多階平行機器零工式多目標排程之模式化與系統之研究

研究生：林純行　　指導教授：張炳騰 博士

東海大學工業工程與經營資訊研究所

摘要

傳統 Job shop 排程模式僅能涵蓋多階單機的生產型態，然而，平行機器(parallel-machine)的生產方式目前已被廣泛的運用於各產業中，許多零工式生產或流線式生產中的製程也都利用平行機器的安排，來增加產能、減低瓶頸工作站負荷和減少交貨延遲時間。現實世界的排程問題並非單一目標可以滿足，需以多目標(multi-objective)的觀點來考慮，而且目標之間每每是彼此衝突。而目前多數關於多目標排程的研究中，僅考慮生產製造方面的績效因子，這些因子多是屬於量性因素(quantitative factors)。事實上，排程環境中所需考量的還需包含有一些關於組織營運策略上的質性因素(qualitative factors)。

本研究將利用遺傳演算法(genetic algorithm)與螞蟻演算法(ant algorithm)優異的搜尋求解功能，建構一以遺傳演算法與螞蟻演算法為基礎考量量性與質性因素的多階平行機器多目標排程系統。排程演算法允許訂單分割，採動態的均勻批量分割(lot splitting)，使一作業可同時在多部機器處理，縮短各作業完成時間。同時給予一修正模式，調整作業分割批量的加工時間，使其能夠彼此重疊而達到生產連續，進而縮短總生產時間。至於適應性函數(fitness function)的設計使用階層次分析法(Analytic Hierarchy Process)將所有的因子做合理權重制定。最後透過 Job shop 排程實證問題，驗證本研究所提方法與求解模式的可行性及分析衡量此系統帶來的效益。

A Study of Modeling and the Scheduling System for Multi-Objective, Multi-Stage Parallel-Machine Job Shop

Student：Chun-Hsing Lin　　Advisor：Dr. Ping-Teng Chang

Institute of Industrial Engineering & Enterprise Information

Tunghai University

Abstract

Conventional job-shop scheduling system can be applied well to multi-stage, single machine producing mode. Nevertheless, parallel-machine producing mode has been generally adopted within either job-shop or flow-shop scheduling system, to enlarge capability, avoid delivery delay, and lighten the load of bottleneck station. There is no more a singular objective in real world scheduling system, but multi objectives that are commonly conflicting to each other. In addition, the effect factors taken account by current multi-objective scheduling research are quantitative factors. Essentially there are more qualitative fators to be considered related to organizations' operating tactics.

This research utilizes genetic algorithm and ant algorithm with their exceptional searching and solutioning function to construct a modeling and scheduling system for multi-stage, parallel-machine, multi-objective job shop that takes account of both quatitative and qualitative factors. This scheduling system allows orders' evenly and actively lot splitting, that is attainable in plenty machines at one time and operations can be accomplished early. In the meantime, the scheduling system offers an amending mold that can adjust the lot splitting process to make each operation join or overlap with each other, and therefore, the continuing operating procedure abbreviates overall working process. As for the equipment of fitness function, it defines all factors' reasonable merits in analytic hierarchy process. Eventually, by concrete testimony through job shop scheduling, it is certified the feasibility of the solution and the benefitual results brought up in analyzing and examining this scheuduling system.

誌謝

鳳凰花開，又到了畢業時節，而歷經了許多低潮時期，終於順利完成了傳說中的論文。回想兩年的研究生涯，首先要感謝指導教授張炳騰博士在學業上的細心指導與對於人生的諸多指引，由衷的感謝老師的教導。

口試期間，承蒙洪堯勳博士與白炳豐博士特別撥冗審閱論文，提供寶貴的意見，使得論文的內容更為完備，在此致上最深的謝意。

在研究所兩年中，感謝同學郁文、嘉偉，多少的日子我們一起披荊斬棘地完成報告，相互扶持地走過最艱難的時光，完成不可能的任務。此外，感謝晴翔與國禎學長，在學業及生活上所給予的支援。以及感謝研究室學弟千展、阿弟、明修、家政，在口試期間的幫忙與在平時提供精神食糧。

最後更感謝我的父母、家人與 Amber，讓我在無憂無慮的環境中完成學業，同時在精神上給我的支持與鼓勵。

畢業在即，將要離開生活六年的東海校園，回想其間的點點滴滴、酸甜苦辣，教人永誌心中，無法忘懷。在此，僅以這份研究成果獻給所有關心我的人，謝謝你們。

林純行 謹誌於

東海大學工業工程與經營資訊研究所

民國九十二年六月

目錄

中文摘要..... I
英文摘要..... II
誌謝..... III
目錄..... IV
表目錄..... VI
圖目錄..... VII
第一章 緒論..... 1
1.1 研究動機與背景..... 1
1.2 研究目的..... 2
1.3 研究假設與範圍..... 2
1.4 研究方法與步驟..... 3
1.5 論文架構..... 4
第二章 文獻探討..... 5
2.1 排程問題之描述..... 5
2.1.1 排程問題之分類..... 5
2.1.2 多目標排程..... 6
2.2 平行機器排程..... 7
2.2.1 平行機器的類型..... 8
2.2.2 單階平行機器排程..... 9
2.2.3 多階平行機器排程..... 11
2.3 遺傳演算法..... 11
2.3.1 常見 Job shop 排程問題的編碼表示法..... 13
2.3.2 常用排序問題之運算子..... 17
2.3.3 排程中遺傳演算法之應用..... 21
2.4 螞蟻系統..... 22
2.4.1 螞蟻系統介紹..... 23
2.4.2 實行螞蟻演算法於 Job shop 排程問題..... 26
2.4.3 排程中螞蟻演算法之應用..... 28
第三章 多階平行機器零工式多目標排程系統架構設計..... 29
3.1 多階平行機器排程法則..... 32
3.1.1 排程演算法..... 32
3.1.2 作業時間修正模式..... 35
3.2 定性因素模式架構..... 39
3.2.1 AHP 法的評估計算..... 41
3.3 定量因素模式架構..... 43

3.3.1 製距績效評估.....43
3.3.2 交期滿足度評估.....44
3.3.3 機器使用率評估.....45
3.4 遺傳演算法之運作.....45
3.4.1 編碼表示法.....45
3.4.2 初始族群的產生.....46
3.4.3 適應性函數之設計.....48
3.4.4 交配運算子設計.....50
3.4.5 突變運算子設計.....51
3.4.6 育種選擇.....52
3.5 螞蟻演算法之運作.....54
3.5.1 多階平行機器排程問題的圖示.....54
3.5.2 記憶陣列之設計.....55
3.5.3 轉移機率規則.....57
3.5.4 費洛蒙更新法則.....59
第四章 系統實證.....61
4.1 實證問題說明.....61
4.2 定性因素評估.....61
4.3 基本參數設定.....68
4.4 遺傳演算法與螞蟻演算法之檢定.....69
4.5 運作實例探討.....72
第五章 結論與未來研究方向.....79
5.1 結論.....79
5.2 未來研究方向.....80
參考文獻.....82

表目錄

表 2.1 排程績效目標.....7
表 2.2 多目標排程文獻整理.....8
表 2.3 單階平行機器排程文獻整理.....10
表 2.4 遺傳演算法之優缺點歸納表.....22
表 2.5 螞蟻系統文獻整理.....23
表 3.1 定性與定量因素表.....29
表 3.2 AHP 評估尺度.....41
表 3.3 各因素之權重制訂.....42
表 3.4 各方案之評估值.....42
表 3.5 各方案之加權排序.....42
表 3.6 訂單平均基因順序.....49
表 3.7 懲罰函數之計算.....49
表 4.1 訂單資料.....62
表 4.2 平行機器數目.....63
表 4.3 定性與定量之權重值.....64
表 4.4 定性因素之權重值.....64
表 4.5 定量因素之權重值.....64
表 4.6 定性因素-各訂單利潤之評估值.....65
表 4.7 定性因素-各訂單顧客歷史交易之評估值.....65
表 4.8 定性因素-各訂單市場考量之評估值.....66
表 4.9 定性因素-各訂單顧客潛在訂單之評估值.....66
表 4.10 各訂單定性因素之評估值.....67
表 4.11 遺傳演算法中參數之設定值.....68
表 4.12 螞蟻演算法中參數之設定值.....69
表 4.13 相同搜尋時間 GA 與螞蟻演算法之最好排程解.....71
表 4.14 GA 排程結果.....72
表 4.15 機器之作業排程.....74
表 4.16 訂單交期滿足.....77
表 4.17 機器使用率.....77
表 4.18 訂單的平均基因順序.....77
表 4.19 搜尋結果所得之訂單順序與定性因素下訂單順序差異.....77

圖目錄

圖 1.1 研究流程圖.....4
圖 2.1 一般遺傳演算法運作流程.....13
圖 2.2 分支圖.....16
圖 2.3 螞蟻之移動行為模式.....24
圖 2.4 Job shop 排程問題的圖示 27
圖 3.1 零工式生產系統示意圖.....30
圖 3.2 多階平行機器多目標排程系統架構圖.....31
圖 3.3 訂單批量分割的現象.....32
圖 3.4 作業的排程流程圖.....36
圖 3.5 原始排程甘特圖.....37
圖 3.6 作業的時間修正流程圖.....39
圖 3.7 修正排程甘特圖.....40
圖 3.8 交期之模糊隸屬函數.....44
圖 3.9 編碼表示法示意圖.....46
圖 3.10 上半部基因初始族群示意圖.....46
圖 3.11 下半部基因初始族群示意圖.....47
圖 3.12 交配示意圖.....51
圖 3.13 突變示意圖.....52
圖 3.14 多階平行機器排程問題的圖示.....54
圖 4.1 遺傳演算法適應函數值趨勢圖.....70
圖 4.2 螞蟻演算法適應函數值趨勢圖.....70

第一章 緒論

1.1 研究動機與背景

在關於作業排程的領域中，一直都有許多研究投入其中，可是真正能落實應用於實務的非常的少。探究其主要原因，大多數排程研究都將現實世界的加工機器分佈情形過度簡化，定義在傳統的單階多機與多階單機生產型態上，但事實並非如此。現今絕大多數的產品需經過多階段的加工步驟才能生產完成，而為了提高產能與減少交貨延遲時間，平行機器的生產方式被廣泛地運用於各產業的生產製造環境中；而少數有針對特定的多階平行機器排程問題所發展出的啟發式派工法則又不具通用性，且無法全盤考量作業的排程組合，致使其排程解不甚理想。本研究中擬以零工式生產型態為基礎，加入平行機器的考量，並允許作業批量分割，使一作業可同時在多部機器處理，縮短作業完成時間。另外為了更接近現實情形，使現場機器能夠生產連續，研究中也將調整各訂單的預排作業時間，使隸屬同一作業的子批量加工時間能盡量靠近。

此外，綜觀近年來有關生產排程的文獻，我們發現多數的排程研究是提出在某一衡量指標之下求得最佳解或是近似最佳解的演算法。可是在現實的製造環境中，所在乎的往往並非是單一的目標，而是彼此相互衝突的多重目標。但是，在多目標排程的文獻中，大都皆僅是考量生產製造上的多重績效衡量，鮮少將公司的策略、顧客重要性所應考量的因素納入。如此一來，雖然生產製造部門所制定的排程有良好的現場績效，但以營運策略、市場行銷的觀點來看卻不盡理想，往往會導致生產與行銷兩部門的衝突，尤其是在充滿相互矛盾因素的環境中。所以，本研究將提出一套以遺傳演算法與螞蟻演算法為基礎的排程方法，除了能考量生產現場績效方面的定量因素之外，還將策略類型的定性因素納入考量，構成一真正完整的多目標排程。

1.2 研究目的

隨著電腦計算能力的增強，許多的求解的搜尋演算法、人工智慧的工具都已發展的相當成熟，然若要將其應用於實務界，始終會存有障礙，排程問題尤是如此；另外，大多數的排程研究均將工廠中的機器分佈型態過度簡化，致使無法真正評估出整體生產系統的最佳排程結果，因此雖然有關排程的研究為數眾多，但是在一般工廠中，仍多以簡單的派工法則做為排程規劃的依據。所以，本研究的目的擬提出一以遺傳演算法與螞蟻演算法為基礎的排程方法。同時，考量現實環境中質性與量性兩種類型因素的多目標訴求。此外，納入多階平行機器生產模式，並讓排程演算法在運作的過程中能自動分割訂單批量，決定出最佳的批量大小與佔有的平行機器數，期發展一能應用於實務的排程之系統。

基於上述，本研究的重點如下：

1. 發展一般化多階平行機器零工式排程模式。
2. 提出一考量行銷策略方面的質性因素與生產績效方面量性因素之多目標排程系統。
3. 將批量分割納入考量，讓整個排程系統更具有真實性與完整性。
4. 提供以遺傳演算法與螞蟻演算法求解具多階平行機器型態多目標排程問題時的設計方式。

1.3 研究假設與範圍

現場排程的目的是藉著分配與協調，來妥善利用有限資源（包括機器、人員、物料、工具等），以滿足生產策略（如訂單交期、前置時間最短、產量最大等）。現場排程若以機器數目配合工作劃分可將排程問題分成數種不同複雜程度的情形[2]，其複雜程度依序如下：

1.單機-單階

2.多機-多階

3.多階-流程式工廠(Flow shop)

4.多階-零工式工廠(Job shop)

這四種生產方式以 Job shop 類型最為複雜，屬於 NP-hard[50]的問題。本研究將以 Job shop 生產類型的排程問題做為研究範圍，同時，每一種類的機器對於不同的產品來說，均有批量的限制，研究中將制定上機批量數來決定是否開機加工。

1.4 研究方法與步驟

本研究主要是透過電腦實驗的方式進行研究，實驗的對象為一假設的 Job shop 排程問題，以此排程問題對本研究所提的遺傳演算法與螞蟻演算法在多目標衡量下分為兩部分各進行實驗，最後再依據實驗數據與結果作分析討論。本研究主要可分為三個步驟：

1. 多目標排程架構之設計

確認排程環境中的多重目標因素，並將其區分為定性與定量二種類型。利用 AHP 法評估定性因素，同時制定量性與質性因素的權重。依據選擇的因素設計演算過程中之多目標函數。建構以遺傳演算法與螞蟻演算法為基礎的多目標排程系統。

2. 多階平行機器模式之建立

在此部分中，對加工作業採取批量分割，並局部調整子批量加工時間，提出多階平行機器的作業排程演算法。

3. 電腦實驗

在此階段中，以步驟 1、2 中所設計的遺傳演算法與螞蟻演算法，進行模擬實驗，藉由實驗數據的分析以驗證本研究之系統架構。

1.5 論文架構

本研究論文的內容共分為五章：第一章說明本研究之動機、目的、範圍、假設、方法與步驟等相關內容；第二章則針對本論文內容所涉及之相關文獻加以探討。包括遺傳演算法、螞蟻演算法、多目標問題及平行機器排程之相關文獻；第三章則根據本論文之目的與文獻所得之啟發，提出遺傳演算法與螞蟻演算法的多階平行機器零工式多目標排程系統架構，詳細闡述系統架構中各項機制之功能設計與運作方式；第四章為論文中系統之實驗結果；第五章則根據本研究所得之結果，說明研究結論與未來發展方向。本論文的進行流程如圖 1.1。

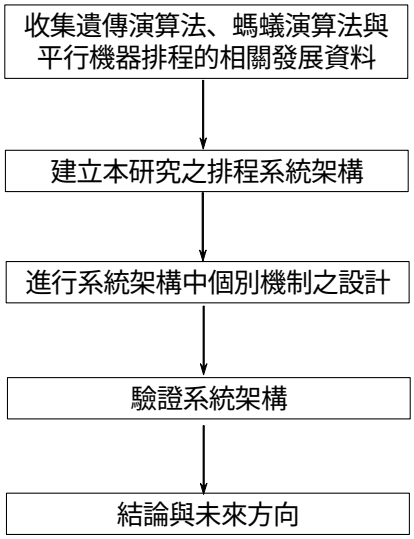

圖 1.1 研究流程圖

第二章 文獻探討

2.1 排程問題之描述

所謂排程是在有限資源的限制下，將訂單、或生產計畫轉換成生產活動的過程，考慮生產活動中各個時間點的最佳配置。換言之，排程問題是決定所有工作在機器上加工的起始時間以及加工順序，所以排程問題乃是同時考慮資源限制和執行限制之最佳化問題。

2.1.1 排程問題之分類

在[2]指出排程問題可以依下列四項資訊加以分類

1.工作及作業之特性：在排程問題中，若訂單數目及作業內容均為固定且為已知，則此種排程問題稱為靜態(static)排程問題；反之，若訂單數目或作業內容會隨時間之變動而改變，則成為動態(dynamic)排程問題。此外，若訂單在機器上的加工時間為固定，則稱此類排程問題具有確定性(deterministic)；反之，若訂單在機器上的加工時間不固定，為某種機率分配，則稱此類排程問題具有隨機性(stochastic)。

2.以工廠中的機器數目及型態可分為

(1)單機單階：一台機器，且訂單只有一個作業。

(2)多機多階：有多部相同的機器，每張訂單在機器需進行多個作業。

(3)多階流程式工廠(Flow shop)：所謂流程式工廠是一多階作業的加工程序，若是所有訂單的機器途程均相同，則此流程型態稱之為淨(Pure)流程式工廠。但若是某些訂單可以跳過某些作業，則稱為非淨(Unpure)流程式工廠。

(4)多階零工式工廠(Job shop)：相較於流程式工廠，在零工式的生產環境每一訂單各有其加工順序。

3.排程之技術限制：此類限制規則定義了訂單受限於機器的特性，例如

(1)一台機器一次僅能加工一個訂單。

(2)訂單沒有一定的加工途程。

4.衡量標準：排程品質之衡量大致可分為兩類

(1)尋求成本最小：常用的衡量標準有生產的固定成本、整備成本、變動成本、存貨成本、延遲成本、與缺貨成本等。運用此類衡量標準必須制訂成本函數始能以成本衡量，而成本函數的設定則較具主觀性。

(2)尋求績效最佳：因衡量績效目標不同所產生排程解亦不盡相同，常見的交期滿足、最小化總製距時間、最小化平均流程時間、最小化最大延遲時間…等，Mellor[38]提出 27 種排程可能考慮的目標，如表 2.1。

2.1.2 多目標排程

關於多目標排程的文獻中[17][32][33][35][41][44][46]羅友廷[4]整理如表 2.2，其所關注的目標多是針對製造現場的生產績效指標，如製距、流程時間、機器使用率、最小延遲時間…等，Baker[8]指出有關這些排程的績效指標大致可以分成兩種類型：現場時間績效(shop time performance)與交期滿足績效(due date performance)。而Brown[11]則提出在整個多目標的環境中，對於在規劃生產排程時應考量市場(marketing)與生產(production)兩大類型的因素。在求解方面，Kim[35]與 Min[41]是以類神經收集現場狀態搭配派工法則來訓練、建構一排程系統。[32][44][46]的研究中採用遺傳演算法來進行搜尋求解，其著重於演算法中各運算子的設計使得求解過程能更有效率，[32][44]為了避免限定演算的搜尋方向，在其研究中的權重是以隨機方式產生。

表 2.1 排程績效目標

1. 設備閒置時間最小化(minimum idle facility investment)
2. 在製品數量最小化(minimum in-process inventory)
3. 設備整備成本最小化(minimum facility set-up cost)
4. 平穩化勞動量(day to day stability of work force)
5. 交期滿足(adherence to promised shipping date)
6. 產出率最大化(maximum output(product rate))
7. 物料持有成本最小化(minimum materials-handling cost)
8. 訂單優先次序的滿足(adherence to arbitrary job priorities), such as arise in dealing with preferred customers, emergency repair parts, etc.
9. 加工合理性(technological feasibility)
10. 變換生產時的敏感度(sensitivity to possible production change)
11. 彈性(general flexibility)
12. (non-dependence on unreliable process)
13. 保留生產力予緊急插單(reverse capacity for rush order)
14. 廠內運送排程最佳化(optimal in-plant transportation schedule)
15. 運送成本最小化(minimum shipping cost)
16. 預期成本最小化(minimum total expected costs, primarily in theoretical investigations)
17. 設備使用率最大化(maximum weighted facility utilization)
18. 人力使用率最大化(maximum utilization of manpower)
19. 勞工績效(optimal assignment of various labor grades)
20. 原物料存貨最小化(minimum raw material in inventory)
21. 最終成品最小化(minimum finished product inventory)
22. 存貨投資最小化(minimum investment in inventories)
23. (minimum obsolescence and deterioration of product)
24. 特定產品製距最小(shortest make-span for certain products)
25. 製距最小化(minimum overall fabrication span)
26. 損失風險最小化(minimum risk of excessive losses)
27. 預防價格的改變(anticipated changes in price)

資料來源：Mellor[38]

2.2 平行機器排程

利用平行機器來加工作業的生產型態目前已被廣泛地運用於各種產業中，因此有關平行機器的作業排程問題一直受到相當的重視。所謂平行機器排程是在探討工作中心(work center)中會設置一部以上功能相同的相似機器的排程問題，每部機器皆可執行同一加工作業，排程的目的便是要妥善地利用多部機器，將工件安排到各個機器中，並決定工件的加工順序，以達到所希望績效指標的最佳化。

2.2.1 平行機器的類型

Baker[7]定義平行機器為「生產的工件，僅需經過單一加工途程即能完成，而同時有多部機器可供利用，此即為平行機器的生產型態」。平行機器依其機器間之功能差異程度大致上分為三類[15]：

