4.690535	8.216884	425.099	381.96833	-11.29
16	MSU65	6	5	36.30	30	29	0	29	3.591275	8.665029	508.643	448.5979	-13.38
17	MSU66	6	6	37.03	30	30	0	30	6.633559	11.40618	697.515	603.4259	-15.59
18	MSU88	8	8	48.20	30	30	0	30	16.26439	20.55877	1807.2	1586.9046	-13.88
19	MSU108	10	8	60.87	30	30	0	30	18.74317	25.75287	3062.58	2636.689	-16.15
20	MSU1010	10	10	59.60	30	30	0	30	27.89431	32.6788	4244.67	3741.183	-13.46
21	LSU33	3	3	19.40	30	22	22	0	-1.801714	0	72.6511	69.44966	-4.61
22	LSU34	3	4	17.33	30	23	23	0	-2.191171	0	84.9997	80.94966	-5.00
23	LSU44	4	4	23.13	30	12	5	7	-2.523685	0.393185	151.589	141.12106	-7.42
24	LSU55	5	5	30.00	30	18	1	17	-0.0887	5.509908	336.035	301.63283	-11.41
25	LSU56	5	6	30.47	30	25	0	25	2.600417	11.6282	439.526	392.44647	-12.00
26	LSU65	6	5	36.83	30	21	0	21	1.8548	8.495279	519.761	476.118	-9.17
27	LSU66	6	6	35.53	30	27	0	27	4.656648	10.87554	649.93	570.70377	-13.88
28	LSU88	8	8	46.77	30	30	0	30	14.62946	19.97439	1838.6	1593.9622	-15.35
29	LSU108	10	8	58.67	30	30	0	30	17.79831	21.88346	2961.63	2559.8613	-15.70
30	LSU1010	10	10	59.53	30	30	0	30	26.75114	30.99131	4337.92	3739.6537	-16.00
Average of all Scenarios					<b>30</b>	<b>27.20</b>	<b>5.83</b>	<b>21.37</b>	<b>7.36</b>	<b>11.17</b>	<b>1145.40</b>	<b>996.78</b>	<b>-11.29</b>

Ave(%)、Max(%)與%Gap 的計算公式皆為  $R = \frac{\text{傳統} - \text{新式}}{\text{傳統}} \times 100\%$ 。

從結果看來，情境規模越大即 Family 數與機台數越大，兩者的計算時間都是越長，而新式的表達法其計算時間又比傳統表達法慢，不過從目標值 Makespan 來看，當情境的規模越大，新式表達法改善的幅度越大，雖然在比較小的情境(如(3,3)(3,4)(4,4)(5,5))新式表達法求解品質較傳統差一些，但是，當規模變大，其平均改善率甚至可以高達兩成，其改善成果是相當顯著的。