1. 相同機器（identical machine）：所有的機器均相同，同一工作在各個機器上所需的加工時間皆相等。此類問題是屬於較典型的平行機器排程問題，考量生產現場中的機器都一樣。

表 2.2 多目標排程文獻整理

作者	排程型態	求解方法	考量之目標	權重制訂
Daniels R. L.[17]	單機排程	派工法則	最小化總流程時間 最小化最大延遲 最小化延遲數目	依照各目標成本制訂
Ishibuchi, H.[32]	流程式排程	遺傳演算法	最小化製距 最小化總流程時間 最小化最大延遲	隨機方式決定
Itoh, K. , et al.[33]	零工式排程	TLAS	最小化平均延遲時間 最小化平均流程時間	設定為 0.5
Kim, C. O., et al.[35] Min, H. S., et al.[41]	彈性製造系統	競爭式類神經訓練機器、工件、儲位以及搬運車選擇之派工法則搭配組合	最小化總流程時間 最小化平均延遲時間 最小化平均流程時間 加工機器平均使用率 搬運車平均使用率 最小化製距 系統平均工單數 平均在製品	依照各單一目標設定水準值，再予以評估。
Murata, T., et al.[44]	流程式排程	遺傳演算法	最小化製距 最小化總延遲 最小化總流程時間	隨機方式決定
Neppalli, V. R., et al.[46]	流程式排程	遺傳演算法	最小化製距 最小化總流程時間	研究中採用二種方法 1.將族群分為兩個子族群，分為依照單一目標進行運作， 2.權重設定為 1:1

資料來源：羅友廷[4]

2. 非等效機器（uniform machine）：同一工作在各個機器上的加工時間會因機器加工速度而不同。此類型的問題考量到機器加工速度的差異，例如工廠採購新型的機器後，新型與舊型機器同時為同一製程加工時，由於新舊機器的性能不同，造成加工速度效率上的差異，便衍生出此類的問題。

3. 不相關機器（unrelated machine）：同一工作在各個機器加工時，工作在不同機器上所需的加工時間不相等，其加工時間並無任何關聯，而機器對不同的工作的加工速率也互有差異。此類的問題包括某些工作只限於在某些機器上加工；某些工作在某些機器上的加工速率會優於其他機器，處理此類問題時，通常難度會比較高。

2.2.2 單階平行機器排程

關於單階平行機器排程的文獻整理如表2.3。Guinet[28]先發展一套啟發式解法，然後再利用模擬退火法改善平均延遲時間最小化或加權延遲時間總和之排程問題。Serafini[53]機器的類型上探討工作可以分割，求解以最大加權延遲時間最小化為績效指標之問題。Piersma[49]針對不相關機器以局部搜尋法（塔布搜尋法、模擬退火法、遺傳演算法），求解總時程最小化的排程問題。Suresh和Chaudhuri[60]考量了最大完工時間與最大延遲時間最小化兩個績效指標，他們發展出以塔布搜尋法為基礎的啟發式解法。Cheng和Gen[14]則對不同的工作給予不同的權重值，以區域搜尋法和遺傳演算法求解，使加權的最大延遲完成時間之和最小化。Gürsel等人[30]則考慮機器有整備時間的問題，並針對一工作可以分割至不同機器上加工之情況求解。Tamimi等人[61]針對總加權延遲時間最小化排程問題，以遺傳演算法求解。Srivastava[56]則以最大完工時間最小化為績效指標，而發展出一個以塔布搜尋法為基礎的啟發式解法。Min和Cheng[42]以遺傳演算法，求解總完工時間最小化為績效指標的問題。Serifoðlu和Ulusoy[54]考量

加權的工作延遲及提早完成時間之和最小化，並假設工作都有不同的到達時間、加工時間、設定時間與交期，發展以遺傳演算法為架構的求解方法。林暘桂[3]則發展啟發式演算法與分枝界限法來求解不相關機器的總加權延遲時間最小化問題。

表 2.3 單階平行機器排程文獻整理

作者	年份	平行機器類型	求解方法	績效衡量指標
Guinet, A.[28]	1995	非等效機器	啟發式解法 模擬退火法	平均延遲時間最小化
Serafini, P.[53]	1996	非等效機器		最大加權延遲時間最小化
Piersma, N., and Van Dijk, W.[49]	1996	不相關機器	局部搜尋法 （塔布搜尋法、 模擬退火法、 遺傳演算法）	總時程最小化
Suresh, V. and Chaudhuri, D.[60]	1996	不相關機器	塔布搜尋法為基礎的啟發式解法	最大完工時間最小化 最大延遲時間最小化
Cheng, R. and Gen, M.[13]	1997	相同機器	區域搜尋法 遺傳演算法	加權的最大延遲完成時間之和最小化
Gürsel, A. S., et al.[30]	1997	相同機器	最佳化方法	延遲工作數最小化
Tamimi, S. A. and Rajan, V. N.[61]	1997	非等效機器	遺傳演算法	總加權延遲時間最小化
Azizoglu, M. and Kirca, O.[6]	1998	相同機器	分枝界限演算法	總加權延遲時間最小化
Srivastava, B.[56]	1998	不相關機器	塔布搜尋法為基礎的啟發式解法	最大完工時間最小化
Min, L. and Cheng, W.[42]	1999	相同機器	遺傳演算法	總完工時間最小化
Serifoðlu, F. and Ulusoy, G.[54]	1999	相同機器	遺傳演算法	加權的工作延遲及 提早完成時間之和最小化
林暘桂[3]	2001	不相關機器	啟發式演算法 分枝界限演算法	總加權延遲時間最小化

資料來源：本研究

2.2.3 多階平行機器排程

關於多階段平行機器的排程研究中，大多是探討流程式生產型態，而其求解方法以使用最佳解和啟發式解的技巧為主。多階平行機器排程問題為一典型的NP-complete問題[26][29]。Nowicki[47]用塔布搜尋法求解，使總排程時間最小化；Sridhar和Rajendran[55]則用模擬退火法求取最小的總流程時間。Jacques等人[34]用分枝界限法求工作數不大時之解。Brah等人[10]發展了具平行機器之流程式生產型態的基本數學模型及其延伸來解其他的排程問題。Santos等人[51]建立了多階平行機器流程式排程問題中，最小總排程時間的全域最小界限（Global Lower Bound）。而針對多階平行機器流程式排程問題所發展出的啟發式法則，比較著名的有Nawaz等人[45]的The Nawaz Heuristic與Campbell等人的The Campbell, Dudek, and Smith (CDS) Procedure，及Hundal和Rajgopal[31]與Santos等人[51]的啟發式法則。

2.3 遺傳演算法

遺傳演算法(Genetic algorithm; GA)為 Holland 於 1975 年所提出，其主要的想法是發展一人工系統(artificial system)模擬自然生態運作的方式，藉由演算法的自行淘汰與部分交換來求解最佳化的問題。Goldberg[27]提到遺傳演算法是以自然選擇與遺傳技術為基礎的搜尋過程。Michalewicz[39]更明確指出遺傳演算法是由五個基礎步驟所構成：

1. 以基因型態表示問題的特徵或解答：遺傳演算法對問題的表示方式是將不同的問題特徵（或變數）分別以一個或一組基因表示，其中一基因是二元(binary)數字，而如同一個體是由數個基因所構成，問題的解答也就是由不同的問題特徵所組成。
2. 創造任意數目的初始解答：在遺傳演算法開始運作之前，需先產生一些初始的解答做為初始的狀態，亦即讓電腦創造一些「數位

個體」，形成原始族群(initial population)再進行演算搜尋。至於產生這些個體的方式，分為隨機或是特定的方式，Forgaty[22]則提出產生初始狀態的方法會影響演算法的搜尋績效。

3. 評估功能的建立-適應性函數(fitness function)的設計：若一個體的適應能力愈高，代表此一個體在環境中存活下來的機會較高，則愈有可能繁殖下一代。相對於適應函數的是在解決問題時之目標函數，藉由目標函數來評判解答接近預設目標的程度，若目標函數值愈高則代表此解答愈接近目標，也愈有機會讓遺傳演算法進一步搜尋到更好的解答個體。

4. 使用基因運算子(genetic operator)產生子代：最常見的基因運算子有以下三個。

(i)複製(reproduction)：此運算子的功能決定哪些個體可以存活至下一代，而根據「適者生存」的原則，適應函數值高的個體應具有較高的機率被選中複製而存留至下一族群。

(ii)交配(crossover)：此運算子的功能是透過交換個體間的基因，以重組個體的基因組合，來擴展搜尋空間。Murata[43]以 10 種不同的交配方式對於流程式排程問題進行電腦模擬測試，以決定不同交換方式的績效優劣。

(iii)突變(mutation)：此運算子的功能是藉由隨機改變個體內的基因，引入新的個體型式，增加新的搜尋空間。突變的發生是隨機的，以使得在求解的過程中能夠搜尋新的領域，避免掉入局部最佳解(local optimal)。Murata[43]以 4 種不同的突變方式來進行試驗。

5. 參數的設定：在遺傳演算法的運算過程中，有許多的參數必須事先設定，如交配機率、突變機率、個體數及族群數等。參數的設定會影響搜尋的績效，Forgaty[22]曾對不同的突變機率與遺傳演算的績效進行評估，在特定的原始族群中，不同的突變機率可以改善搜尋績效。一般遺傳演算法運作的流程大致如圖 2.1。

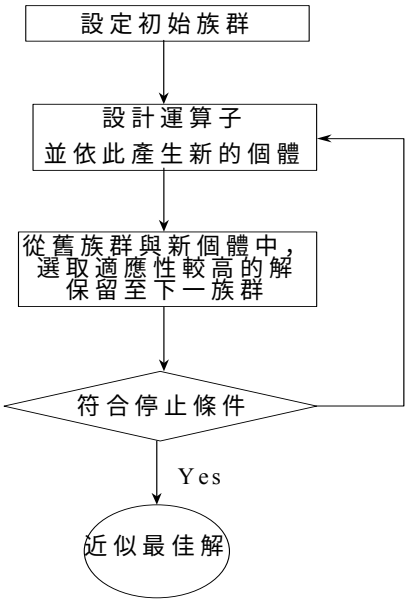

圖 2.1 一般遺傳演算法運作流程

2.3.1 常見 Job shop 排程問題的編碼表示法

在 Cheng[13]的研究報告中，曾調查使用遺傳演算法求解 JSP 時，常見的表示方法。說明如下：

1.Operation-based representation

此類型的編碼方式是將一個排程編碼成作業的順序，而染色體中的每一個基因代表一作業。亦即以一個 m 部機器與 n 個訂單的 Job shop 排程問題(Job-shop Scheduling Problem；JSP)而言，它的每一染色體包含有 m×n 個基因。但是，由於訂單本身的作業有一定的加工先後順序限制，經過交配後所產生的基因順序不能保證能獲得一可行的排程，所以這種編碼方式必須加入修正調整，研究中的修正方式是將同一訂單中的作業在染色體中皆以同一個代號表示。

在這個表示法中，如果問題大小是 n 個訂單與 m 部機器，染色體會有 n ×m 個基因，而染色體中每個訂單只會出現 m 次，每個基因是表示工作內的相對步驟，如此一來使得每一染色體所產生的排列皆是可行解。以一 3 ×3 的問題為例，如果染色體中的基因排列是[2 1 1 1 2 3 2 3 3]，經過解碼轉換後為[O_{21} O_{11} O_{12} O_{13} O_{22} O_{31} O_{23} O_{32}

O_{33}] ，O_{ij} 表示第 i 張訂單的第 j 個作業，最後在排程時依照作業加工順序來進行排程規劃。

2.Job-based representation

此類型只有將訂單做編碼，得到一個訂單的優先順序，排程時是先安排訂單優先次序為 1 的訂單之所有作業，之後，依訂單的優先順序完成整個排程。此法的原則是現決定訂單優先順序，再按其作業程序來安排所有作業。

但是，這種方式與 shifting bottleneck heuristic 有類似的假設：當每一機器的排程是最佳化的時候整個排程即是最佳解，所以這種編碼方式並不能保證可以包含最佳解於其解答空間之內。

3. Preference list-based representation

以一個 m 部機器與 n 個訂單的 JSP 而言，此法的表達方式：每一染色體由 m 個子染色體（subchromosome）所構成，每一子染色體是一個長度為 n 的符號字串，每一個符號代表一個機器的相關作業。子染色體並不是該機器上作業的加工順序，而是該機器的喜好表列（preference list)。整個排程是先找出每部機器的第一喜好作業，再依據每一訂單作業順序限制，判斷機器的第一喜好作業是否可以排入排程之中，若不能則保留。否則找出機器的下一喜好作業，持續至完成所有作業。這種方法只能排出非延遲(non-delay)的排程，可能無法將最佳解含入其中。

4. Job pair relation-based representation

此法是利用一個二元矩陣來，該矩陣是表示二個訂單在機器上的先後關係。其二元的關係定義如下

X_{ijm}＝ 1；如果在機器 m 上，訂單 i 處理時間比訂單 j 早。

0；其它。

因此一個 3 ×3 的問題染色體如果是[0 1 0 1 0 1 1 1 0]其二元矩

陣如下：

訂單(1, 2)在機器(m_1 , m_2 , m_3)：x_{121} x_{122} x_{123} = 0 1 0

訂單(1, 3)在機器(m_1 , m_2 , m_3)：x_{131} x_{132} x_{133} = 1 0 1

訂單(2, 3)在機器(m_1 , m_2 , m_3)：x_{231} x_{232} x_{233} = 1 1 0

然後再依照這個二元矩陣完成整個排程。這個方法最大的缺點是隨著問題變大，會產生愈多不合理的染色體，必須加入其它的調整法則來修正。

5. Priority rule-based representation

這種方法中每一染色體代表派工法則的順序，基因演算法是用來尋求出一個較好的派工法則之順序。以一個 m 部機器與 n 個訂單的 JSP 為例，一個染色體是一個 n×m 的字串（p_1 , p_2 ,…, $p_{n\,m}$），p_i 表示第 i 個循環所使用的派工法則，整個步驟如下：首先列出每個訂單的第一個作業做為可選擇作業，選取加工時間最少的作業，若加工此作業的機器還需要加工這循環中的其它作業，則使用該循環的派工法則決定何者應被選取，進入下一循環。下一循環開始時須將上一循環中加工被選取的訂單之下一作業納入本循環的可選擇作業。每一循環選取一個作業，直到完成整個排程。使用 Priority rule-based 編碼方式所得的排程解，其缺點是品質很不穩定。

6. Disjunctive graph-based representation

這個方法也可以視為是 Job pair relation-based representation 的一種。排程的問題可以利用分支圖（如圖 2.2）來表達，G＝（N, A, E）：N 代表節點來表示所有的作業，A 代表用來連結同一訂單相連的作業（實線部分），E 用來連結同一機器上的作業（虛線部分）。圖中箭頭方向代表作業之間的優先次序，N 與 A 都是確定的，而為了確保機器上的作業不會有加工順序衝突的現象發生，每個機器的作業節點不可以行成一循環(acyclic)，用 e_{ij} 來表示 E 中的每一個箭頭。

e_{ij}＝1，當箭頭的方向是由節點 i 至節點 j，亦即先加工作業 i 再加工作業 j。

e_{ij}＝0，當箭頭的方向是由節點 j 至節點 i，亦即先加工作業 j 再加工作業 i。

利用分支圖來做編碼時，在 n×m 的 JSP 中其染色體為一包含有 n×m 個基因 e_{ij}。以圖 2.2 為例，其基因如下:

[e_{15} e_{19} e_{59} e_{24} e_{28} e_{48} e_{36} e_{37} e_{67}]

=[0 0 1 1 0 0 0 1 1]

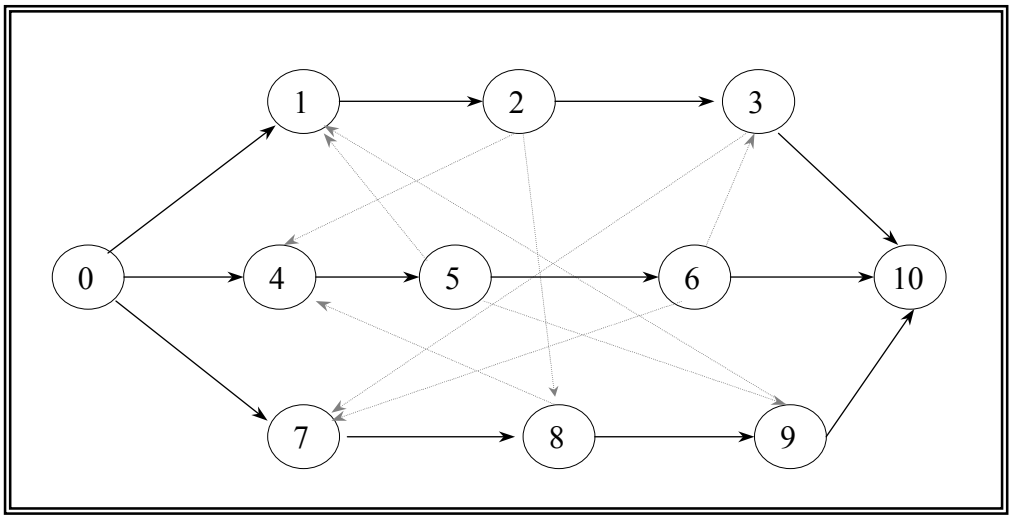

圖 2.2 分支圖

7. Random key representation

在 Random key representation 中，所有的解答被編碼為一串隨機數（random key），用這些數字作為排序的依據。如果問題是 n 個訂單和 m 個機器，每個基因（隨機數）被視為兩個部分，一個是整數部分，其集合為{1, 2, …, m}（代表在哪一個機器執行，因此數字 1 至 m 的出現次數一定是 n 個），另一部份是介於 0~1 的小數，在解碼的時候要將同機器數字來做比較，後面小數部分的大小代表作業的優先次序。假設在一 3 ×3 的問題中有一染色體如下：

[1.34 1.09 1.88 2.66 2.91 2.01 3.23 3.21 3.44]

在機器 1 上的小數部分按照小到大排列順序為[2, 1, 3]，而在機器 2 上的順序為[3, 1, 2]，機器 3 上的順序為[2, 1, 3]，因此將這個染色體轉換成實際機器順序下：

[O_{21} O_{11} O_{31} O_{32} O_{12} O_{22} O_{23} O_{13} O_{33}]

使用這種方式來轉換的排程很容易違反訂單加工途程的限制，必須搭配一些特別的解碼方式。另外，在產生初始族群、交配、突變時所產生的染色體整數部分必須符合數目的限制。

上述之表達法可以分為二個類型：直接式與間接式。所謂的直接式是將一個排程解編碼成一染色體，在透過基因演算法的運算，得到一較佳的排程。而間接式的染色體則是以優先順序法則做為編碼依據，如 Priority rule-based representation 中，其是要透過基因演算法找出較佳的派工法則之順序。

2.3.2 常用排序問題之運算子

使用遺傳演算法求解時，依據問題特性的不同，必須設計適當運算子，讓整個演算過程能順利的進行。在解決排序問題時，一個染色體是由許多“不重覆”的數字基因所構成的字串。在這個限制之下，如果以傳統簡易遺傳演算法中所提的運算子來搜尋，很有可能會產生一個個體中有兩個相同的數字基因。為了避免發生重覆的情況，必須要設計新的運算子，來產生合理的子代。在[21]提及關於遺傳演算法中交配運算子可以分為以下幾種類型：

1.單點交配(one-point crossover)：根據育種選擇策略選擇的二個母體，隨機產生二個切點進行交配。

父代 1：[7 3 | 7 6 1 3] ➔ 子代 1：[7 3 | 4 5 2 2]

父代 2：[1 7 | 4 5 2 2] ➔ 子代 2：[1 7 | 7 6 1 3]

子代 1 中第一部份的基因是繼承父代 1 基因而來，第二部分繼承父代 2，子代 2 則是繼承父代 1 第二部份的基因與父代 2 第一部份的

基因。

2.雙點交配(two-point crossover)：隨機產生二個切點，交換父代彼此的基因。

父代 1：[7　3 | 7　6 | 1　3] ➔ 子代 1：[7　3 | 4　5 | 1　3]

父代 2：[1　7 | 4　5 | 2　2] ➔ 子代 2：[1　7 | 7　6 | 2　2]

3. N 點交配(N-point crossover)：隨機產生 3 至 n 個切點數，決定父代交換奇數或是偶數部分的基因。以下為例：產生 3 個切點（1、2、4）交換奇數部分的基因，產生二子代。

父代 1：[7 | 3 | 7　6 | 1　3] ➔ 子代 1：[7 | 7 | 7　6 | 2　2]

父代 2：[1 | 7 | 4　5 | 2　2] ➔ 子代 2：[1 | 3 | 4　5 | 1　3]

4.均於交配(uniform crossover)：每一基因的位置皆產生一個 0 ~ 1 的亂數，假如亂數> 0.5，則該位置的基因不交換。舉列而言，在下列中的中產生的亂數依序為 0.2，0.7，0.9，0.4，0.6，0.1，產生的子代如下：

父代 1：[7　3　7　6　1　3] ➔ 子代 1：[7　7　4　6　2　3]

父代 2：[1　7　4　5　2　2] ➔ 子代 2：[1　3　7　5　1　2]

在[1]的研究中，將一些常用在排序問題的運算子，整理如下。

1.PMX (Partially Matched Crossover)

(1)隨機產生兩個切點。

個體 A：[　9　8　4 | 5　7　6 | 1　3　2　]

個體 B：[　8　7　1 | 2　3　6 | 9　5　4　]

(2)將個體 A 與個體 B 在兩切點中的基因互調。

個體 A：[　9　8　4 | 2　3　6 | 1　3　2　]

個體 B：[　8　7　1 | 5　7　6 | 9　5　4　]

(3)將個體 A 位於切點之外重覆的基因與個體 B 位於切點之外重覆的基因互調。

個體 A：[9 8 4 | 2 3 6 | 1 7 5]

個體 B：[8 3 1 | 5 7 6 | 9 2 4]

2. LOX(Linear Order Crossover)

(1)隨機產生兩個切點。

個體 A：[9 8 4 | 5 7 6 | 1 3 2]

個體 B：[8 7 1 | 2 3 6 | 9 5 4]

(2)將個體 A 切點中的所有位元，在個體 B 中以*代替；B 者亦然。

個體 A：[9 8 4 | 5 7 * | 1 * *]

個體 B：[8 * 1 | 2 3 * | 9 * 4]

(3)將*往中間移動，使得兩個切點中的位元皆為*。

個體 A：[9 8 4 | * * * | 5 7 1]

個體 B：[8 1 2 | * * * | 3 9 4]

(4)將原本個體 A、B 切點中的位元互調。

個體 A：[9 8 4 | 2 3 6 | 5 7 1]

個體 B：[8 1 2 | 5 6 7 | 3 9 4]

3. SX(Simple Crossover)

(1)隨機產生 1 個切點。

個體 A：[9 8 4 | 5 7 6 1 3 2]

個體 B：[8 7 1 | 2 3 6 9 5 4]

(2)保留切點左邊的基因，切點右邊的位元以其在另一個體的順序填入。

個體 A：[9 8 4 | 7 1 2 3 6 5]

個體 B：[8 7 1 | 9 4 5 6 3 2]

4. RX(Random Crossover)

(1)隨機產生兩個切點。

個體 A：[9 8 4 | 5 7 6 | 1 3 2]

個體 B：[8 7 1 | 2 3 6 | 9 5 4]

(2)切點兩旁的基因保留不變，兩切點中的基因以隨機產生。

個體 A：[9 8 4 | 6 5 7 | 1 3 2]

個體 B：[8 7 1 | 3 6 2 | 9 5 4]

5. CX(Cycle Crossover)

(1)在個體 A 中任選一個基因，假設選到 9，其相對在個體 B 為 1，將其標示起來。

個體 A：[9 8 2 1 7 4 5 6 3]

個體 B：[1 2 3 4 5 6 7 8 9]

(2)個體 A 中基因為 1 的位元在個體 B 中是４，將其標示起來。

個體 A：[9 8 2 1 7 4 5 6 3]

個體 B：[1 2 3 4 5 6 7 8 9]

(3)個體 A 中基因為 4 的位元在個體 B 中是 6，將其標示起來。

個體 A：[9 8 2 1 7 4 5 6 3]

個體 B：[1 2 3 4 5 6 7 8 9]

(4)個體 A 中基因為 6 的位元在個體 B 中是 8，將其標示起來。

個體 A：[9 8 2 1 7 4 5 6 3]

個體 B：[1 2 3 4 5 6 7 8 9]

(5)重覆以上的步驟，直到最後的標示回到 9。

個體 A：[9 8 2 1 7 4 5 6 3]

個體 B：[1 2 3 4 5 6 7 8 9]

(6)將個體 A、B 中沒有被標示的基因互換。

個體 A：[9 8 2 1 5 4 5 6 3]

個體 B：[1 2 3 4 7 6 7 8 9]

6. OM(Order-based Mutation)：

任選兩個基因，將其互調即可。

[9 8 4 5 7 6 1 3 2] ➔ [9 8 4 1 7 6 5 3 2]

7. PM(Position based Mutation)

任選兩個基因，假設為 5、1，將 7、6、1 往前移，再將 5 填入原本基因 7 的位置即可。

[9 8 4 5 7 6 1 3 2] ➔ [9 8 4 7 6 1 5 3 2]

2.3.3 排程中遺傳演算法之應用

Pinedo[50]指出，遺傳演算法可以被應用在結構未知的問題上，Wellman[62]則提到遺傳演算法已經成功地應用在最佳化求解的問題，如排程、運輸問題、旅行者問題等。Kim[36]以 GA 為基礎發展一啟發式搜尋技術解決 Job shop 的排程問題。Petty[48]提出分散式的遺傳演算法，將一族群分割成較小的族群，再分別進行 GA 運作，以提高求解效率。Murta[43]以製距最小為目標，評估 GA 分別結合局部搜尋法、模擬退火法在 Flow Shop 排程問題上的表現。而應用 GA 求解多目標的問題[32][44][46]，在育種選擇時通常會採用精華保留策略(elite preserve strategy)，所謂保留精華的策略乃指在演算的過程中，選擇每一代族群時，另外再保留在單一目標中有最佳評估值（目前為止）的個體。

基本上遺傳演算法與其他搜尋方式比較起來，能夠以較短的時間搜尋到不錯的解答，對問題變數的多寡影響相對的比較小，有其他搜尋演算法所缺乏的優點。但是在實際應用時，遺傳演算法在不同的問題上仍有一些缺點，尚待克服。今將遺傳演算法的優缺點歸納如下表：

表 2.4 遺傳演算法之優缺點歸納表

	遺傳演算法
優點	**1. 多點同步搜尋** 遺傳演算法同時考慮搜尋空間上多個點而不是單一個點，因此可以較快地獲得整體最佳解(global optimum)，同時也可以避免陷入區域最佳解(local optimum)的機會，此項特性乃是遺傳演算法的最大優點。 **2. 使用適應函數** 遺傳演算法的運作過程只使用適應函數的資訊而不需要其它輔助的資訊(例如:可微分)，因此可以使用各種型態的適應函數(例如:多目標、非線性或以知識為基礎)，並可節省計算資源避免繁複的數學運算。 **3. 機率式的搜尋** 遺傳演算法是使用機率規則(stochastic)的方式去引導搜尋方向，而不是用明確(deterministic)的規則，因此較能符合各種不同類型的最佳化問題。 **4. 基因編碼** 遺傳演算法是以參數集合之編碼進行運算而不是參數本身，因此可以跳脫搜尋空間上的限制。
缺點	**1. 運算子的設計問題** 根據不同的問題，需要設計不同功能的運算子，以提高搜尋效率。若單純使用簡易遺傳演算法的運算子，搜尋速度會受到影響。 **2. 重複搜尋的問題**。 因為遺傳演算法並沒有記憶功能，且其運作過程只與適應函數相關，因此往往在搜尋的過程中，重複搜尋到相同的點，增加系統搜尋的時間。

2.4 螞蟻系統

螞蟻系統(Ant system)為 Dorigo 等人於 1991 年所提出。誠如其名，螞蟻系統乃是根據自然界中螞蟻覓食的行為模式所發展出來的演

算法，其最大的特色為人造螞蟻(artificial ant)根據路徑上的費洛蒙(pheromone)多寡來搜尋求解組合最佳化的問題。至今有許多研究利用螞蟻系統求解 NP-Hard 的問題，整理如表 2.5。

表 2.5 螞蟻系統文獻整理

求解問題	作者	年
旅行推銷員問題 (Traveling Salesman Problem；TSP)	Dorigo et al.[20]	1996
	Dorigo et al.[19]	1997
	Stützle and Hoss[58]	1997
	Freisleben and Merz [23]	1999
二次分配問題 (Quadratic Assignment Problem；QAP)	Gambardella and Dorigo[24]	1997
	Maniezzo[37]	1999
	Stützle and Dorigo[57]	1999
車輛途程問題 (Vehicle Routing Problem；VRP)	Bullnheimer et al.[12]	1999
	Gambardella et al.[25]	1999
網路途程問題 (Network Routing Problem)	Di Caro and Dorigo[18]	1998
排程問題 (Scheduling Problem)	Stützle[59]	1997
	Michel and Middendorf [40]	1998
	Bauer et al.[9]	1999

資料來源：本研究

2.4.1 螞蟻系統介紹

螞蟻演算法(Ant algorithm)之發展起源於自然界中螞蟻之移動行為。當螞蟻離開蟻窩尋找食物時，會分泌一種稱為費洛蒙(pheromone)之荷爾蒙，螞蟻行經一路徑之機會與該路徑上所殘留之費洛蒙成正比。越多螞蟻走過的路徑則遺留之費洛蒙越多，而遺留越多費洛蒙又會吸引越多螞蟻行走該路徑；相對的，其餘路徑的費洛蒙會漸漸蒸發。因此，當螞蟻面臨兩條路以上之抉擇時，其行走某一路線之機率與其遺留費洛蒙的數量有關，而越短之路線其螞蟻通過時間短，導致最短路線上遺留之費洛蒙量越多，進而誘使更多螞蟻行經最短路徑，最後所有螞蟻將趨於最短路徑行走，如圖 2.3 所示。螞蟻演算法便是

模擬真實螞蟻行為，學習其移動搜尋的方式，以人造螞蟻(artificial ant)進行最佳決策之搜尋工作，逐次搜尋出較短之路徑，最後人造螞蟻將沿最短路徑，而求得最佳解。

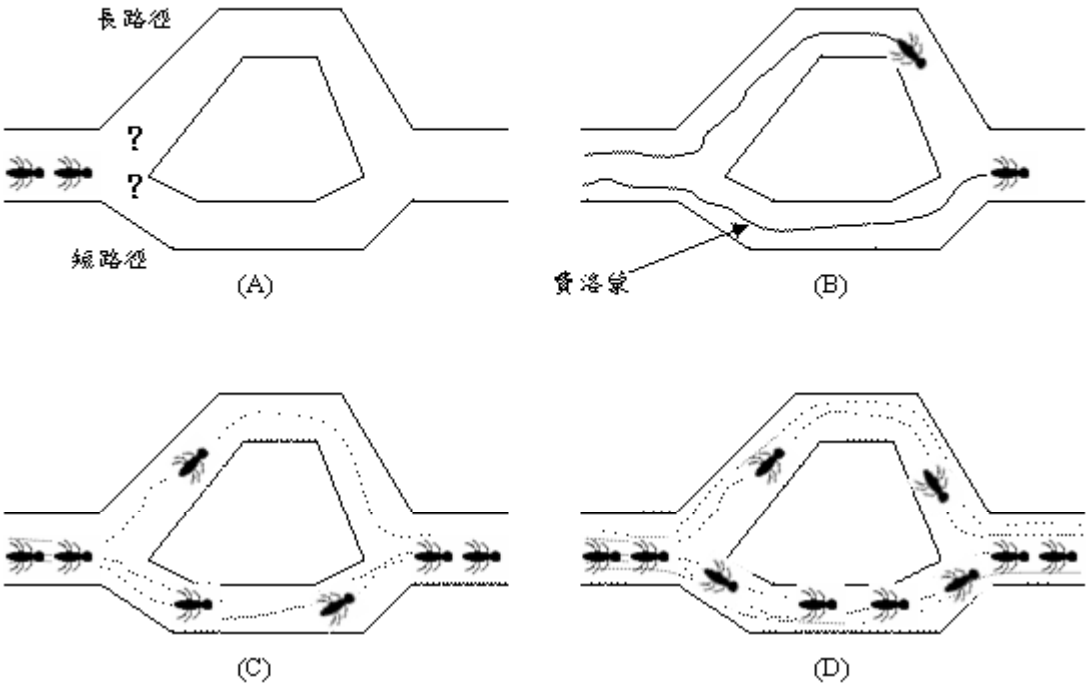

圖 2.3 螞蟻之移動行為模式

Dorigo 等人[20]指出螞蟻演算法的步驟如下：

1. 初始設定(Initialize)：在每一路徑上，設定相同的初始費洛蒙數量，並將人造螞蟻放至搜尋起始點上。
2. 設定記憶陣列(tabu list)：將每一隻人造螞蟻的搜尋起始點加入其所攜帶的記憶陣列中。
3. 選擇路徑(edge)與結點(town)：每隻人造螞蟻依據式 2.1 選擇要走的結點與經過的路徑，並移動至該結點上，將此結點加入記憶陣

列中，重複此步驟直到填滿記憶陣列。

$$p_{ij}^{k}(t)=\begin{cases}\dfrac{[\tau_{ij}(t)]^{\alpha}\cdot[\eta_{ij}]^{\beta}}{\sum_{k\in allowed_k}[\tau_{ik}(t)]^{\alpha}\cdot[\eta_{ik}]^{\beta}} & if \quad j\in allowed_k \\ 0 & otherwise\end{cases} \qquad \text{式（2.1）}$$

其中，

$\tau_{ij}(t)$：在時間點 t 時路徑(i , j)上所殘留的費洛蒙數量

η_{ij}：路徑(i , j)的能見度(visibility)，等於路徑(i , j)長度的倒數

α,β：控制費洛蒙與能見度間相對重要性的參數

$p_{ij}^{k}(t)$：第 k 隻螞蟻從結點 i 要到結點 j 的轉移機率

4. 計算增加的費洛蒙強度(intensity of trail)：依據記憶陣列之資訊，更新到目前為止所找到的最短總路徑；並計算出每一螞蟻在其搜尋的路徑上貢獻的費洛蒙大小，如式 2.2 與式 2.3。

$$\Delta\tau_{ij}^{k}=\begin{cases}\dfrac{Q}{L_k} & if \quad (i,j)\in tour\ described\ by\ tabu_k \\ 0 & otherwise\end{cases} \qquad \text{式（2.2）}$$

$$\Delta\tau_{ij}=\sum_{k=1}^{m}\Delta\tau_{ij}^{k} \qquad \text{式（2.3）}$$

其中，

Q：代表費洛蒙數量的常數

L_k：第 k 隻螞蟻所走的總路徑長度

$\Delta\tau_{ij}^{k}$：第 k 隻螞蟻殘留在路徑(i , j)上的每單位距離費洛蒙數量

5. 更新費洛蒙強度：依據式 2.4 更新所有路徑的費洛蒙強度，並使

$\Delta\tau_{ij} = 0$。

$$\tau_{ij}(t+n) = \rho \cdot \tau_{ij}(t) + \Delta\tau_{ij} \qquad \text{式（2.4）}$$

其中，

ρ：蒸發係數

6. 停止條件：如果尚未達到終止搜尋的條件，便清空所有的記憶陣列，回步驟 2 重覆進行以上步驟，否則得到此次搜尋的最短總路徑。

Dorigo 等人[20]介紹螞蟻演算法之特色為：

- 確實的回饋：能夠快速的發現新的起始解。
- 分散的計算：就是所謂的多點搜尋，能夠避免過早的收斂。
- 使用積極的貪心法則：能在發展起始解的時候，較早發現可接受解。

2.4.2 實行螞蟻演算法於 Job shop 排程問題

Colorni 等人[16]與 Dorigo 等人[20]均曾在其研究中敘述如何以螞蟻演算法來求解 Job shop 排程問題，而 Zwaan 和 Marques[63]更實際地以螞蟻演算法求解不同機器與工作數的 Job shop 排程問題，並探討相關參數的最佳化設定。

使用螞蟻演算法來求解 Job shop 排程問題，首先必須將問題以圖形清楚地表達，如圖 2.4，其為一 $2/3/G/C_{max}$ 的 Job shop 範例。圖上的結點均代表一作業，屬於同一訂單的結點以單向水平路徑相連，限制了加工途程的順序，剩餘的路徑則是雙向的，而原點 0 與訂單的第一個作業以單向路徑相連，其設計是為了使排程能夠初始化，人造螞蟻一開始便是在原點上。在搜尋過程中，每條路徑（路徑(i , j)）同

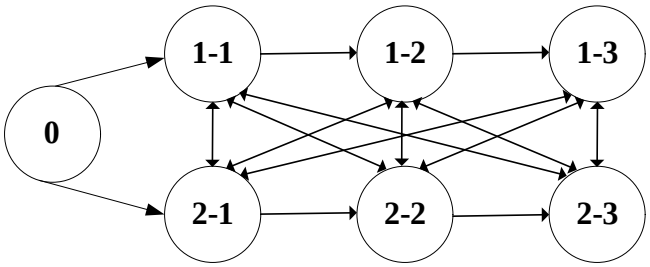

圖 2.4 Job shop 排程問題的圖示

樣有其費洛蒙與能見度，不同的是此處能見度為結點 j 作業的加工處理時間的倒數，而需特別注意的是雙向路徑上路徑(i , j)與路徑(j , i)的能見度並不相等。

螞蟻演算法最後搜尋出的結點順序即代表排程規劃時排入作業的順序，因此另一個重要的問題乃是如何確保螞蟻能夠搜尋出合理解，不違反加工途程。針對此問題，Dorigo[20]讓每隻螞蟻配備三組記憶陣列：*G* 陣列包含尚未搜尋過的結點，*S* 陣列包含在現在的時間點上允許搜尋的結點，而 *Tabu* 陣列則記錄搜尋過的結點。圖 2.4 中初始的陣列如下：

$G_k=\{O_{11}, O_{12}, O_{13}, O_{21}, O_{22}, O_{23}\}$

$S_k=\{O_{11}, O_{21}\}$

$Tabu_k=\{\ \}$

另外，為了防止搜尋過早陷入區域最佳解，會加入一變異運算子（variation），當此運算子被觸發，便取代轉移機率規則，幫助螞蟻在搜尋空間中拓展出不同的求解方向。假如變異運算子被觸發，螞蟻將不考量路徑上費洛蒙多寡，而僅靠能見度指引其搜尋，如式 2.5。最後，當螞蟻演算法完成一個循環的搜尋後，*Tabu* 陣列所記錄的資訊即為一排程解。

$$\begin{cases} p_{ij}=\dfrac{\eta_{ij}}{\sum_{j\in nodes\ allowed}\eta_{ij}} & ,n<v \\ State\ Transition\ Rule & ,n>v \end{cases} \qquad \text{式（2.5）}$$

n : *random number between [0 1]*
v : *percentage of variation [0 1]*

2.4.3 排程中螞蟻演算法之應用

Colorni 等人[16]與 Dorigo 等人[20]在其研究中提及以螞蟻演算法來求解 Job shop 排程問題的概念與方法。而 Zwaan 和 Marques[63]以最大完工時間最小化為績效指標，實際地以螞蟻演算法求解不同機器與工作數的 Job shop 排程問題，並加入變異運算子，探討相關參數的最佳化設定，比較最後的排程結果。Bauer[9]以最小化總延遲時間為績效指標，以螞蟻演算法求解單機排程問題。T'kindt 則以螞蟻演算法為基礎，發展一啟發式搜尋技術解決 2 台機器特例的 Flow shop 排程問題，並與不同的啟發式法則作比較。Stützle[59]以螞蟻演算法求解一般化的 Flow shop 排程問題，與不同的啟發式法則作比較。Blum 和 Sampels 則在求解 Flow shop 排程問題時，提供不同的費洛蒙表達方式。

第三章 多階平行機器零工式多目標排程系統架構設計

在現實的排程環境中，生管排程人員所需考量的因素非常的多，這些因素可分為定性因素與定量因素（如表 3.1）。一般所考慮的定量因素有：交期、機器使用率、製距，…等。現實環境中還存有另外的質性因子，例如：公司策略因素、顧客的歷史交易、或是該訂單所需物料的情形，…等。而目前多數關於排程的研究，僅只考慮定量方面的因素，很少提及定性的因素，且其所發展的排程演算法多是以單一目標作為排程績效衡量的基礎，但一般製造環境的生產排程問題並非單一目標可以滿足，皆需以多目標的觀點來考慮。

表 3.1 定性與定量因素表

定性因素	定量因素
＊ 市場考量 ＊ 該訂單的顧客 ＊ 該訂單顧客的潛在訂單 ＊ 該訂單顧客歷史交易 ＊ 訂單的利潤	＊ 製距時間 ＊ 交期滿足 ＊ 現場使用率/負荷 ＊ 訂單延遲數目 ＊ 延遲時間 ＊ 等待時間

傳統的 Job shop 排程問題研究中，定義每一階作業所需的加工機器種類均只有一台機器。然而現今工廠的生產型態卻不是這樣，為了能夠提高產能及減少瓶頸工作站的負荷，均會在工作中心(work center)內設置功能相同的平行機器，如圖 3.1 所示。訂單在不同工作中心間流動，在各工作中心內進行不同的加工程序(operation)，藉著訂單作業的批量分割，一作業可同時在多部機器處理，而縮短各作業之完成時間，達到交期準確的目標。

本研究試圖將定性與定量兩種因素同時納入排程規劃，並模式化具有平行機器生產型態的 Job shop 類型排程問題。目前在排程問題中，平行機台排程問題是困難度極高的組合最佳化 NP-hard 問題，隨