由圖 4.1 到圖 4.6，可以較清楚得看出不管是 SSU、MSU 或是 LSU 新式染色體解讀法在大規模時會較傳統染色體表達法來得好。

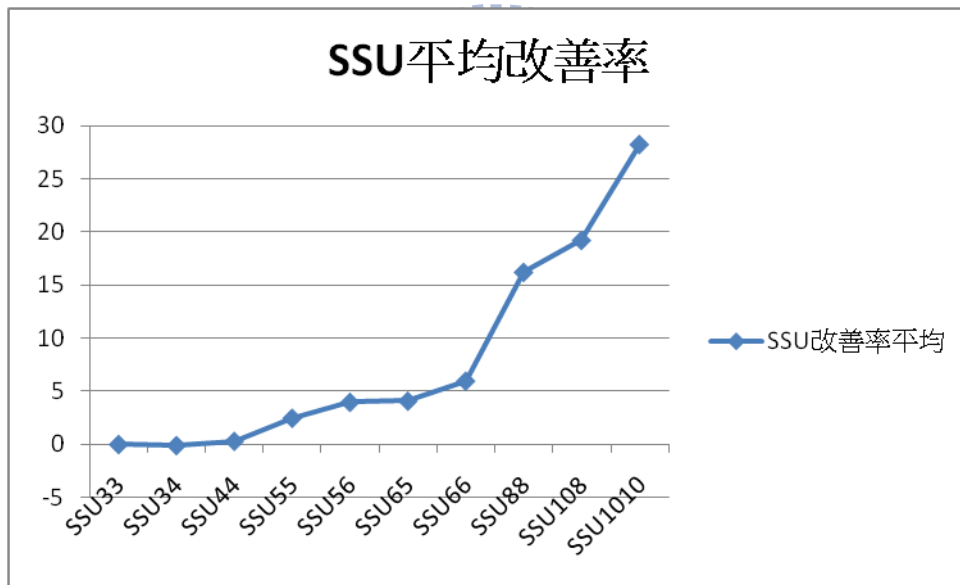


圖 4.1 SSU 平均改善率

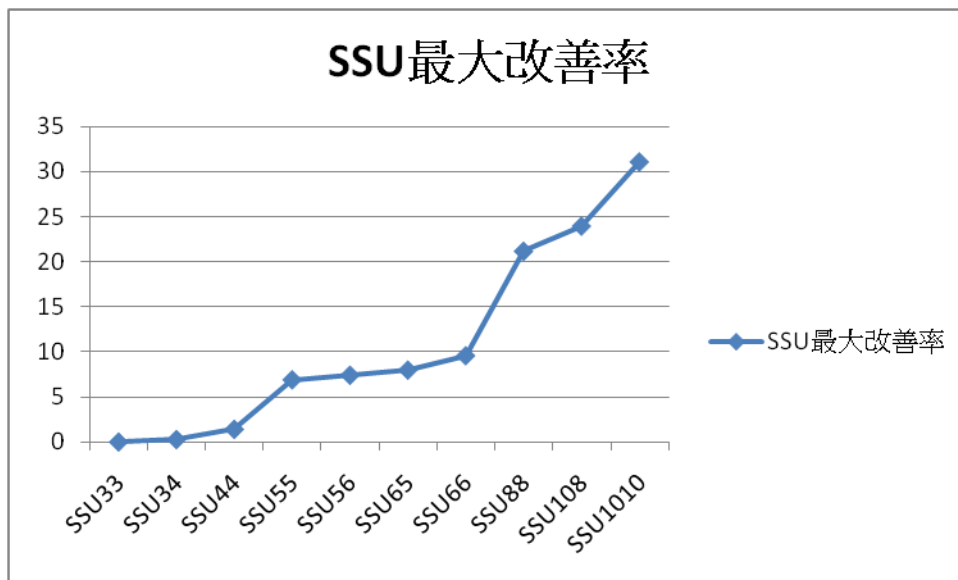


圖 4.2 SSU 最大改善率

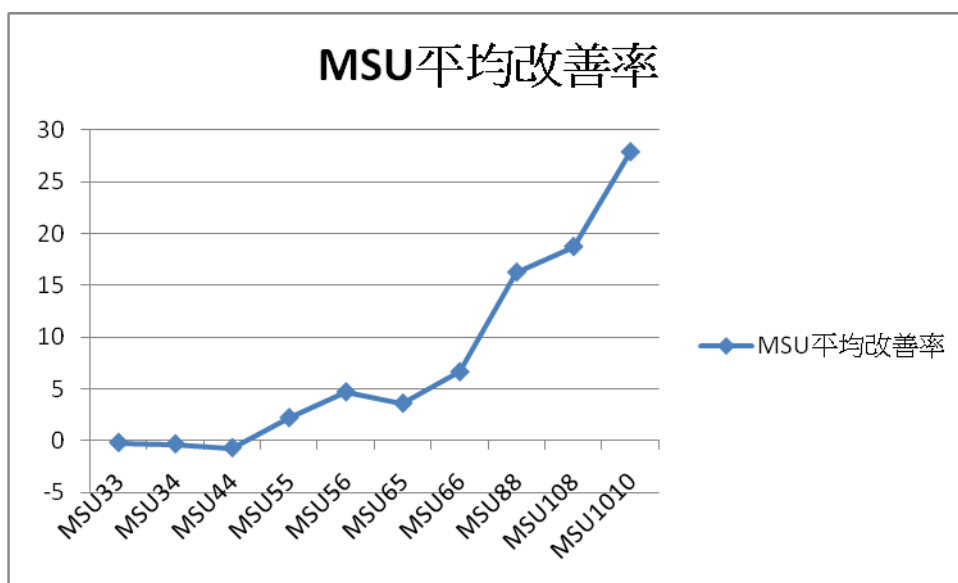


圖 4.3 MSU 平均改善率

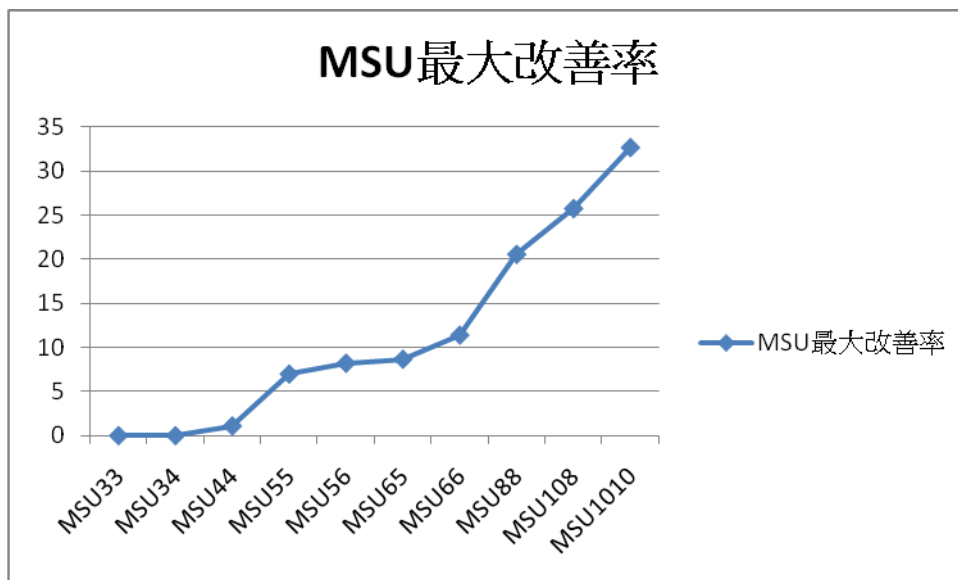


圖 4.4 MSU 最大改善率

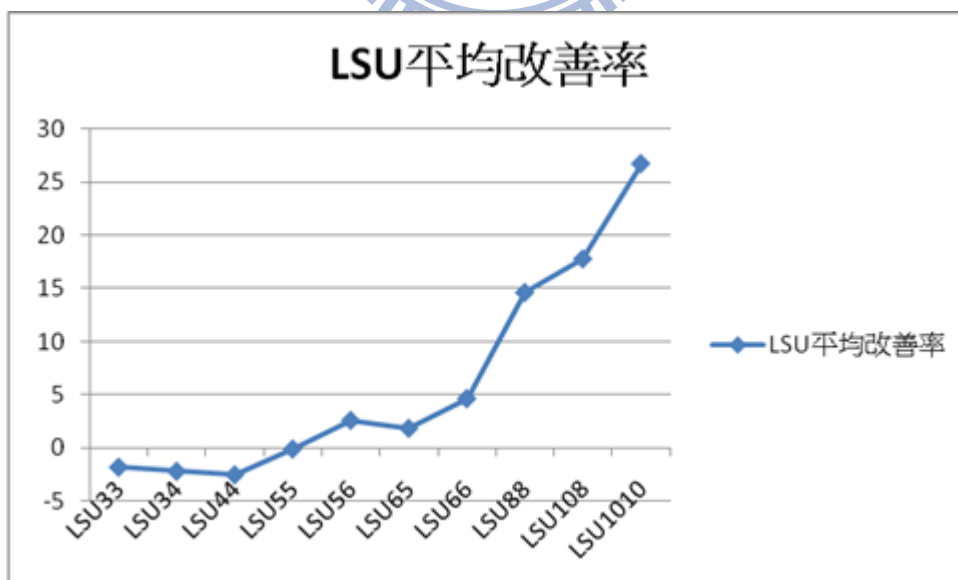


圖 4.5 LSU 平均改善率

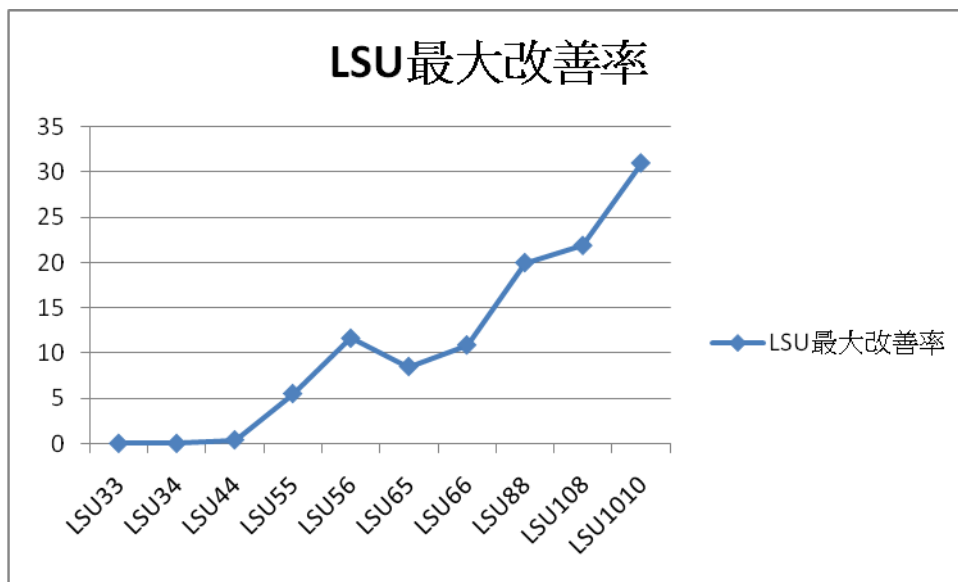


圖 4.6 LSU 最大改善率

### 4.3 小規模新式表達法輸的原因

新式表達法其求解雖然是按照邏輯去搜尋 Family 的加工排序，但由於是一階段一階段下去搜尋，所以，其實蠻容易理解，我們所找到的解可能成為區域的最佳解，按各個階段一步一步搜尋時，確實是蠻快且蠻合乎邏輯的搜尋方式，但是，若是 Family 的加工順序互換是發生在較後方的機台，會比發生在較早開始的機台時，其整體 Makespan 會更小，這個時候，新式的表達法就可能無法找到這個好的解。