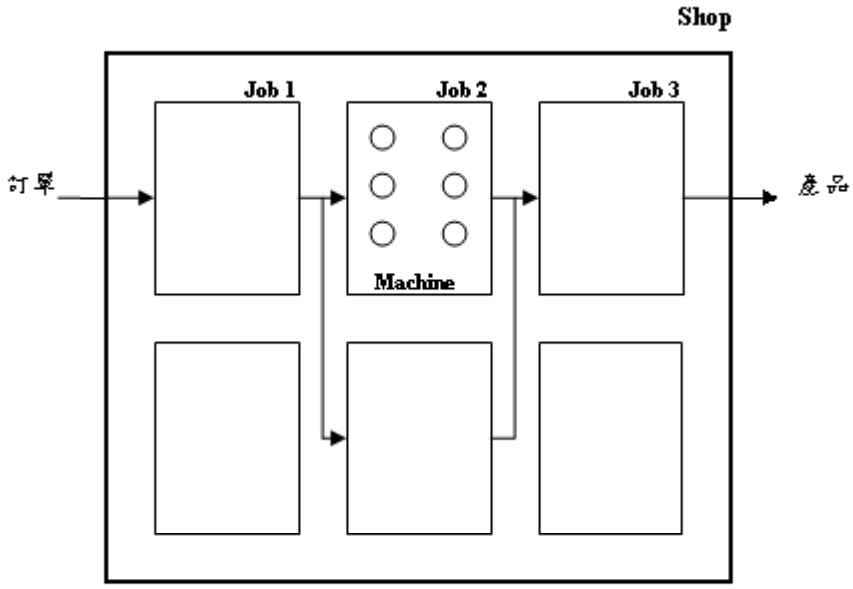

圖 3.1 零工式生產系統示意圖

著問題的規模與複雜度的增加，若欲使用最佳化方法求解，遺傳基因演算法與螞蟻演算法均是發展中優異的最佳化方法。所以研究中將以遺傳演算法與螞蟻演算法分別進行搜尋求解，藉由制定多目標適應函數(multi-objective fitness function)來處理定量方面的因素。關於定性因素方面，採用以訂單為導向的方式來考量，每一訂單依照所需考量的因素來評估，建立一訂單優先順序。再將此訂單優先順序轉為一懲罰函數納入目標函數中，做為遺傳演算法與螞蟻演算法運作時的限制。

本研究提出一多階平行機器多目標排程模式，其系統架構如圖 3.2 所示。

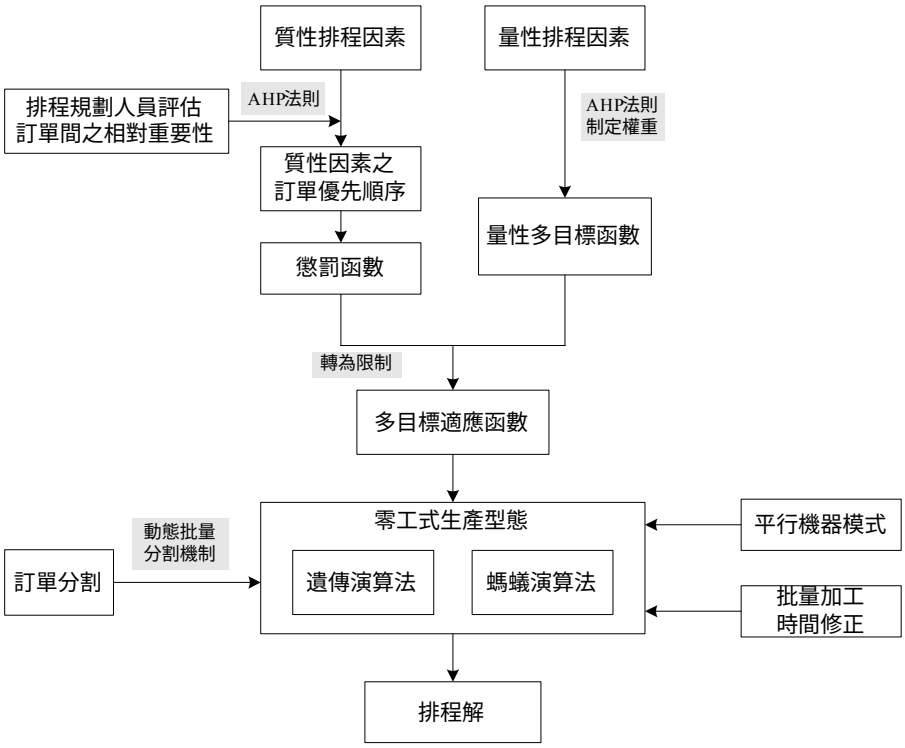

圖 3.2 多階平行機器多目標排程系統架構圖

3.1 多階平行機器排程法則

為了能更貼近現實環境的生產模式，本研究允許訂單有一動態的批量分割機制。在加工每一作業之前，先將訂單的總批量分割成數個子批，使訂單能同時在多部機器加工，藉此能有效運用平行機器，發揮其功用。

3.1.1 排程演算法

本研究考慮批量分割，如圖 3.3。欲加工的訂單按照其途程順序在第一個工作中心加工之前，會有一動態批量分割機制（演算法的設計中會提及）均勻地分割訂單的總批量，而子批數也就等於此訂單加工時在工作中心需佔有的機器數，子批數上限為此工作中心擁有的機器數，待所有子批均加工完畢，批量分割機制再一次分割訂單，做為下一途程加工批量大小與需佔機器數的依據，批量分割的機制在每個工作中心與工作中心之間均需執行一次，直到此訂單加工完畢。

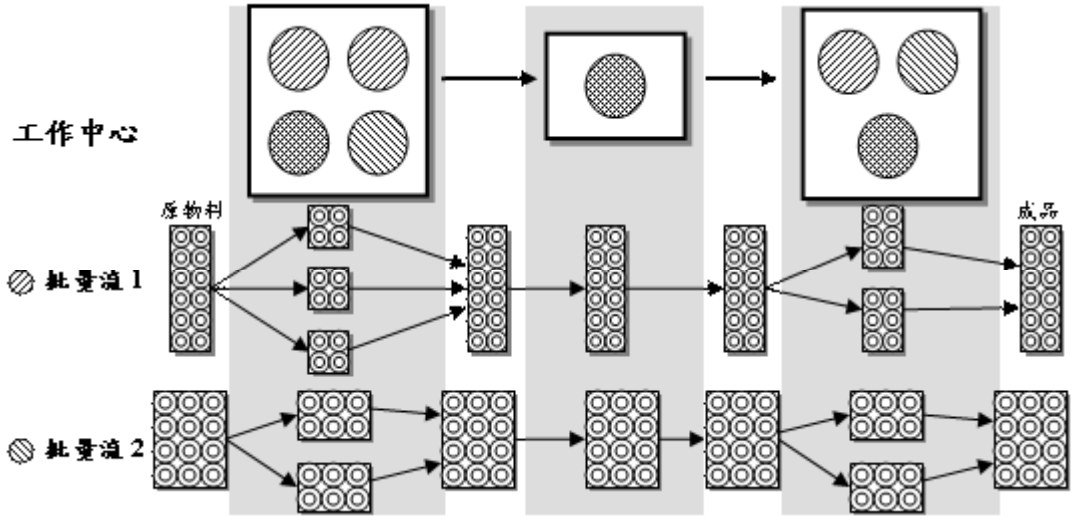

圖 3.3 訂單批量分割的現象

本研究中對於每張訂單而言，只有唯一的途程，每一作業的單位加工時間假設為固定且明確的，最後每項工作的生產途程假設為已知。這些訂單的生產方式是以接單生產（make-to-order）的方式進行，以反映產品在市場上的快速變化並具生產彈性。

排程演算法主要是以將產能集中的概念，以減少機器閒置時間與縮短訂單作業完成時間為目的所發展出來的。每一個作業依據由演算法所搜尋出的優先順序安排於排程計畫內，在完成實際的排程過程中，重覆進行搜尋一個最早空閒區間能夠容納欲排入之作業。傳統多階單機的排程問題，因其為單一作業獨佔機台時間，可以很容易且快速地做出判斷，但在多階平行機器且允許批量分割的環境之下，每個批量均需對所有平行機器進行判斷，相較於傳統情形複雜度提高許多。

本研究排程模式為零工式生產型態，並考慮為數眾多的平行機器問題，基本假設如下：

1. 生產系統不只生產單一產品，而是生產多種不同產品。
2. 有多個工作中心，每個工作中心代表一個加工途程。
3. 當時間開始時，所有訂單皆已準備就緒。
4. 每一工作中心對每一產品有上機批量限制的下限，如果批量不足不予以開機加工。解決方法為逐一減少佔有機器的數目，直至切割出的批量大小達到下限。
5. 訂單總批量大小均超過上機批量限制。
6. 機器設置時間已包含於加工時間中。
7. 每個批量在每個工作中心只加工一次。
8. 每部機器同一時間內最多只能處理一批量。
9. 批量在加工過程中不能被中斷。
10. 在加工機器上，某些時段已有加工作業在其上加工中。
11. 需等待作業的分割批量在前一個工作中心全部加工完成後，才能到下一個工作中心進行加工，運送時間忽略不計。
12. 在具有多部機器的工作中心中，每部機器皆被視為相同機器。

13. 每個批量在每一工作中心中，只需經過其中任一部機器加工處理。

訂單經過前述批量分割機制後，便可得知欲佔有的機器數（假設為 M 台機器）。逐一判斷所有的平行機器，其空閒時區能否容納此批量加工時間，若否，則找尋機器中下一空閒時區，直至可被排入，最後，為了減少機器閒置時間，挑選出空閒時區最早的 M 台平行機器，將批量作業分別排入此 M 台機器排程中，作業排程流程如圖 3.4。排程演算法步驟如下：

步驟一：欲排之作業經過動態批量分割機制，將訂單分割為數個子批。

步驟二：判斷分割出的批量大小是否達到上機批量限制的下限。若無達到，則再執行一次步驟一。若有，則得到加工此作業需要 M 台平行機器的資訊。

步驟三：如果欲排之批量作業是其訂單加工途程中的第一個作業，則執行步驟四，否則執行步驟八。

步驟四：若執行此批量作業之平行加工機器目前無安排其他作業，則直接將此批量作業排入。否則執行步驟五。

步驟五：尋找平行加工機器中的最早空閒時間區段。

步驟六：依據平行加工機器之空閒狀態與此批量作業的開始時間，計算此批量作業的結束時間。

步驟七：判斷批量作業之結束時間與平行加工機器之下一作業之開始時間，兩者在時間上是否有衝突，若有，則繼續尋找機器下一空閒時間區段，執行步驟六。若無衝突產生，則此批量作業完成排程。

步驟八：若執行此批量作業之平行加工機器目前無安排其他作業，則依據途程中的前一作業的結束時間作為該批量作業的開始時間。計算此批量作業的結束時間完成此批量作業之排程。

否則執行步驟九。

步驟九：以途程中的前一作業的結束時間為起點，尋找平行加工機器中的最早空閒時間區段。

步驟十：判斷平行加工機器之空閒時間狀態以及途程前一作業之完成時間，計算此批量作業的開始時間與批量作業的結束時間。

步驟十一：判斷批量作業的結束時間與平行加工機器中下一作業之開始時間是否有衝突。若有，則繼續尋找機器下一空閒時間區段，執行步驟十。若無衝突產生，則執行步驟十二。

步驟十二：判斷所有的平行加工機器是否均完成批量作業的排程。若尚有加工機器未完成批量作業排程，則尋找下一未完成排程的平行加工機器，執行步驟三。若均完成，執行步驟十三。

步驟十三：選定批量作業開始時間最早的 M 台平行加工機器，並將 M 台平行加工機器中最早的批量作業開始時間作為該作業開始時間，最晚的批量作業結束時間作為該作業結束時間，則此作業完成排程。

雖然以此種排程演算法所得的排程結果能有效地減少機器閒置時間而使得產能集中，但是相對地會造成在多部機器加工的同一訂單批量加工時間誤差過大。有鑑於此，下一小節中將局部修正批量作業的加工時間。

3.1.2 作業時間修正模式

本研究中將生產批量採取批量分割，目的是讓作業可在一製程中分散至多部平行機器上同時加工，使得加工作業能夠重疊，縮短總生產時間。

但是上述排程演算法有可能會造成作業分割批量的加工區間相差甚遠，使得在製品(work-in-process)囤積於工作中心的時間過長，徒

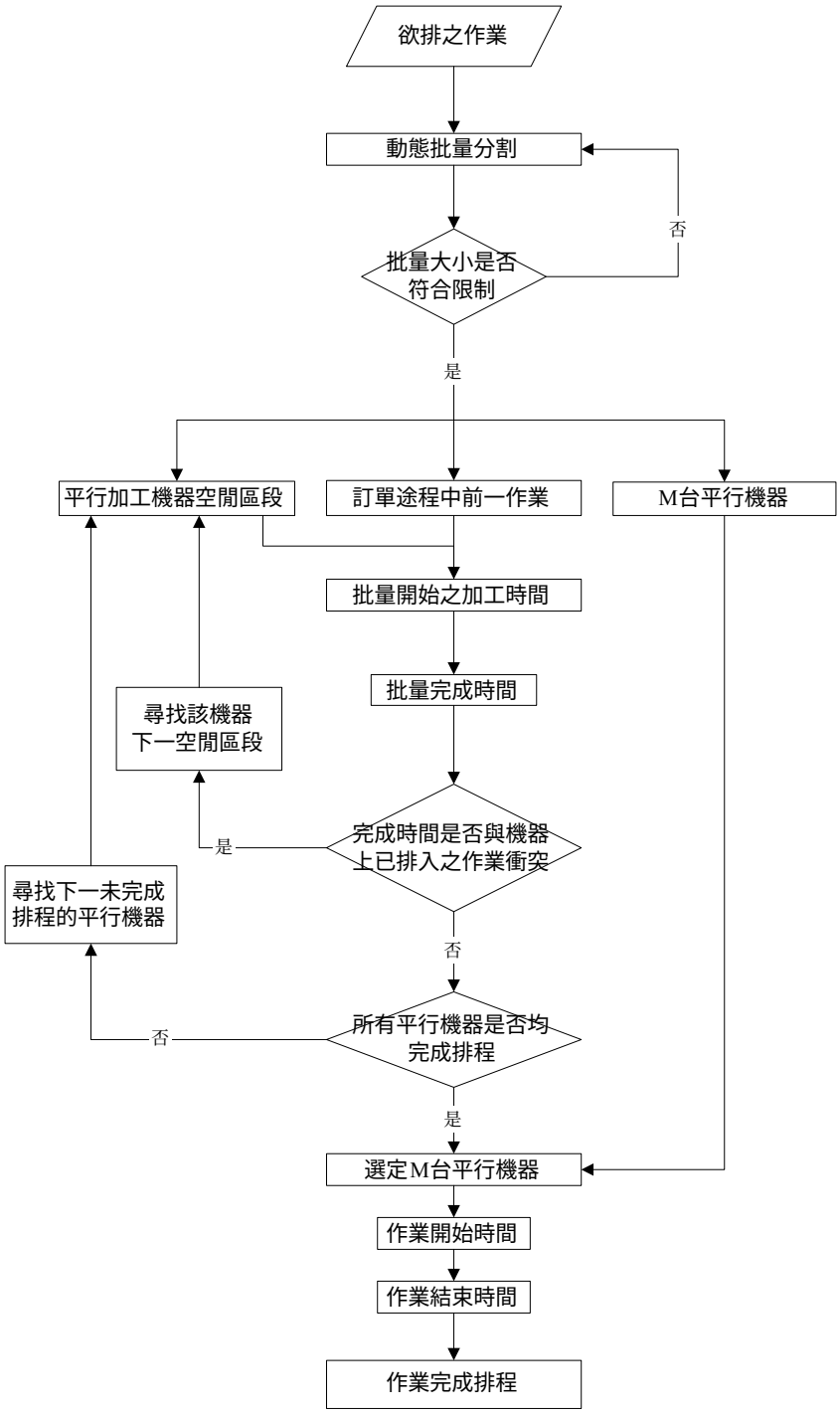

圖 3.4 作業的排程流程圖

增許多不必要的生產成本，如圖 3.5 所示。加工某作業所需的工作中心一共有三台平行機器 M1、M2 與 M3，而作業欲分割成三個批量分別至機器上加工，虛線方塊代表先前在平行機器上已排定之作業加工

時間，實線方塊代表此作業所排定的加工時間。由於 M1 上已排定作業較多的關係，使得子批 1 的加工區間較子批 2 與子批 3 晚了許多，因此作業的子批 2 與子批 3 加工完畢後，需經過長時間的等待，子批 1 才能結束加工，生產批量才算全部加工完畢。

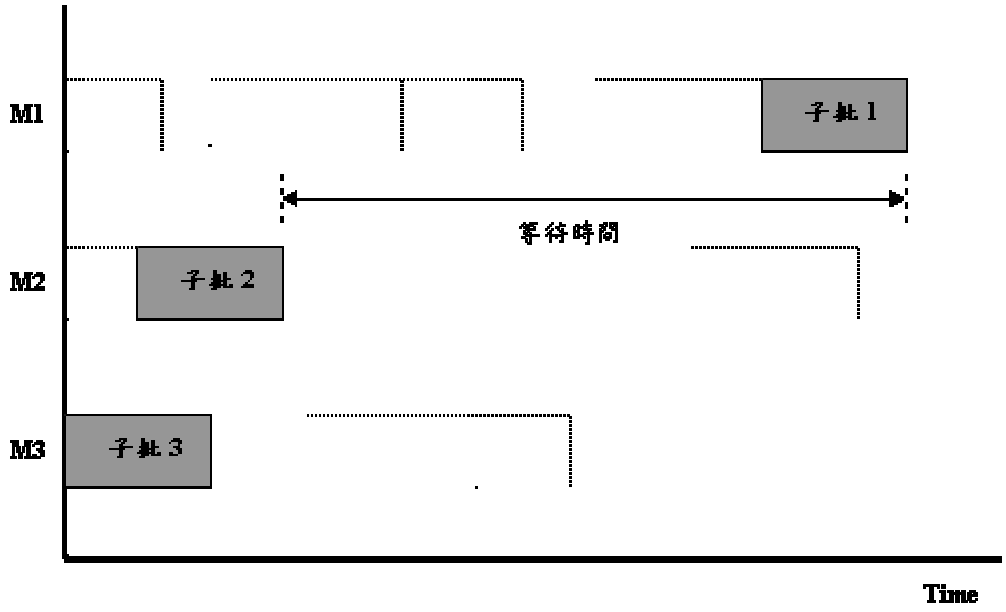

圖 3.5 原始排程甘特圖

在此例中，子批 2 與子批 3 在等待時間中均為在製品，等待時間明顯過長，使在製品管理困難，並造成同一種產品的生產不連續，間接地讓機器的操作與換模等相關步驟複雜化，這對排程來說，並不是一個最好的結果。

因此本研究在每一作業排程時間排定後，增設作業時間修正模式。以子批中最晚的加工結束時間為限制，調整各子批的加工時間，使其盡量靠近最晚的加工結束時間，讓作業分割批量的加工區段能彼此重疊，達到生產連續，而在製品的等待時間也能減到最低。作業修正流程如圖 3.6。修正模式的步驟如下：

步驟一：如果欲修正之子批是作業中加工結束時間最晚的子批，則子批的作業時間無需調整，執行步驟八。否則執行步驟二。

步驟二：往後尋找平行加工機器中的下一空閒時間區段，將子批作業時間往後遞移。

步驟三：依據平行加工機器之空閒狀態，計算修正後此子批作業的開始與結束時間。

步驟四：如果子批作業之結束時間沒有超過子批中最晚的加工結束時間，執行步驟七。否則執行步驟五。

步驟五：往前尋找平行加工機器中的上一空閒時間區段，將子批作業時間往前遞移。

步驟六：依據平行加工機器之空閒狀態，計算修正後此子批作業的開始與結束時間。執行步驟八

步驟七：如果子批作業不是現階段機器上已排入作業中的最後一個作業，則繼續尋找機器下一空閒時間區段，執行步驟二。否則執行步驟八。

步驟八：判斷所有子批是否均經由修正模式調整完成，若尚有子批未調整，則尋找下一未調整子批，執行步驟一。否則所有子批的加工時間均調整完畢，則此作業完成修正。

上例中作業時間修正結果如圖 3.7 所示。以子批 1 的加工結束時間為限制，將子批 2 與子批 3 的作業加工時間往後遞移，使各子批間的作業時間重疊，工作中心能連續生產同一類產品與減少在製品囤積的時間。換句話說，經由修正模式，子批 2 與子批 3 可不必太早在機器上進行加工；相較於原來無修正的情形來說，在機器上空出的較早閒置產能尚可被其餘未排入加工之作業利用，使其能提早開始加工，達成交期準確的目標。