以圖 4.1 與圖 4.2 為例，假使最佳的 Family 加工排序是如圖 4.1 所呈現，雖然 Family 的加工排序只有發生在機台四，但是對整體而言最擁有最低的 Makespan，而如果我們使用新式的表達法去搜尋 Family 加工排序時，我們可能在機台 2 時就把 Family3 和 Family2 作交換，因為新式的表達法是以一階段一階段來作搜尋，所以一旦在在機台 2 作了交換後，就沒辦法達到機台 4 的才作 Family 順序交換的以得到最佳解，因此使得新式的表達法可能有這樣的缺點。

但是當情境變大時，新式表達法卻能很快的收斂得到一個較好的解，而傳統的表達法，其收斂效果較差，得到的解就沒有比新式的表達法來得好。



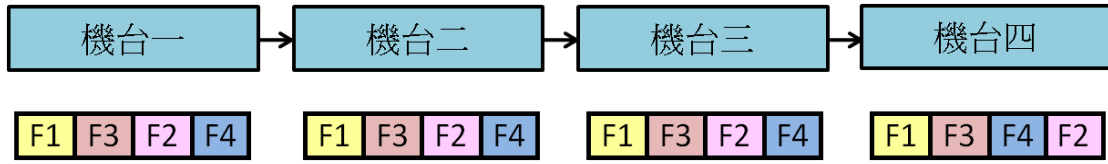


圖 4.7 Family 在各機台最佳排序(例子)