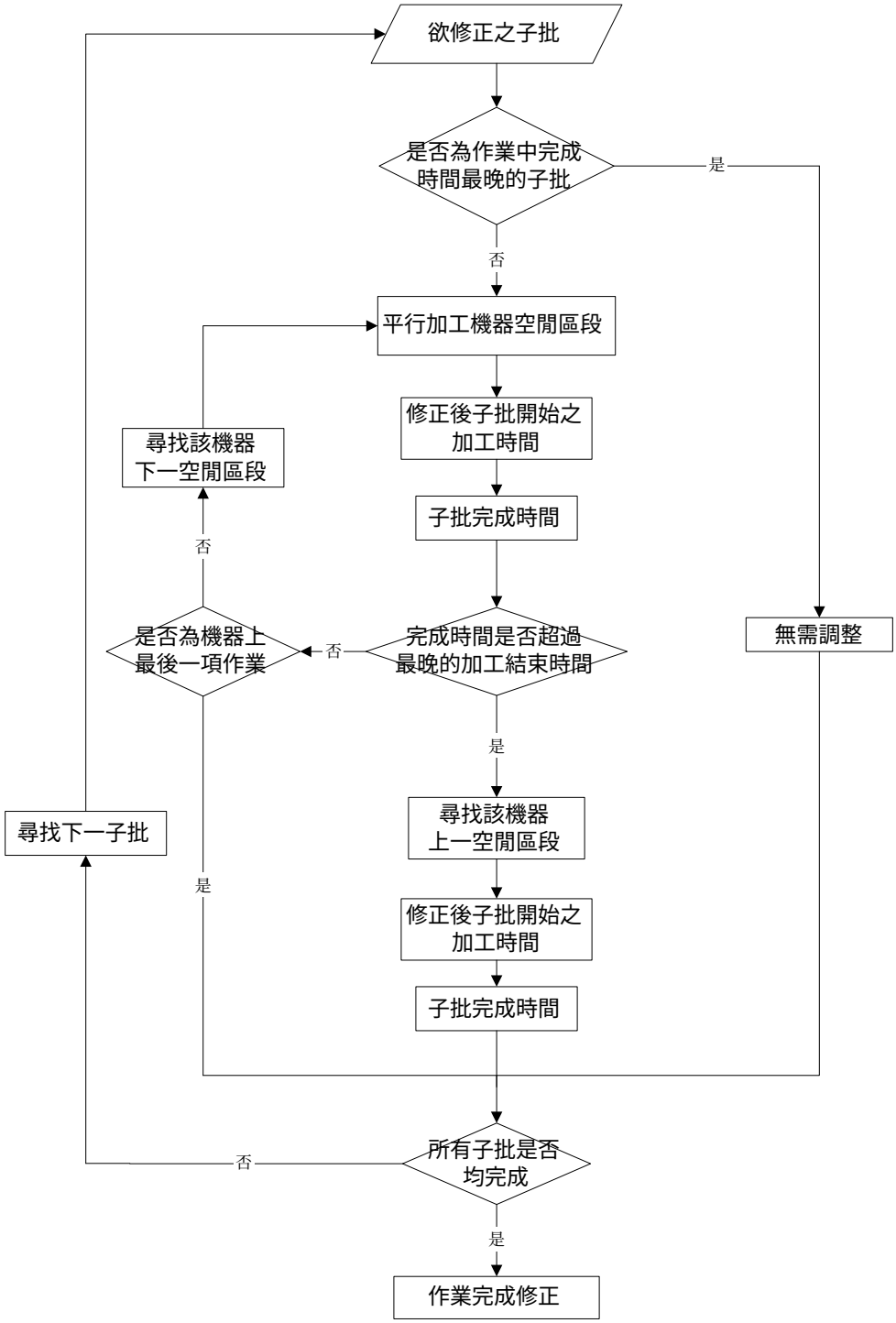

圖 3.6 作業的時間修正流程圖

3.2 定性因素模式架構

本研究中所指之定性因素乃是不能明確加以量化或是不需精確數量化的因素。以市場因素而言，該張訂單產品在市場中的定位將是

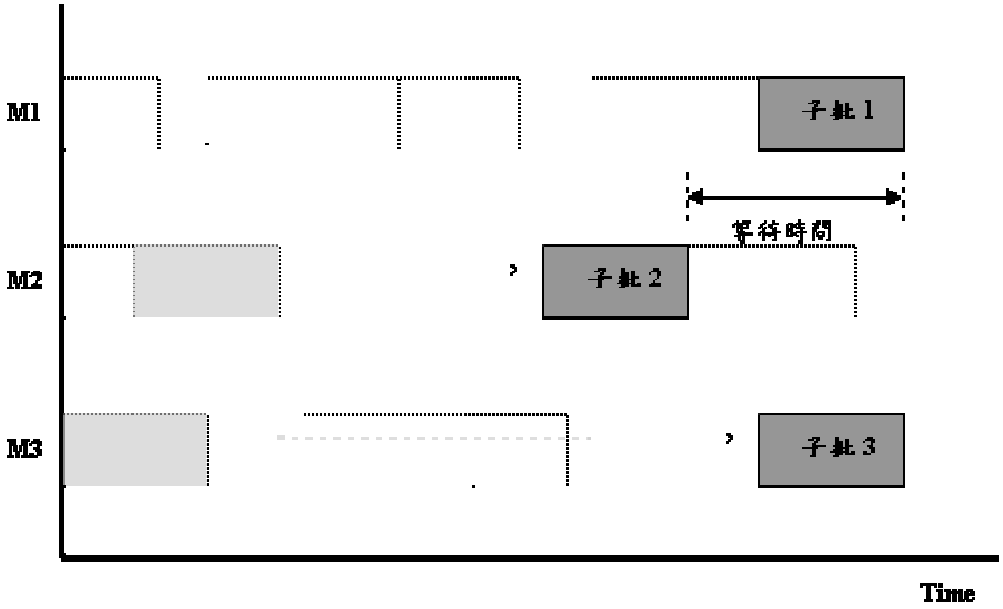

圖 3.7 修正排程甘特圖

主要關鍵，假若該產品的市場競爭者眾多，則產品完工的時效性顯的十分重要，因為其能使產品有更好的競爭優勢。換句話說，此訂單在市場因素之下應該擁有較優先的製造順序，但是此因素並不能直接加以量化。

訂單所屬的顧客也是屬於無法明確量化的定性因素，對公司而言，不同顧客的重要性必然會有程度上的差異，可能是因為彼此間有策略聯盟或是其它策略上的因素所造成。同樣地，每個顧客未來潛在的訂單利潤，也應納入此階段中應考量的因素之一。

此外，本研究中的定性因素包含不須被明確量化的因素，例如顧客以往的交易記錄與訂單的利潤。我們希望排程規劃人員在評估訂單製造的優先順序時，根據這些因素資料做主觀性的比較判斷。本研究中將利用 AHP 法來評估訂單在各個定性因素下的重要性，進而計算出訂單在定性因素下的優先順序。以下說明研究中 AHP 法的評估計算。

3.2.1 AHP 法的評估計算

關於如何設定各因素權重值的方法有很多，本研究中將利用階層式分析程序法來訂定權重值。AHP 法是由 Saaty[52]所提出，是用來評估方案或是因素間相對權重，其原理是採用配對比較法(pairwise comparison approach)來設定因素間的相對權重，將評估因素的重要性以配對的方式兩兩比較，並給予比較之分數，再加以計算各因素的權重值。

AHP 法最大的優點是在於當人們面臨多個評估因素情形下，提供決策者一客觀的方法來決定各因素的權重。同時，AHP 也可以應用於多個方案的選擇評估。AHP 法中的評估尺度如表 3.2

表 3.2 AHP 評估尺度

評估尺度	定義
1	同等重要
3	稍重要
5	重要
7	極重要
9	絕對重要
2、4、6、8	相鄰尺度的中間值
以上數值之倒數	如上定義之相對不重要程度

舉例說明如下：假設目前有 5 張訂單，而所需考量的因素有 4 個(F_1、F_2、F_3、F_4)。

步驟一：將所有因素配對以表中的評估尺度來做比較。（若規劃人員評估 F_1 較 F_2 稍不重要，評估尺度為 1/3，相反的 F_2 較 F_1 稍重要，評估尺度為 3。）

步驟二：計算此評估尺度矩陣的特徵值(eigenvalue)，取其最大的特徵值(λ_{max})。

步驟三：計算 λ_{max} 對應的特徵向量(eigenvector)。

步驟四：將此特徵向量正規化(normalize)，此正規化特徵向量即為各個因素相對的權重值(W_i)，如表 3.3。

表 3.3 各因素之權重制訂

因素	F_1	F	F_3	F_4	特徵值 $\lambda_{max}=4.11$	權重 W_i
F_1	1	1/3	2	7	0.3773	0.24
F_2	3	1	5	9	0.8976	0.58
F_3	1/2	1/5	1	5	0.2192	0.14
F_4	1/7	1/9	1/5	1	0.0628	0.04

步驟五：規劃人員在因素 F_i 的考量下，將所有訂單配對比較評估，利用步驟一至步驟四的計算方式，求出一正規化特徵向量，此即為每一張訂單在因素 F_i 下的評估值(E_i)，如表 3.4。其中 E_{ij} 表示訂單 j 在因素 i 的評估值。

表 3.4 各方案之評估值

F_i	訂單 1	訂單 2	訂單 3	訂單 4	訂單 5	特徵值 λ_{max}	評估值 E_{ij}
訂單 1	1	5	2	3	3	0.7758	0.4055
訂單 2	1/5	1	1/3	1/2	1/3	0.1277	0.0668
訂單 3	1/2	3	1	3	2	0.4912	0.2568
訂單 4	1/3	2	1/3	1	1/2	0.203	0.1061
訂單 5	1/3	3	1/2	2	1	0.3153	0.1648

步驟六：計算出所有的評估值(E_i)。

步驟七：計算各訂單的加權值，並加以排序，如表 3.5。

表 3.5 各方案之加權排序

F_i	F	F_2	F	F	加	排序
W_i	0.24	0.58	0.14	0.04		
評估值	E_{1j}	E_{2j}	E_{3j}	E_{4j}		
訂單 1	0.4055	0.30	0.10	0.15	0.2913	1
訂單 2	0.0668	0.20	0.25	0.25	0.1770	3
訂單 3	0.2568	0.20	0.30	0.15	0.2256	2
訂單 4	0.1061	0.15	0.15	0.35	0.1745	4
訂	0.1648	0.15	0.20	0.10	0.1586	5

此加總值排序順序就是所有訂單在定性因素考量下的優先順序。本階段利用 AHP 法評估每個訂單在質性因素下的重要性，亦即利用 AHP 所求得的質性訂單順序，作為遺傳演算法與螞蟻演算法中的限制。

3.3 定量因素模式架構

在定量因素的考量方面，將所有的定量因素轉換為單一多目標函數，各目標間的權重藉由 AHP 法則來制定。在本研究演算法搜尋求解的過程中，不單是利用一項指標來評估一個體解的優劣，相對的是強調如何在多個相衝突的目標中折衷求得一最適解，為了能正確利用經由 AHP 法則制定的權重值，在適應函數的設計中每一項目標評估值都應被正規化後再加權計算。本研究在定量因素方面將考量排程製距、訂單的交期滿足度及機器使用率三項衡量指標。

3.3.1 製距績效評估

製距的評估方式為第一張訂單之起始作業到最後一張訂單的最後一個作業結束所需的時間。首先由個體解排程結果求出此次排程製距，再將製距評估值正規化，製距評估值正規化的計算方式如式 3.1

$$\frac{\min MS}{MS_x} \qquad \text{式（3.1）}$$

其中，

minMS：到目前為止，搜尋過程中最小的製距

MS_x：個體解 x 的製距評估值

在搜尋求解的過程中，當找到一個體解的製距比 minMS 小，則 minMS 就會被取代，以上述方式來正規化製距評估值會造成同一個體

在不同族代的適應函數值不相同的現象發生。舉例來說，在第 n 代中，min*MS* 等於 80 且某一個體的製距為 100，製距評估值正規化後為 80/100=0.8，而如果在第(n+1)代，搜尋到一新個體的製距為 70，若前一個體有被保留至(n+1)代的族群中，則此個體在(n+1)代中的正規化製距評估值等於 70/100=0.7，會造成其適應函數值降低的現象。

雖然在演算的過程中，會發生同一個體在不同代中有不同的適應函數值。但是以相對的角度而言，在同一代中的個體適應函數值高者恆高，低者恆低，對於演算法中的機制不會造成明顯的影響。

3.3.2 交期滿足度評估

本研究中將交期定義於模糊集合上，如圖 3.8。個體解的交期滿足度評估方式為在該次排程結果中，先將每張訂單完成時間利用模糊隸屬函數求算出交期滿足度，再將該次排程中所有訂單的交期滿足度取平均值即為所求。公式如下所示：

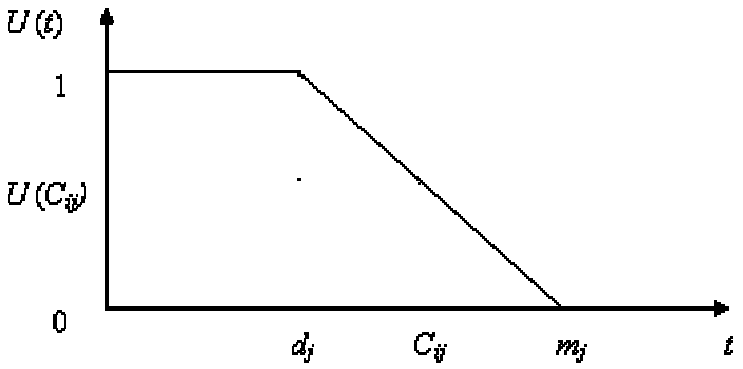

圖 3.8 交期之模糊隸屬函數

$$\begin{cases} U(C_{ij}) = 1 & , \quad C_{ij} \le d_j \\ U(C_{ij}) = \dfrac{m_j - C_{ij}}{m_j - d_j} & , \quad d_j \le C_{ij} \le m_j \\ U(C_{ij}) = 0 & , \quad m_j \le C_{ij} \end{cases} \qquad \text{式（3.2）}$$

$$D_i = \frac{\sum_{j=1}^{n} U(C_{ij})}{n} \qquad \text{式（3.3）}$$

其中，

n：該次排程之訂單數目

$U(C_{ij})$：個體解 i 的第 j 張訂單之交期滿足度

C_{ij}：個體解 i 的第 j 張訂單完成時間

d_j：訂單 j 之限定交期

m_j：訂單 j 之交期上限

D_i：個體解 i 之交期評估值

3.3.3 機器使用率評估

排程規劃時除了製距、交期的評估之外，機器使用率也是常見的績效衡量指標。關於機器使用率的評估方式是先個別加總每一機器上每項作業的加工時間，再除以機器上最後一項作業的完成時間，可得到每一機器個別之使用率，再將所有機器使用率取平均值即為所求。

3.4 遺傳演算法之運作

本節將說明研究中關於遺傳演算法的設計及其運算過程。

3.4.1 編碼表示法

本研究中遺傳演算法個體分為上半部與下半部基因，編碼方式如下：

1. 基因數目等於所有訂單作業數總和的兩倍。
2. 個體中上半部的基因，每一基因代表一作業的排程的優先順序，而基因的排列是先依照訂單再依照訂單本身的作業順序。也就是說第一個基因為訂單一的第一個作業，第二個基因為訂單一的第二個作業，直至訂單一的最後一個作業；而其下一個基因即為訂單二的第一個作業，依此類推。

3. 下半部基因的數目，與上半部相等，每一基因代表一百分比，也就是一作業在其加工途程所指定的工作中心中佔有的平行機器數的百分比。而基因的排列方式與上半部基因相同。上下半部的基因兩兩相對，也就是一作業對應了兩個基因。編碼的示意圖如圖 3.9。

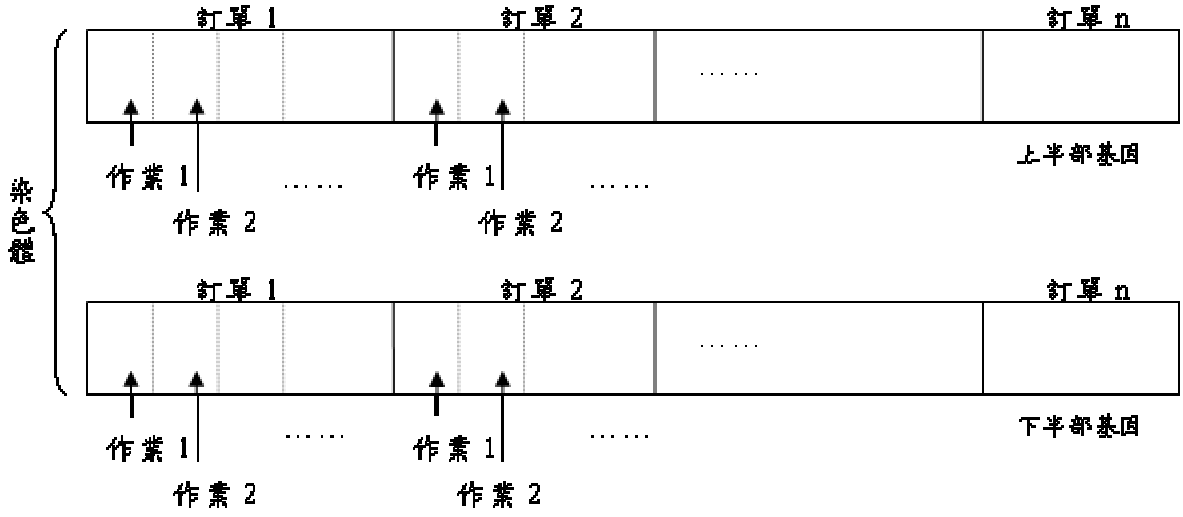

圖 3.9 編碼表示法示意圖

3.4.2 初始族群的產生

為了能讓系統在起始搜尋時，對於每一狀態空間(state space)都有同等機會，本研究將採用隨機的方式產生初始族群。假設一共有 N 個作業，亦即每一個體有 2N 個基因，對於上半部基因來說，初始族群產生方式是隨機產生一個 1~N 不重複的數字串列個體的起始基因，如圖 3.10。

	作業 1	作業 2	…	作業 m
訂單 1				
訂單 2				
…				
訂單 n				

將 1~ (n ×m)的數字以隨機方式填入

圖 3.10 上半部基因初始族群示意圖

但是，以這種方式產生初始族群會有不合理解的現象發生，例如，訂單 1 有 5 個作業，而其基因分別為[5 , 8 , 6, 10 ,12]，我們發現訂單 1 中的作業 3 因為加工途程的限制必須在作業 2 之後方可開始加工，但作業基因優先於作業 2，此個體為一不合理解，關於這一點，我們將藉由一啟發式修正的方式將其合理化。修正方式是將不合理的兩個基因做交換。在上例中，我們將 8 與 6 的互換之後的個體即為一合理解。

下半部基因代表的是百分比數值，設計上給予 1~10 十個數字，1 代表佔有工作中心 10%的機器數，2 代表佔有工作中心 20%的機器數，依此類推。初始族群產生方式為在每一基因上隨機產生 1~10 中的一個數字，如圖 3.11。

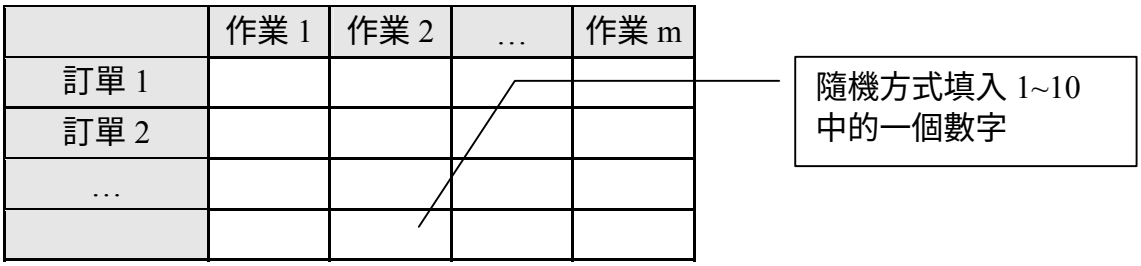

圖 3.11 下半部基因初始族群示意圖

在下半部基因的設計上，產生一動態批量分割機制。佔有的機器數多寡便決定了訂單作業分割批量的大小，佔有的機器數愈多，批量愈小，相對的佔有的機器數愈少，批量愈大。因本研究考慮上機批量限制的關係，上述產生初始族群的方法也會有不合理解的現象。例如，訂單 1 的總批量為 120，其作業 1 指定的工作中心有十台機器，上機批量限制為 35，而作業 1 的基因為[4]，代表作業 1 需佔有工作中心 40%的加工機器，也就是 4 台機器，所以每台機器上的加工批量為 120/4=30，沒有達到上機批量限制，此個體為一不合理解，修正的方式是將基因碼逐次減 1，直到達到上機批量。上例中，將 4 減 1 得到基因碼為 3，加工批量變為 40，此個體即為一合理解。

3.4.3 適應性函數之設計

本研究所設計的適應性函數由兩個部分構成：多目標函數與懲罰函數。多目標函數是針對製造現場中的績效衡量指標，懲罰函數則是利用 AHP 法所求得質性訂單順序計算轉換而來。由於是利用遺傳演算法來搜尋出較佳的作業順序，其中每一個體皆是一組作業順序，將其轉換為訂單順序，算出其與質性訂單順序的差異作為懲罰函數。

懲罰函數之制訂

在遺傳演算法運作的過程中，我們是將第 3.2 節中所計算出之定性因素下訂單順序作為限制，演算所得的個體基因再與此訂單順序比較，研究中以懲罰函數來處理違反定性因素訂單順序限制的情形。首先，我們必須求出個體解中訂單的順序，計算方式是由個體的基因順序算出每張訂單平均的基因順序，將其排序。懲罰函數的計算方式是加總排序後的訂單順序與定性因素訂單順序之間的總差異平方和，再將此懲罰值正規化，將懲罰值除以可能的最大差異平方和。正規化的計算如式 3.4：

(i)當訂單數 n 為偶數時，其可能的最大差異平方和是　　式（3.4）

$$2[(n-1)^2+(n-3)^2+\cdots+3^2+1^2]$$

(ii)當訂單數 n 為奇數時，其可能最大差異平方和是

$$2[(n-1)^2+(n-3)^2+\cdots+4^2+2^2]。$$

例如當訂單數 n=7 時，且其定性因素訂單順序為：訂單 1　5　3　4　7　2　6

其有最大的差異平方和的順序是：訂單 6　2　7　4　3　5　1

可能最大差異平方和等於 $2\ [\ 6^2+4^2+2^2\ ]=56$

以 5 張訂單舉例說明如下：假設這些訂單透過 AHP 得到在質性

因素的訂單順序為：（訂單 3）→（訂單 5）→（訂單 2）→（訂單 1）→（訂單 4），而由遺傳演算法所得的個體解中的基因與訂單順序的計算如表 3.6，懲罰函數之計算如表 3.7。

表 3.6 訂單平均基因順序

	作業 1	作業 2	作業 3	作業 4	作業 5	加總	平均基因順序	排序
訂單	1	6	11	16	21	55	11	1
訂單	2	7	12	17	22	60	12	2
訂單 3	3	8	13	18	23	65	13	3
訂單 4	4	9	14	19	24	70	14	4
訂單 5	5	10	15	20	25	75	15	5

表 3.7 懲罰函數之計算

個體解中的順序	訂單 1	訂單 2	訂單 3	訂單 4	訂單 5	總差異量
	訂單 3	訂單 5	訂單 2	訂單 1	訂單 4	
差異量	$(-2)^2$	$(-3)^2$	1^2	3^2	1^2	24

然後將總差異量正規化

$24 / 2(4^2+2^2)=0.6$

而為了能正確的利用由 AHP 法所計算出的定性因素與定量因素權重，所以整個適應函數值計算如式 3.5

$$f(x) = W_1 \cdot (\Sigma\, \alpha_i \cdot f_i) + W_2 \cdot (1 - p(x)) \qquad \text{式（3.5）}$$

其中，

$f(x)$：個體解 x 的適應函數值。

$p(x)$：個體解 x 的懲罰函數值。

W_1：定量因素的權重。

W_2：定性因素的權重。

α_i：定量因素 i 的權重。

f_i：定量因素 i 的評估值。

$W_1 + W_2 = 1$，$\sum \alpha_i = 1$。

3.4.4 交配運算子設計

由於染色體中包含了作業排程的優先順序與在其加工途程所指定的工作中心中佔有的平行機器數的百分比兩項資訊，因此一般常用排序問題之交配運算子已無法使用，本研究引用[54]所提出的MCUOX(Multi-Component Uniform Order-Based Crossover Operator)交配方法並加以改良，使遺傳演算法運用於多階平行機器排程模式的搜尋求解能更有效率，交配的步驟如下：

步驟一：在育種選擇策略選出的母體中，隨機選擇二條染色體作為父代。

步驟二：隨機選取此二父代染色體其中之一。

步驟三：由上半部基因的第一個基因開始搜尋，將被選取的父代中還未被指派至子代的作業排程優先順序數字，填入子代染色體空白的上半部基因中。

步驟四：從該優先順序數字在二條父代染色體中所對應的下半部基因百分比數字，隨機選擇一個作為此作業佔有的平行機器數百分比，填入相同位置的下半部基因中。

步驟五：重複步驟二　四直到所有子代染色體中的基因被填滿數字，子代染色體便產生。

舉例來說，如圖 3.12。假設從育種選擇選出的母體中，隨機挑出了二條染色體，指定其為父代 1 與父代 2，隨機選擇一父代，假設選到父代 1，然後從父代 1 上半部基因的第一個基因開始，將未被指派的數字(數字 4)，填入子代空白的上半部基因(第一位置)中，然後隨機選擇該數字在二父代中對應的下半部基因百分比數字(數字 1 與 7)，

得到此作業的下半部基因數字為 7；重複上述步驟，假設第二次選到父代 2，從父代 2 上半部基因中，將未被指派的數字(數字 2)，填入子代空白的上半部基因(第二位置)中，隨機選擇數字 2 在二父代中對應的下半部基因百分比數字(數字 2 與 6)，得到此作業的下半部基因數字為 6；重覆上述步驟直到所有基因填滿數字，子代染色體便產生。

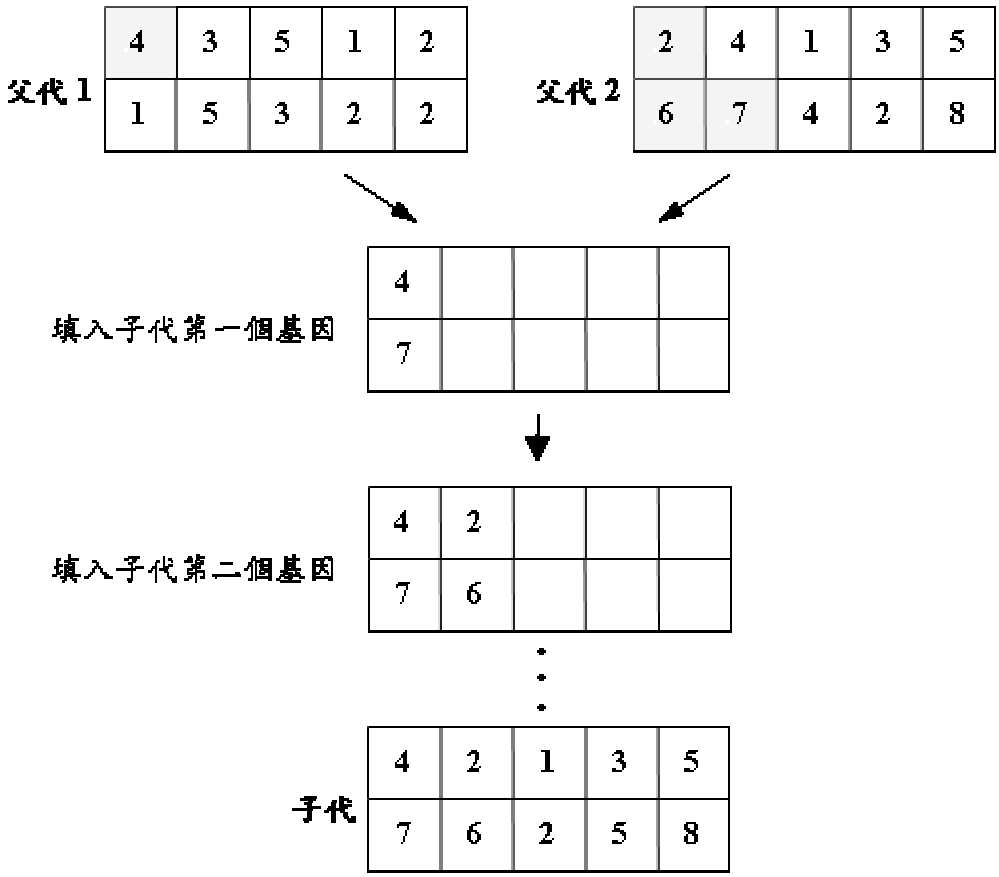

圖 3.12 交配示意圖

3.4.5 突變運算子設計

在突變運算子的設計上是以考量每一基因為主，不同於以往大部分的研究著重於考慮每一染色體的突變機率，本研究對每一基因均給予突變機率來決定是否需要進行突變，希望藉此能拓展遺傳演算法的

搜尋空間，使其搜尋能更有效率；另外由於多階平行機器排程問題的特殊型態，需考量作業排程的優先順序與其佔有的平行機器數目，因此本研究採取以下突變方法。

新產生的子代染色體中的上半部基因，若依突變機率決定要突變，則隨機選擇同一染色體中的另一上半部基因，二基因中的數字互換，而在此二基因對應的下半部基因中，則重新隨機選擇百分比數字填入，若選到與原先相同的數字，一樣將此數字填入下半部基因中。

下圖所示即為此染色體上半部基因中的第二位置依突變機率決定要突變，再隨機選擇第四位置，兩位置的上半部基因數字交換，代表第二位置的作業與第四位置的作業排程順序互換，而其對應的下半部基因位置的數字也重新隨機產生，由 5,2 變為 7,4，代表作業佔有的平行機器數目也重新選擇。

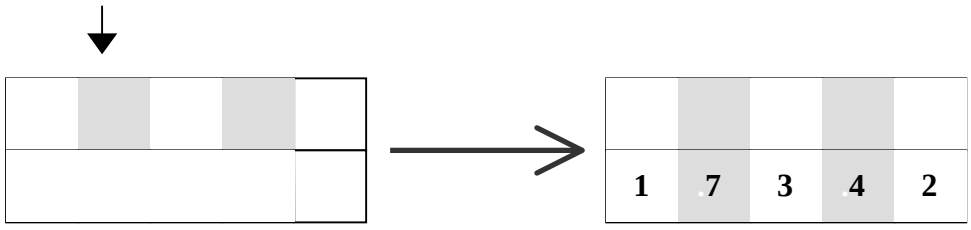

圖 3.13 突變示意圖

3.4.6 育種選擇

在遺傳演算法中，選取個體來產生下一子代，通常是藉由輪盤法(roulette wheel)來進行此一機制。而關於輪盤法中每個槽(slot)大小的設計方式一般常用的是直接以個體的適應函數值來設計，如式 3.6

$$P(x_i) = \frac{f(x_i)}{\sum_i f(x_i)} \qquad \text{式（3.6）}$$

其中，

$P(x_i)$：個體 x_i 被選中的機率。

$f(x_i)$：個體 x_i 的適應函數值。

但是，以這種方式來執行選擇機制，隨著族群演化，族群中個體的適應函數值逐漸提高，而且彼此之間的差距愈趨減小，如此一來，將會造成最佳解與最差解被選中的機率接近 1：1，使得整個演算過程中族群的適應函數值發生停滯。

基於上述，本研究中在育種選擇機制上採用另一種方式，以排序法(Ranking)來設計輪盤法中槽的大小。這種方式是先將族群中的個體依照其適應性函數值由高排至低，並給予排序的次序，每一個體再依其排序來決定其在輪盤中所佔槽之大小，如式 3.7。

$$P(x_i) = \frac{R(x_i)}{\sum_i R(x_i)} \qquad \text{式（3.7）}$$

其中，

$P(x_i)$：個體 x_i 被選中的機率。

$R(x_i)$：個體 x_i 的排序值。

一般利用遺傳演算法求解多目標問題，在育種選擇的機制中通常會採用保留精華的策略(elite preserve strategy)。所謂精華保留策略是指在產生下一族群時，除了保留適應函數值較高者（總體目標），另外再保留各單一目標評估值較佳的個體解至下一代，藉此讓族群中保留有各種目標的優良個體，再透過交配運算子來產生有更好適應函數值的個體解。基於此一點，本研究中也將採用精華保留策略，讓系統的搜尋過程更有效率。

3.5 螞蟻演算法之運作

本節將說明研究中關於螞蟻演算法的設計及其運算過程。

3.5.1 多階平行機器排程問題的圖示

使用螞蟻演算法進行零工式排程系統的搜尋求解之前，必須將問題以圖形表達，並定義清楚，以利演算法之運作。

本研究所欲探討的為多階平行機器零工式排程問題，在圖形的表達上簡化如圖 3.14 所示。圖中除了標號 0 的結點代表排程起始點，與每一訂單的第 1 項作業以單向路徑相連之外，其餘黑色的結點均代表一作業，相互之間沒有直接的路徑連接。標號 1-1 的結點代表訂單 1 的第 1 項作業，標號 1-2 的結點代表訂單 1 的第 2 項作業，其餘類推；而灰色結點代表百分比數值，也就是作業在其加工途程所指定的工作中心中佔有的平行機器數的百分比。本研究採用十個不同的百分比數值，依序由 10%、20%・・・到 100%，此處為了說明方便，僅圖示了三個結點，標號 1 代表 10%，標號 2 代表 20%，標號 3 代表 30%，而除了排程起始點外，灰色結點均與每一黑色結點有雙向路徑連接，但灰色結點彼此間並沒有任何的路徑相連。

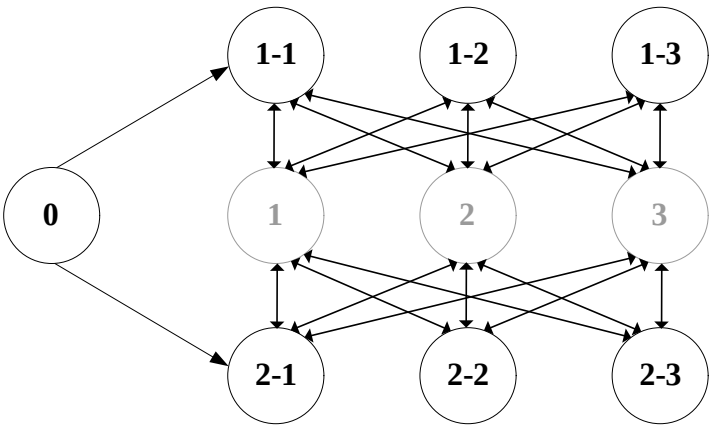

圖 3.14 多階平行機器排程問題的圖示

3.5.2 記憶陣列之設計

為了能確保螞蟻演算法能搜尋到合理解，不違反作業加工途程的限制，因此需要使用記憶陣列來指引並記錄螞蟻搜尋的軌跡，以便讓螞蟻能選擇出正確的結點，走向一合理的路徑，完成解的搜尋。本研究讓每一隻螞蟻配備有四組記憶陣列：

1. 第一組為 *G* 陣列。包含了螞蟻尚未搜尋過的結點中代表作業的結點，也就是記錄尚未排入排程的作業，前一節的例子中，第 k 隻螞蟻在排程起始點所配備的 *G* 陣列為 $G_k=\{O_{11}, O_{12}, O_{13}, O_{21}, O_{22}, O_{23}\}$，其中 O_{11} 代表標號為 1-1 的結點，O_{12} 代表標號為 1-2 的結點，其餘類推。
2. 第二組為 *S* 陣列。包含了螞蟻在走下一步時容許搜尋的所有結點，此陣列限制了螞蟻搜尋的路徑，使其能符合每一訂單作業加工途程的限制，例子中在排程起始點所配備的 *S* 陣列為 $S_k=\{O_{11}, O_{21}\}$。
3. 第三組為 *Tabu1* 陣列。包含了螞蟻已經搜尋過的結點中代表作業的結點，也就是記錄已排入排程的作業，例子中在排程起始點所配備的 *Tabu1* 陣列為 $Tabu1_k=\{\ \}$，為一空集合。
4. 第四組為 *Tabu2* 陣列。同樣也包含了螞蟻已經搜尋過的結點，不同的是這些結點是代表百分比數值的結點，也就是此陣列是紀錄已排入排程的作業在其加工途程所指定的工作中心中佔有的平行機器數的百分比，例子中在排程起始點所配備的 *Tabu2* 陣列為 $Tabu2_k=\{\ \}$，也為一空集合。

在螞蟻進行每一階段的搜尋並選擇結點之後，均需更新此螞蟻配備的記憶陣列，更新步驟如下：

步驟一：依據 *S* 陣列的資訊，可知此階段容許搜尋的結點（作業）為

何，代入轉移機率公式（下一節中會提及）中計算，當一結點（作業）被選定後，將此結點（作業）加入 *Tabu1* 陣列，同時在 *G* 陣列與 *S* 陣列中將此結點（作業）刪除。

步驟二：如果選擇的結點（作業）不是其所屬訂單中的最後一項作業，便將其後一項作業代表的結點加入 *S* 陣列中。

步驟三：根據另一轉移機率公式（下一節中會提及），再選定一結點（百分比數值），將此結點（百分比數值）加入 *Tabu2* 陣列。

步驟四：重複以上步驟直到 *G* 陣列為空集合，最後的 *Tabu1* 陣列與 *Tabu2* 陣列所提供的資訊即為多階平行機器零工式排程問題的排程解。

同樣地利用上一節的例子說明，在排程起始時，依據 *S* 陣列=$\{O_{11}, O_{21}\}$，可知容許搜尋的結點（作業）為標號 1-1 的結點與標號 2-1 的結點，假設選定的結點為 O_{11}，*Tabu1* 陣列便新增了 O_{11} 變為$\{O_{11}\}$，*G* 陣列與 *S* 陣列則分別刪除 O_{11} 而更新為$\{O_{12}, O_{13}, O_{21}, O_{22}, O_{23}\}, \{O_{21}\}$。因為 O_{11} 不是其所屬訂單中的最後一項作業，因此將其後一項作業 O_{12} 加入 *S* 陣列中，最後 *S* 陣列=$\{O_{21}, O_{12}\}$。在這一階段中，還需選定一結點（百分比數值），假設選到的結點為 P_2（P_2 代表標號為 2 的結點），*Tabu2* 陣列便新增了 P_2 變為$\{P_2\}$；重複上述步驟，第二階段依據 *S* 陣列=$\{O_{21}, O_{12}\}$，假設選定的結點（作業）為 O_{12}，則 *Tabu1* 陣列變為$\{O_{11}, O_{12}\}$，*G* 陣列與 *S* 陣列則更新為$\{O_{13}, O_{21}, O_{22}, O_{23}\}, \{O_{21}, O_{13}\}$。假設第二次選定的結點（百分比數值）為 P_3，*Tabu2* 陣列變為$\{P_2, P_3\}$；重複以上步驟直到 *G* 陣列為空集合，代表所有的作業已排入排程規劃，而螞蟻最後所配備的 *Tabu1* 陣列與 *Tabu2* 陣列，即為一合理排程解。

在 *Tabu2* 陣列的設計上，與遺傳演算法的下半部基因雷同，也產生一動態批量分割機制。佔有的機器數多寡便決定了訂單作業分割批量的大小，佔有的機器數愈多，批量愈小，相對的佔有的機器數愈少，

批量愈大。因考慮上機批量限制的關係，需注意的是上述 *Tabu2* 陣列產生的方法會有不合理解的現象。在上例中假設訂單 1 的總批量為 120，其作業 2 指定的工作中心有十台機器，上機批量限制為 50；而作業 2 對應的 *Tabu2* 陣列元素為 P_3，代表作業 2 需佔有工作中心 30% 的加工機器，也就是 3 台機器，所以每台機器上的加工批量為 120/3=40，沒有達到上機批量限制，此螞蟻搜尋出一不合理解。修正的方式是將標號逐次減 1，把螞蟻的搜尋路徑導向新標號所在的結點，直到達到上機批量。上例中，將 3 減 1 得到結點為 P_2，加工批量變為 60，即為一合理解。

3.5.3 轉移機率規則

本研究在螞蟻演算法進行的每一階段中，螞蟻均需分別搜尋二種性質不同的結點並選擇，以符合研究中所探討的排程模式，因此會有二種轉移機率規則。

第一種轉移機率規則如式 3.8，可決定這一階段欲排入排程的作

$$p1_{ij}(t)=\frac{[\tau_{ij}(t)]^{\alpha}\cdot[1/d_{ij}]^{\beta}}{\sum_{j\in \text{允許搜尋的結點(作業)}}[\tau_{ij}(t)]^{\alpha}\cdot[1/d_{ij}]^{\beta}} \qquad \text{式（3.8）}$$

其中，

τ_{ij}：在結點 i 與結點 j 間的路徑上所殘留的費洛蒙數量。

$1/d_{ij}$：結點 i 與結點 j 間的路徑的能見度。

$p1_{ij}$：螞蟻在結點 i 上，移往結點 j 的機率。

業為何。而因為本研究所探討的為一多目標問題，因此在能見度的設計上也為一多目標函數，不只考量作業的加工時間，還納入了訂單交期與定性因素的訂單優先順序，其中交期與定性因素排序二項數值事

先需經過正規化成為評估值型式，計算方式如式 3.9 與式 3.10。所以整個能見度函數值計算如式 3.11。

$$DD_j = \frac{(\max PT)\cdot(dd_j)}{\max dd} \qquad \text{式（3.9）}$$

$$QS_j = \frac{(\max PT)\cdot(qs_j)}{\max qs} \qquad \text{式（3.10）}$$

其中，

DD_j：結點 j 所屬訂單的交期評估值

QS_j：結點 j 所屬訂單的定性因素排序評估值

maxPT：最長的作業加工時間

maxdd：最晚的訂單交期

maxqs：最大的定性因素訂單排序

dd_j：結點 j 所屬訂單的交期

qs_j：結點 j 所屬訂單的定性因素排序

$$d_{ij} = V_1 \cdot (PT_j) + V_2 \cdot (DD_j) + V_3 \cdot (QS_j) \qquad \text{式（3.11）}$$

其中，

PT_j：結點 j 的作業加工時間

V_1：加工時間的權重

V_2：交期的權重

V_3：定性因素訂單優先順序的權重

$V_1 + V_2 + V_3 = 1$

第二種轉移機率規則如式 3.12，可決定這一階段排入排程的作業

$$p2_{ij}(t)=\frac{[\tau_{ij}(t)]^{\alpha}}{\sum_{\substack{i=欲排入排程的結點(作業)\\ j\in 允許搜尋的結點(百分比)}}[\tau_{ij}(t)]^{\alpha}} \qquad \text{式（3.12）}$$

在其加工途程所指定的工作中心中佔有的平行機器數的百分比。與式 3.8 不同的地方在於結點 *i* 代表欲排入排程的作業；結點 *j* 則代表此作業佔有平行機器數的百分比比例，與前式中的結點 *j* 性質不同，也因此在第二種轉移機率規則中無需考量能見度，只根據路徑上殘留的費洛蒙數量多寡進行機率選擇。