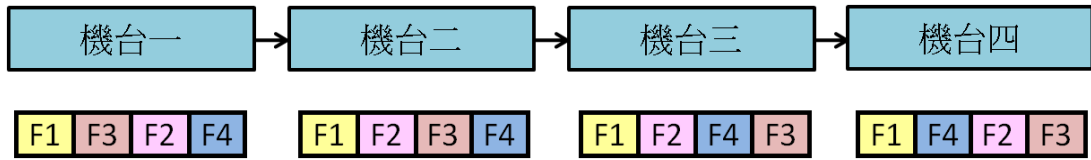


圖 4.8 新式表達法所能求解的最佳 Family 加工排序(例子)



## 第五章 結論與未來建議

本論文所採用的目標值是 Makespan，而未來可以使用別的目標值作為新的績效指標來評估，例如延遲時間(Tardiness)。而本篇論文針對的是 Flow Shop 的排程問題來作改善，未來，可以從不同的生產型態來做延伸，像是 Job Shop 的排程問題，相信也是非常值得研究的議題。

不過其實從本論文的數據看來，在小情境下仍是有些缺陷，雖然在大情境，本篇論文有相當大的貢獻，但是，由於沒辦法通用所有情境，所以未來也能從這個部分試試有沒有什麼改善方法。

而在求解的效率上，不管是傳統的染色體解讀法或是新式的染色體解讀法，都是花費蠻長的電腦計算時間，所以，也可以嘗試看看有沒有更好的解讀方式，能夠同時提升電腦計算時間，亦能同時讓各情境下皆有好的表現。



## 參考文獻

Franca, P. M, Gupta, Jatinder N.D, Mendes, A. S, Moscato, P., Veltink, K. J, “Evolutionary algorithms for scheduling a flowshop manufacturing cell with sequence dependent family setups”, *Computers & Industrial Engineering*,48,pp.491-506,2005.

Kuo-Ching Ying, Jatinder N.D. Gupta , Shih- Wei Lin and Zen-Jung Lee, “Permutation and non-permutation schedules for the flowline manufacturing cell with sequence dependent family setups”, *International Journal of Production Research*, 2009, 1-16, iFirst

Kuo-Ching Ying and Shih-Wei Lin “Multi-heuristic desirability ant colony system heuristic for non-permutation flowshop scheduling problems” Volume 33, Numbers ,2007

Liao, C. J., Liao, L. M., Tseng, C. T., “A performance evaluation of permutation vs. non-permutation schedules in a flowshop”, *International Journal of Production Research*, 44, 20, pp. 4297–4309, 2006.

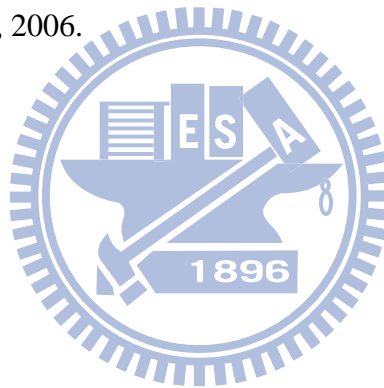
Lin, S. W, Gupta, Jatinder N.D,Ying, K. C, Lee, Z. J, “Using simulated annealing to schedule a flowshop manufacturing cell with sequence-dependent family setup times”, *International Journal of Production Research*,1,pp.1-13,2008.

Lin, S. W, Ying, K. C, Lee, Z. J, “Metaheuristics for scheduling a non-permutation flowline manufacturing cell with sequence dependent family setup times ”, *Computers & Operations Research* 36(2009)1110-1121

Lin, S. W, Ying, K. C, Lee, Z. J, In press. “Metaheuristics for scheduling a non-permutation flowline manufacturing cell with sequence dependent family setup times”,*Computers & Operations Research*,2008.

Ruiz, R. , Stutzle, T., “A simple and effective iterated greedy algorithm for the permutation flowshop scheduling problem”, *European Journal of Operational Research* 177, pp.2033–2049, 2007.

Ruiz, R., Maroto, C., “A genetic algorithm for hybrid flowshops with sequence dependent setup times and machine eligibility,” *European Journal of Operational Research*, 169, pp.781–800, 2006.



呂佳玟，“應用基因演算法與家族式派工於傳輸整合步進機在小批量情境下之排程問題”，交通大學，工業工程與管理學碩士，2009

李師尹，“應用基因演算法於求解模糊環境下之單元製造系統問題”，大葉大學，工業工程研究所碩士，2002