一般利用螞蟻演算法求解問題時，為了防止搜尋過早陷入區域最佳解，會加入變異運算子（variation），當此運算子被觸發，便取代轉移機率規則，幫助螞蟻在搜尋空間中拓展出不同的求解方向。基於此原因，本研究在轉移機率規則中也加入變異運算子。在第一種轉移機率規則中，假如變異運算子被觸發，螞蟻將不考量路徑上費洛蒙多寡，而僅靠能見度指引其搜尋；第二種轉移機率規則中沒有能見度的考量，當觸發了變異運算子，同樣的，螞蟻也不考量路徑上費洛蒙的多寡，因此在無任何資訊提供的情況之下，其求解過程將變為隨機搜尋。

3.5.4 費洛蒙更新法則

利用螞蟻演算法來搜尋零工式排程問題，在路徑的費洛蒙更新上，全體更新法（Global Update Rule）中的精華策略（elitist strategy）已被證實比其他更新法則優良，因此本研究也採用此法。

當一個循環中所有螞蟻均搜尋出完整解，也就找到了經過結點的順序，此時便可以更新搜尋路徑上的費洛蒙，費洛蒙全體更新法則如式 3.13。此更新法則包含二個動作，首先是讓所有路徑上的費洛蒙蒸發一部份，接著是在到目前為止構成最佳排程解的路徑上，增加其費

洛蒙，而增加的費洛蒙數量計算如式 3.14。因為本研究為一多目標問

$$\tau_{ij}(t+n) = (1-\rho)\cdot\tau_{ij}(t) + \rho\cdot\Delta\tau_{ij}(t+n) \qquad \text{式（3.13）}$$

$$\Delta\tau_{ij}(t+n) = \begin{cases} \dfrac{Q}{L(best\_so\_far)} & ,if \ \ edge\,ij \in best\_so\_far\ solution \\ 0 & ,otherwise \end{cases} \qquad \text{式（3.14）}$$

其中，

ρ：蒸發係數

Q：每單位距離的費洛蒙數量

L *(best_so_far)*：到目前為止最好的製距評估值

題，因此必須將螞蟻演算法搜尋完畢得到的多目標適應性函數（其設計方式與遺傳演算法中的適應性函數相同）調整為製距評估值型態，調整公式如式 3.15。

$$L(best\_so\_far) = \left(\frac{1}{f(best\_so\_far)}\right)\cdot AC \qquad \text{式(3.15)}$$

其中，

f *(best_so_far)*：到目前為止最好的適應函數值

AC：調整係數

第四章 系統實證

本章節將以一個 Job shop 排程問題為例，依據第三章所述的系統架構來進行驗證。首先於 4.1 小節中說明實證問題之假設；4.2 小節說明定性因素評估與各績效指標權重值訂定過程；4.3 小節對相關參數進行介紹；4.4 小節檢定遺傳演算法與螞蟻演算法；4.5 小節中，對最後的排程結果作一探討分析。

4.1 實證問題說明

在實證中 Job shop 排程環境假設為：現場有 8 種不同種類的機器，每一種類的機器有 1~3 台不等的平行機器，平行機器的類型屬於相同機器，本次排程規劃週期中有 10 張訂單，每一訂單均代表一種產品。為了能更接近現實環境的情況，針對每一種類產品的各加工途程，制定上機批量數的下限，如果切割出的批量數沒有大於此限制，代表加工此批量不符合經濟效益，便不予以開機加工；而訂單之交期為一線性模糊的三角模糊數，與傳統 Job shop 不同的是每張訂單的作業數並不相同。訂單的資料如表 4.1，每一種類機器的平行機器數如表 4.2。

4.2 定性因素評估

在定性因素考量方面，本章實證中將考量訂單本身的利潤、該訂單顧客以往的歷史交易、該訂單產品在市場上的因素以及該訂單顧客未來的潛在訂單四項因素。定量因素上，以排程的製距、訂單交期滿足度以及機器使用率作為評估指標。而適應性函數中定性與定量以及各定性因素、定量因素之權重必須由規劃人員評判各因素之重要性做配對比較(如表 4.3、表 4.4、表 4.5)，關於定性因素也是由規劃人員

表 4.1 訂單資料

訂	單位數	作業	加工機器種類	批量限制	加工時間	交期時間	交期時間
1	120	1	3	55	0.183	55	65
		2	4	75	0.067		
		3	7	57	0.175		
		4	1	40	0.125		
		5	6	83	0.108		
		6	8	48	0.167		
2	85	1	8	22	0.271	100	110
		2	3	35	0.200		
		3	1	27	0.165		
		4	2	37	0.188		
		5	5	55	0.200		
		6	7	27	0.224		
3	200	1	3	38	0.105	104	114
		2	2	89	0.090		
		3	4	186	0.035		
		4	5	38	0.080		
		5	8	100	0.120		
4	140	1	6	51	0.079	121	131
		2	5	41	0.121		
		3	4	98	0.071		
		4	2	118	0.136		
		5	7	62	0.129		
		6	1	44	0.114		
5	90	1	1	30	0.167	60	70
		2	3	39	0.211		
		3	7	27	0.211		
		4	5	20	0.200		
6	135	1	1	103	0.126	72	82
		2	4	75	0.067		
		3	2	44	0.126		
		4	8	56	0.178		
		5	5	28	0.141		
		6	6	90	0.089		

7	155	1	2	31	0.097	80	90
		2	7	47	0.129		
		3	1	49	0.142		
		4	8	36	0.135		
		5	5	15	0.148		
8	170	1	4	146	0.082	110	120
		2	6	57	0.088		
		3	7	50	0.100		
		4	2	80	0.100		
		5	3	73	0.082		
		6	8	54	0.129		
9	210	1	2	70	0.043	77	87
		2	1	22	0.090		
		3	7	63	0.095		
		4	5	90	0.033		
		5	4	196	0.071		
		6	3	92	0.105		
10	105	1	8	32	0.124	50	60
		2	5	67	0.105		
		3	3	49	0.143		
		4	2	39	0.076		
		5	6	47	0.190		

表 4.2 平行機器數目

機器種類	**1**	**2**	**3**	**4**		**6**	**7**	
數目	3	2	2	1	3	2	3	3

依據各因素來對訂單作兩兩配對比較(如表 4.6、表 4.7、表 4.8、表 4.9)。最後計算出各訂單在定性因素下的加權值，求出定性因素之訂單順序（表 4.10）。所以，在本章實證中定性因素之訂單順序為

訂單 4=>訂單 1=>訂單 3=>訂單 9=>訂單 8=>訂單 7=>訂單 6=>

訂單 2=>訂單 5=>訂單 10

由表 4.3 與 4.5 可得，遺傳演算法與螞蟻演算法進行時的適應性函數如下：

fitness = 0.75(0.28 製距＋0.65 交期滿足＋0.07 機器使用率)

＋0.25(1－懲罰函數值)

表 4.3 定性與定量之權重值

	定性因素	定量因素	λ＝1.995	評估值
定性因素	1	1/3	0.3148	0.25
定量因素	3	1	0.9492	0.75

表 4.4 定性因素之權重值

定性因素	訂單利潤	歷史交易	市場因素	潛在訂單	λ＝4.1224	評估值
訂單利潤	1	3	2	5	0.8028	0.47
歷史交易	1/3	1	1/3	3	0.2683	0.16
市場因素	1/2	3	1	3	0.5153	0.30
潛在訂單	1/5	1/3	1/3	1	0.1342	0.07

表 4.5 定量因素之權重值

定量因素	製距	機器使用率		λ＝3.0537	評估值
製距	1	5	1/3	0.3916	0.28
機器使用率	1/5	1	1/7	0.1005	0.07
交期滿足	3	7	1	0.9146	0.65

表 4.6 定性因素-各訂單利潤之評估值

利潤	Order_1	Order_2	Order_3	Order_4	Order_5	Order_6	Order_7	Order_8	Order_9	Order_10	λ＝16.2806	評估值
Order_1	1	3	1/2	1	4	5	4	5	3	5	0.4572	0.1554
Order_2	1/3	1	1/3	1/2	2	4	1	3	1/2	4	0.233	0.0792
Order_3	2	3	1	2	4	5	4	4	3	5	0.5148	0.175
Order_4	1	2	1/2	1	4	5	3	4	2	5	0.4009	0.1363
Order_5	4	1/2	1/4	1/4	1	3	1/2	2	1/3	1/2	0.2241	0.0762
Order_6	5	1/4	1/5	1/5	1/3	1	1/3	1/2	1/4	1/2	0.193	0.0656
Order_7	1/4	1	1/4	1/3	2	3	1	2	1/2	3	0.1868	0.0635
Order_8	1/5	1/3	1/4	4	1/2	2	1/2	1	1/3	1/2	0.1777	0.0605
Order_9	1/3	2	1/3	2	3	4	2	3	1	4	0.3193	0.1085
Order_10	1/5	1/4	1/5	5	2	2	1/3	2	1/4	1	0.2348	0.0798

表 4.7 定性因素-各訂單顧客歷史交易之評估值

歷史交易	Order_1	Order_2	Order_3	Order_4	Order_5	Order_6	Order_7	Order_8	Order_9	Order_10	λ＝10.3237	估值
Order_1	1	2	1/3	1/4	2	3	1/3	3	1/2	4	0.2215	0.0852
Order_2	1/2	1	1/3	1/4	1	2	1/3	2	1/2	3	0.1554	0.0598
Order_3	3	3	1	1/2	3	4	1	5	2	5	0.4423	0.17
Order_4	4	4	2	1	4	4	2	5	3	5	0.638	0.2454
Order_5	1/2	1	1/3	1/4	1	2	1/3	2	1/2	3	0.1554	0.0598
Order_6	1/3	1/2	1/4	1/4	1/2	1	1/4	2	1/3	2	0.1082	0.0416
Order_7	3	3	1	1/2	3	4	1	5	2	5	0.4423	0.17

Order_8	1/3	1/2	1/5	1/5	1/2	1/2	1/5	1	1/4	2	0.0851	0.0328
Order_9	2	2	1/2	1/3	2	3	1/2	4	1	4	0.2845	0.1094
Order_10	1/4	1/3	1/5	1/5	1/3	1/2	1/5	1/2	1/4	1	0.0676	0.026

表 4.8 定性因素-各訂單市場考量之評估值

	Order_1	Order_2	Order_3	Order_4	Order_5	Order_6	Order_7	Order_8	Order_9	Order_10	λ 37	評估值
Order_1	1	4	3	2	5	3	4	1/2	2	5	0.4753	0.1839
Order_2	1/4	1	1/2	1/3	3	1/2	2	1/5	1/3	2	0.1284	0.0497
Order_3	1/3	2	1	1/3	4	1/2	3	1/4	1/2	3	0.1841	0.0712
Order_4	1/2	3	3	1	5	2	4	1/2	2	4	0.3751	0.1452
Order_5	1/5	1/3	1/4	1/5	1	1/4	1/2	1/7	1/4	1/2	0.0599	0.0232
	1/3	2	2	1/2	4	1	3	1/4	1/2	4	0.2269	0.0878
Order_7	1/4	1/2	1/3	1/4	2	1/3	1	1/5	1/4	2	0.096	0.0372
Order_8	2	5	4	2	7	4	5	1	3	5	0.65	0.2515
Order_9	1/2	3	2	1/2	4	2	4	1/3	1	4	0.296	0.1145
Order_10	1/5	1/2	1/3	1/4	2	1/4	2	1/5	1/4	1	0.0924	0.0358

表 4.9 定性因素-各訂單顧客潛在訂單之評估值

潛在訂單	Order_1	Order_2	Order_3	Order_4	Order_5	Order_6	Order_7	Order_8	Order_9	Order_10	λ 26	評估值
Order_1	1	0.5	0.25	0.2	0.5	0.33	0.25	2	0.25	3	0.1076	0.0414
Order_2	2	1	0.33	0.2	1	0.2	0.33	2	0.5	3	0.1456	0.056
Order_3	4	3	1	0.5	3	2	1	4	0.5	5	0.3508	0.1349
Order_4	5	5	2	1	5	3	2	5	2	7	0.625	0.2403

Order_5	2	1	0.33	0.2	1	0.5	0.33	2	0.25	3	0.1403	0.054
Order_6	3	2	0.5	0.33	2	1	0.5	3	0.33	4	0.2259	0.0869
Order_7	4	3	1	1/2	3	2	1	4	1/2	5	0.3508	0.1349
Order_8	1/2	1/2	1/4	1/5	1/2	1/3	1/4	1	1/5	2	0.0853	0.0328
Order_9	4	4	2	1/2	4	3	2	5	1	5	0.4974	0.1913
Order_10	1/3	1/3	1/5	1/7	1/3	1/4	1/5	2	1/5	1	0.0716	0.0275

表 4.10 各訂單定性因素之評估值

	利	歷史交易	市場考量	在訂單	加權	排	訂單順序
權	0.47	0.16	0.30	0.07			
Order_1	0.1554	0.0852	0.1839	0.0414	0.1447	2	4
Order_2	0.0792	0.0598	0.0497	0.056	0.0656	8	1
Order_3	0.175	0.17	0.0712	0.1349	0.1403	3	3
Order_4	0.1363	0.2454	0.1452	0.2403	0.1637	1	9
Order_5	0.0762	0.0598	0.0232	0.054	0.0562	9	8
Order_6	0.0656	0.0416	0.0878	0.0869	0.0699	7	7
Order_7	0.0635	0.17	0.0372	0.1349	0.0776	6	6
Order_8	0.0605	0.0328	0.2515	0.0328	0.1114	5	2
Order_9	0.1085	0.1094	0.1145	0.1913	0.1162	4	5
Order_10	0.0798	0.026	0.0358	0.0275	0.0544	10	10

4.3 基本參數設定

在進行演算法運作之前，我們先就本機制中的相關參數進行介紹。在遺傳演算法的基本參數設定方面，族群大小為 20 條個體，個體中的基因編碼在 3.4.1 節中已提及；而在突變機率的制定上，因本研究考慮每一基因的突變，因此給予較小的突變機率，以符合突變的原則。其他相關設定如表 4.11 所示。

表 4.11 遺傳演算法中參數之設定值

參數	參數值
群體大小	20
演算代數	300
交配方式	MCUOX 之改良
	0.0009
精華保留個體數	1

而在螞蟻演算法的相關參數設定方面，因其參數較多，而本研究主要為發展系統架構，不著重於相關參數的最佳化設計，因此依據 Zwaan 和 Marques[63]所提出的參數數值與參數制定的原則，並對本實證例子進行測試後，得到基本參數的設定值如表 4.12 所示。

為了要與遺傳演算法在下一節中比較求解結果, 因此群體的螞蟻數與染色體數目相同。能見度的權重值之所以為 0，是因為在本研究多目標平行機器模式的轉移機率規則中加入能見度的考量並無法幫助螞蟻演算法搜尋出更好的解，甚至會更差，因此只考慮費洛蒙權重。而變異運算子的變異機率同樣是為了要和遺傳演算法比較而決定出來的。

表 4.12 螞蟻演算法中參數之設定值

參數種類	參數	參數值
演算參數	群體螞蟻數	20
	演算次數	360
參數	初始之路徑費洛蒙數量	1
	費洛蒙權重值	10
	能見度權重值	0
	變異機率	0.0018
費洛蒙更新參數	費洛蒙更新方式	Best so far
	費洛蒙蒸發係數	0.01
	螞蟻釋放的費洛蒙數量	91
	精華螞蟻數	1
	調整係數	75

4.4 遺傳演算法與螞蟻演算法之檢定

本小節將以遺傳演算法與螞蟻演算法兩種搜尋方式在相同的搜尋時間下所得之最好的排程解進行檢定。

圖 4.1 為 GA 搜尋過程中適應函數值趨勢圖，雖然在研究中有採用精華保留，理論上在搜尋過程中每一代之最佳解的適應性應該為遞增的數列，但是在圖 4.1 中，我們發現每一代之最佳解的適應值在某些族代中有降低的現象，這是因為適應性函數中關於製距指標設計所造成的現象。例如，在某一族代中最佳解的製距為 100，搜尋過程中最短的製距為 80，在此最佳解製距的評估值為 80/100＝0.8，而在下一代中找到一個體解的製距為 70，但此個體在所有因素指標的加權評估值低於原最佳解，所以無法取代原最佳解，但是此時最佳解在製距方面的指標降低為 70/100＝0.7，造成最佳解適應性降低的情形。

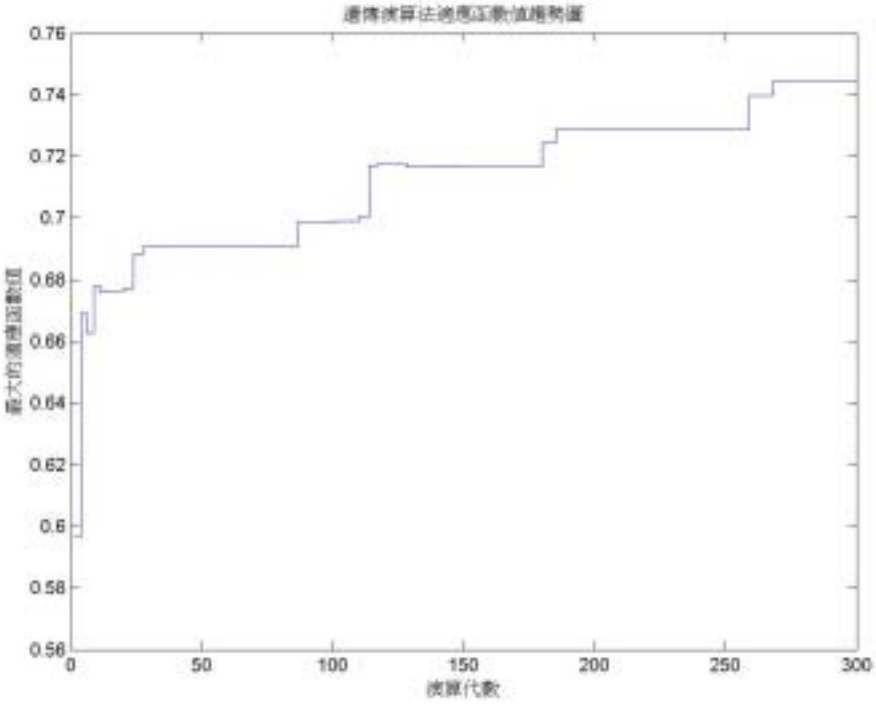

圖 4.1 遺傳演算法適應函數值趨勢圖

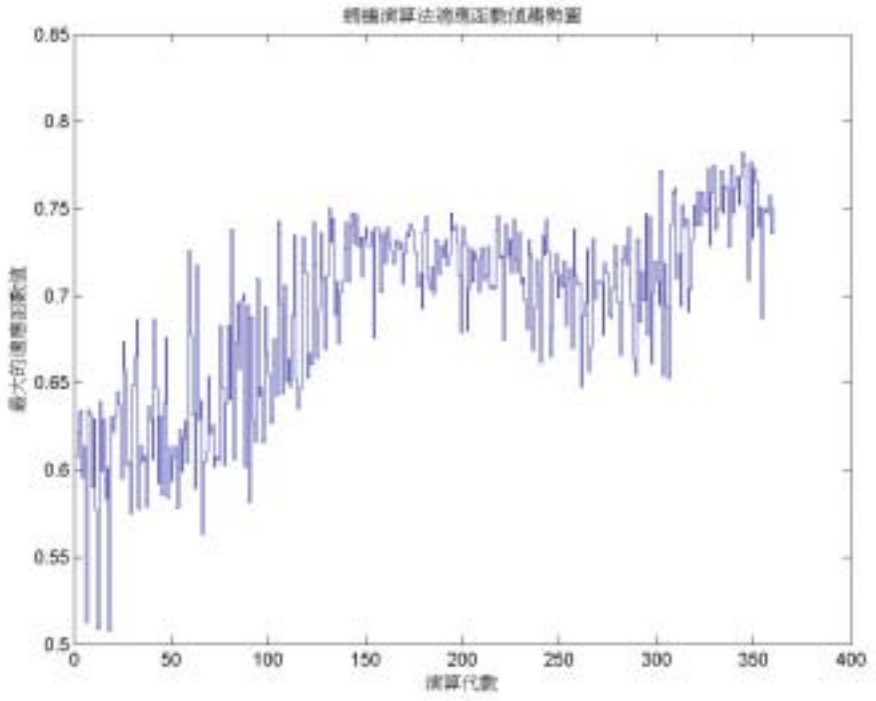

圖 4.2 螞蟻演算法適應函數值趨勢圖

圖 4.2 為螞蟻演算法搜尋過程中適應函數值趨勢圖，螞蟻演算法無類似 GA 中精華保留的機制，而是以求解路徑上殘留的費洛蒙多寡進行搜尋，而本研究較不著重於參數的最佳化設定，因此每一代最佳解的適應性較不穩定，趨勢有升有降，但是以圖 4.2 整體的最佳適應函數值走勢看來仍為一遞增的數列，螞蟻演算法在本研究中所假設的 Job shop 排程問題上仍能逐漸地搜尋到較理想的排程解。表 4.13 為 GA 與螞蟻演算法在相同的搜尋時間下分別對 Job shop 實證問題進行 15 次的實驗數據。

表 4.13 相同搜尋時間 GA 與螞蟻演算法之最好排程解

GA				螞蟻演算法			
實驗回數	最好解	實驗回數	最好解	實驗回數	最好解	實驗回數	最好解
1	0.788822	9	0.759497	1	0.773129	9	0.793058
2	0.768796	10	0.773865	2	0.737481	10	0.815765
3	0.800691	11	0.785260	3	0.811603	11	0.762574
4	0.711316	12	0.819368	4	0.710932	12	0.776420
5	0.755485	13	0.693311	5	0.768603	13	0.763677
6	0.744337	14	0.771606	6	0.762476	14	0.761402
7	0.738138	15	0.764582	7	0.715620	15	0.762633
8	0.794373	平均值	**0.7646298**	8	0.782409	平均值	**0.7665188**
標準差	**0.0332693**			標準差	**0.0294785**		

假設檢定:

$H_0 : \mu_{GA} = \mu_{ANT}$

$H_1 : \mu_{GA} \neq \mu_{ANT}$

以 $\alpha = 0.05$ 為顯著水準，

$$t = \frac{\overline{GA} - \overline{ANT}}{\sqrt{\frac{(15-1)\cdot s_{GA}^2 + (15-1)\cdot s_{ANT}^2}{15+15-2}(\frac{1}{15}+\frac{1}{15})}} = \frac{0.764630 - 0.766519}{\sqrt{((0.033269)^2 + (0.029479)^2)\cdot\frac{14}{28}\cdot\frac{2}{15}}} = -0.165$$

$\because |t| < t_{\alpha/2} = 2.05$，

所以，Do not reject H_0，

亦即在本實證問題上，在相同搜尋時間之下藉由 GA 之解所得的最好適應函數值與螞蟻演算法所得的最好適應函數值無顯著差異。

4.5 運作實例探討

以下說明 GA 的排程結果（如表 4.14）。

表 4.14 GA 排程結果

訂單	作業	加工機器種類	上半部基因	下半部基因	加工機器數	加工批量	作業	作業結束時間
1	1	3	3	5	2	60	0	11.00
	2	4	7	4	1	120	14.00	22.00
	3	7	19	6	2	60	22.00	32.50
	4	1	31	3	1	120	32.50	47.50
	5	6	32	4	1	120	53.50	66.50
	6	8	38	4	2	60	72.17	82.17
2	1	8	13	8	3	28	6.50	14.17
	2	3	18	1	1	85	39.00	56.00
	3	1	41	5	2	43	61.67	68.67
	4	2	47	9	2	43	68.67	76.67
	5	5	48	3	1	85	76.67	93.67
	6	7	55	9	3	28	93.67	100.00
3	1	3	25	5	2	100	11.00	21.50
	2	2	28	9	2	100	21.50	30.50
	3	4	34	5	1	200	57.67	64.67
	4	5	39	5	2	100	64.67	72.67
	5	8	42	6	2	100	82.17	94.17
4	1	6	10	8	2	70	0	5.50
	2	5	16	3	1	140	5.50	22.50
	3	4	17	4	1	140	31.00	41.00
	4	2	26	4	1	140	43.50	62.50
	5	7	44	6	2	70	62.50	71.50

	6	1	50	5	2	70	71.50	79.50
5	1	1	11	7	3	30	17.00	22.00
	2	3	12	8	2	45	22.00	31.50
	3	7	52	1	1	90	54.33	73.33
	4	5	54	2	1	90	73.33	91.33
6	1	1	2	3	1	135	0	17.00
	2	4	8	4	1	135	22.00	31.00
	3	2	9	8	2	68	31.00	39.50
	4	8	24	6	2	68	39.50	51.50
	5	5	43	6	2	68	51.50	61.00
	6	6	49	2	1	135	61.00	73.00
7	1	2	29	6	2	78	4.50	12.00
	2	7	35	9	3	52	47.67	54.33
	3	1	36	7	3	52	54.33	61.67
	4	8	37	5	2	78	61.67	72.17
	5	5	53	2	1	155	72.17	95.17
8	1	4	4	4	1	170	0	14.00
	2	6	20	2	1	170	14.00	29.00
	3	7	27	6	2	85	39.17	47.67
	4	2	45	3	1	170	47.67	64.67
	5	3	46	1	1	170	68.67	82.67
	6	8	51	7	3	57	94.17	101.50
9	1	2	6	6	2	105	0	4.50
	2	1	14	4	2	105	22.00	31.50
	3	7	21	10	3	70	32.50	39.17
	4	5	22	6	2	105	39.17	42.67
	5	4	33	9	1	210	42.67	57.67
	6	3	40	9	2	105	57.67	68.67
10	1	8	1	5	2	53	0	6.50
	2	5	5	3	1	105	6.50	17.50
	3	3	15	9	2	53	31.50	39.00
	4	2	23	6	2	53	39.50	43.50
	5	6	30	9	2	53	43.50	53.50

各機器安排作業的情形如表 4.15

表 4.15 機器之作業排程

加工	機器編號	工順序	加工作業		結束時間
1	1	1	6-1	0	17.00
		2	5-1	17.00	22.00
		3	9-2	22.00	31.50
		4	1-4	32.50	47.50
		5	7-3	54.33	61.67
		6	2-3	61.67	68.67
		7	4-6	71.50	79.50
	2	1	5-1	17.00	22.00
		2	9-2	22.00	31.50
		3	7-3	54.33	61.67
		4	2-3	61.67	68.67
		5	4-6	71.50	79.50
	3	1	5-1	17.00	22.00
		2	7-3	54.33	61.67
2	1	1	9-1	0	4.50
		2	7-1	4.50	12.00
		3	3-2	21.50	30.50
		4	6-3	31.00	39.50
		5	10-4	39.50	43.50
		6	4-4	43.50	62.50
		7	2-4	68.67	76.67
	2	1	9-1	0	4.50
		2	7-1	4.50	12.00
		3	3-2	21.50	30.50
		4	6-3	31.00	39.50
		5	10-4	39.50	43.50
		6	8-4	47.67	64.67
		7	2-4	68.67	76.67
3	1	1	1-1	0	11.00
		2	3-1	11.00	21.50
		3	5-2	22.00	31.50
		4	10-3	31.50	39.00

		5	2-2	39.00	56.00
		6	9-6	57.67	68.67
		7	8-5	68.67	82.67
	2	1	1-1	0	11.00
		2	3-1	11.00	21.50
		3	5-2	22.00	31.50
		4	10-3	31.50	39.00
		5	9-6	57.67	68.67
4	1	1	8-1	0	14.00
		2	1-2	14.00	22.00
		3	6-2	22.00	31.00
		4	4-3	31.00	41.00
		5	9-5	42.67	57.67
		6	3-3	57.67	64.67
5	1	1	10-2	6.50	17.50
		2	9-4	39.17	42.67
		3	6-5	51.50	61.00
		4	3-4	64.67	72.67
		5	2-5	76.67	93.67
	2	1	4-2	5.50	22.50
		2	9-4	39.17	42.67
		3	6-5	51.50	61.00
		4	3-4	64.67	72.67
		5	5-4	73.33	91.33
	3	1	7-5	72.17	95.17
6	1	1	4-1	0	5.50
		2	8-2	14.00	29.00
		3	10-5	43.50	53.50
		4	1-5	53.50	66.50
	2	1	4-1	0	5.50
		2	10-5	43.50	53.50
		3	6-6	61.00	73.00
7	1	1	1-3	22.00	32.50
		2	9-3	32.50	39.17
		3	8-3	39.17	47.67
		4	7-2	47.67	54.33
		5	4-5	62.50	71.50

		6	2-6	93.67	100.00
	2	1	1-3	22.00	32.50
		2	9-3	32.50	39.17
		3	8-3	39.17	47.67
		4	7-2	47.67	54.33
		5	4-5	62.50	71.50
		6	2-6	93.67	100.00
	3	1	9-3	32.50	39.17
		2	7-2	47.67	54.33
		3	5-3	54.33	73.33
		4	2-6	93.67	100.00
8	1	1	10-1	0	6.50
		2	2-1	6.50	14.17
		3	6-4	39.50	51.50
		4	7-4	61.67	72.17
		5	1-6	72.17	82.17
		6	3-5	82.17	94.17
		7	8-6	94.17	101.50
	2	1	10-1	0	6.50
		2	2-1	6.50	14.17
		3	6-4	39.50	51.50
		4	7-4	61.67	72.17
		5	3-5	82.17	94.17
		6	8-6	94.17	101.50
	3	1	2-1	6.50	14.17
		2	1-6	72.17	82.17
		3	8-6	94.17	101.50

排程結果：結束時間（製距）=101.50

搜尋過程中最短之製距為：100.00

將最後之個體的製距正規化為：100.00/101.50=0.9852

各訂單的交期滿足如表 4.16

表 4.16 訂單交期滿足

訂單 1	訂單 2	訂單	訂單 4	單 5	訂單	訂單 7	訂	訂單 9	訂	平均 交期滿足
0	1.00	1.00	1.00	0	0.90	0	1.00	1.00	0.65	0.655

各機器的使用率如表 4.17

表 4.17 機器使用率

機器 1-1	機器 1-2	機器 1-3	機器 2-	機器 2-2	機器 3-1	機器	機器 4-1	機器 5-1	機器 5-2
0.8658	0.4633	0.2000	0.7891	0.7630	0.9738	0.7209	0.9742	0.5231	0.6131
機器 5-3	機器 6-1	機器 6-2	機器 7-	機器 7-2	7-3	機器	機器 8-2	機器 8-3	平均機器 使用率
0.2417	0.6541	0.3767	0.4767	0.4767	0.3867	0.6502	0.5517	0.2463	0.5762

根據本次搜尋結果，各訂單的平均基因順序如表 4.18

表 4.18 訂單的平均基因順序

	訂單 1	單 2	訂單	訂單 4	5	訂單 6	7	訂單	訂單 9	訂單 10
訂單基因順序總和	130	222	168	163	129	135	190	193	136	74
訂單基因平均順序	21.67	37	33.6	27.17	32.25	22.5	38	32.17	22.67	14.8
排序	2	9	8	5	7	3	10	6	4	1

本次搜尋結果所得之訂單順序與 4.2 節中定性因素下訂單順序差異如

表 4.19

表 4.19 搜尋結果所得之訂單順序與定性因素下訂單順序差異

排序	1	2	3	4	5	6	7	8	9	10	總差異量
定性因素訂單順序	訂單 4	訂單 1	訂單 3	訂單 9	訂單 8	訂單 7	訂單 6	訂單 2	訂單 5	訂單 10	
搜尋之訂單順序	訂單 10	訂單 1	訂單 6	訂單 9	訂單 4	訂單 8	訂單 5	訂單 3	訂單 2	訂單 7	
排序差異	36	0	9	0	16	1	1	1	9	9	82

總差異量正規化：82/330=0.2485

適應函數=0.75・(0.28・0.9852 + 0.65・0.6550 + 0.07・0.5762) + 0.25・(1-0.2485)

=0.7443

而在螞蟻演算法的排程結果方面，只要將 *Tabu1* 陣列中所記錄的作業順序轉換為遺傳演算法中上半部基因的排序形式，*Tabu2* 陣列也同樣地隨 *Tabu1* 陣列次序轉換為下半部基因形式之後，便可依照上述 GA 評估適應函數值的步驟評估出螞蟻演算法的多目標函數值。

第五章 結論與未來研究方向

5.1 結論

本研究中將零工式排程問題擴充成多階平行機器模式進行探討，並考慮批量分割的機制。在多目標排程系統的設計上先計算定性因素之訂單順序，透過懲罰函數的制訂將其納入適應函數，再藉由遺傳演算法與螞蟻演算法分別進行搜尋求解。

在第 4 章中以一 Job shop 排程問題作為系統架構的驗證，根據實驗結果，本研究有下列幾項結論：

1. 以遺傳演算法與螞蟻演算法分別搜尋求解，所得到的排程解之品質（適應函數）雖然無法保證可以找到最佳解，但是藉由本研究所設計之系統運作可以找出一個讓使用者接受的近似最佳解或可接受解。同時也驗證了本研究多階平行機器多目標排程模式之可行性。
2. 在一般多目標排程的研究，其皆是以現場製造的績效，例如製距、使用率等做為評估因子，本研究除了考量現場績效，另外也將質性因素納入考量。
3. 目前在關於螞蟻演算法的研究中，多是探討單一績效衡量指標的問題，本研究發展出一多目標螞蟻演算法模式。
4. 在機器型態分佈方面，本研究發展出符合現實生產環境，複雜性極高的一般化多階平行機器排程模式。
5. 排程演算法設計中考慮作業的動態批量分割，使一作業同時在多部機器上處理，能有效地縮短總生產時間，發揮平行機器的功用。
6. 本研究考慮上機批量的限制，能更加地符合工廠中生產現況。

7. 提供以遺傳演算法與螞蟻演算法求解具多階平行機器型態多目標排程問題時的設計方式。

5.2 未來研究方向

雖然本研究經由實驗驗證，確實有達到預期之效果，但在發展本研究架構與問題實作的過程中，發現一些值得進一步探討的問題，將可使本研究更為的完善，以下歸納出幾項值得探討的未來研究方向：

1. 本研究中，關於系統之績效評估是以適應性函數的高低做為評判的準則，短期（本排程週期）而言，系統可以獲得一近似最佳解。但在長期中，系統的績效評判應該要考量其它策略方面的因素，在此，本研究建議將所提之系統架構結合系統模擬的方法來進行長期的運作。
2. 以方法論而言，本研究著重於發展系統架構，對於其中運算子並無做深入的探討與設計，但在實證的過程中，常會產生相同的個體解，所以後續研究可以針對演算法中各運算子，提出更有效率的運算子設計。
3. 研究中以 Job shop 排程問題做為驗證應用之範圍，建議在後續研究可以嘗試將本研究之系統架構擴展至更複雜的排程問題上，如彈性製造系統之排程問題。
4. 雖然在相同搜尋時間之下藉由遺傳演算法所得的解與螞蟻演算法之解所得的適應函數值無明顯地差異，但在觀察二者適應函數值變化時發現螞蟻演算法能快速地找尋出可接受的解，但搜尋時適應函數值變化的幅度較大，且有時會搜尋出較差的解，而遺傳演算法則有較穩定的解，未來或許可將二種演算法結合，在短時間內能求得更好的解。

經由實際驗證，證實本研究所提出的多階平行機器多目標排程系

統之可行性，未來後續研究可以針對上述幾點，作更深入的探討與研究。

參考文獻

[1] 吳信儀，「以改良之進化策略演算法解決排序問題之研究－SRS演算法與多重工作者系統之發展」，東海大學工業工程研究所碩士論文，1996。

[2] 林我聰，*現場排程專家系統-應用個體導向技術建立之研究*，資訊與電腦公司出版，1994。

[3] 林暘桂，「不相關平行機器總加權延遲時間最小化之排程問題」，朝陽科技大學，碩士論文，2001。

[4] 羅友廷，「模糊多目標混合式遺傳演算法在零工式排程系統之應用」，東海大學工業工程研究所碩士論文，1999。

[5] 蘇木春，*機器學習類神網路、模糊系統以及基因演算法則*，全華出版社，1997。

[6] Azizoglu, M. and Kirca, O., "Tardiness minimization on parallel machines", *International Journal of Production Economics*, Vol. 55, pp163-168, 1998.

[7] Baker, K. R., *Introduction to Sequencing and Scheduling*, John Wiley & Sons, New York, 1974.

[8] Baker, R. K., "Sequence rules and due-date assignments in a job shop", *Management Science*, Vol. 30, No. 9, 1984.

[9] Bauer, A., Bullnheimer, B., Hartl, R.F. and Strauss, C., "An ant colony optimization approach for the single machine total tardiness problem", *In Proc. of CEC'99*, IEEE Press, Piscataway, NJ, pp 1445–1450, 1999.

[10] Brah, S.A., Hunsucker, and J.L., "Branch and Bound Algorithm for The Flowshop with Multiple Processors", *European Journal of Operational Research*, pp88-99, 1991.

[11] Brown, J. R., et al., "Priority class scheduling : product scheduling for multi-objective environment", *Production Planning and Control,* Vol. 8, No. 8, pp762-770, 1997.

[12] Bullnheimer, B., Hartl, R. F., and Strauss, C., "An improved ant system algorithm for the vehicle routing problem", *Ann. Oper. Res.* 89, 1999.

[13] Cheng, R. and Gen, M., "Parallel machine scheduling problems using mimetic algorithms", *Computer and Industrial Engineering*, Vol. 33, No. 3-4, pp761-764, 1997.

[14]Cheng, R., Gen, M. and Y. Tsujimura, "A tutorial survey of job-shop scheduling problems using genetic algorithms-1 representation", *Computers Ind. Engng.*, Vol. 30, No. 4, pp983-997, 1996.

[15]Cheng, T. C. E. and Diamond, J. E., "Scheduling Two Job Classes on Parallel Machines", *IIE Transactions*, Vol. 27, pp689-693, 1995.

[16]Colorni, A., Dorigo, M., Maniezzo, V. and Trubian, M., "Ant System for Job-Shop Scheduling" *Belgian Journal of Operations Research, Statistics and Computer Science*, Vol. 34, No. 1, pp39-53, 1994.

[17]Daniels, R. L., "Incorporating performance information into multi-objective scheduling", *European Journal of Operational Research,* Vol. 77, pp272-286, 1994.

[18]Di Caro, G. and Dorigo, M., "AntNet: distributed stigmergetic control for communications networks", *J. Artif. Intell. Res*, 9, pp317–365, 1998.

[19]Dorigo, M. and Gambardella, L. M., "Ant colony system: a cooperative learning approach to the traveling salesman problem" *IEEE Trans. Evolut. Comput*, 1, pp53–66, 1997.

[20]Dorigo, M., Maniezzo, V. and Colorni, A., "Ant system: optimization by a colony of cooperating agents" *Systems, Man and Cybernetics, Part B, IEEE Transactions*, Vol. 26, Issue. 1, pp29-41, Feb. 1996.

[21]Fang, H. L., "Genetic Algorithm in timetabling and scheduling", Ph. D. dissertation, Department of Artificial Intelligent, University of Edinburgh, 1994.

[22]Forgaty, T. C., "Varying the probability of mutation in the genetic algorithm", *Proceedings of the 3rd International Conference on Genetic algorithms*, pp104-109, 1989.

[23]Freisleben, B and Merz, P., "A genetic local search algorithm for solving symmetric and asymmetric traveling salesman problems" *Proceedings of the IEEE International Conference on Evolutionary Computation (ICEC'96)*, IEEE Press, Piscataway, USA, pp616–621, 1996.

[24]Gambardella, L. M. and Dorigo, M., "HAS-SOP: hybrid ant system for the sequential ordering problem" *Technical Report IDSIA*, IDSIA, Lugano, Switzerland, pp11-97, 1997.

[25]Gambardella,L.M., Taillard, É. D., and Agazzi, G., "MACS-VRPTW: a multiple ant colony system for vehicle routing problems with time windows" D. Corne, M. Dorigo, F. Glover (Eds.), *New Ideas in Optimization*, McGraw-Hill, London, pp63–76, 1999.

[26] Garey M. R. and Johnson D. S., *Computers and Intractability: A Guide to the Theory of NP-completeness*, Freeman, San Francisco, 1979.

[27] Goldberg, D. E., *Genetic Algorithm in Search, Optimization and Machine Learning*, Addision-Wesley Publishing Co., 1989.

[28] Guinet, A., "Scheduling independent jobs on uniform parallel machines to minimize tardiness criteria" *Journal of Intelligent Manufacturing*, Vol. 6, No. 2, pp95-103, 1995.

[29] Gupta, J. N. D., "Two-Stage, Hybrid Flowshop Scheduling Problem" *J. Oper. Res. Soc.*, Vol. 39, pp359-364, 1988.

[30] Gürsel, A. S., Pico, F. and Santiago, A., "Identical machine scheduling to minimize the number of tardy jobs when lot-splitting is allowed" *Computer and Industrial Engineering*, Vol. 33, No. 1-2, pp 277-280, 1997.

[31] Hundal, T. S. and Rajgopal, J., "An Extension of Palmer's Heuristic for the Flow-shop Scheduling Problem" *International Journal of Production Research*, Vol. 26, No. 6, pp1119-1124, 1988.

[32] Ishibuchi, H. and T. Murata, "A multi-objective genetic local search algorithm and its application to flowshop scheduling", *IEEE Transactions on System, Man, and Cybernetics-Part C : Application and Review,* Vol. 28, No. 3, pp392-403, August 1998.

[33] Itoh, K. D. Huang and T. Enkawa, "Twofold look-ahead search for multi-criterion job shop scheduling", *Int. J. Prod. Res.,* Vol. 31, No. 9, pp2215-2234, 1993.

[34] Jacques Carlier and Ismaïl Rebaï, "Two Branch and Bound Algorithms for the Permutation Flow Shop Problem", *European Journal of Operational Research*, 90, pp238-251, 1996.

[35] Kim, C. O., et al., "Integration of inductive learning and neural networks for multi-objective FMS scheduling", *Int. J. Prod. Res.,* Vol. 36, No. 9, pp2497-2509, 1998.

[36] Kim, G. H. and C. S. G. Lee, "An evolutionary approach to the job-shop scheduling problem", *Proceedings IEEE International Conference on Robotics and Automation,* Vol. 1, pp501-506, 1994.

[37] Maniezzo, V., "Exact and approximate nondeterministic tree-search procedures for the quadratic assignment problem", *INFORMS J. on Computing*, 11, pp358–369, 1999.

[38] Mellor, P., "A review of job shop scheduling ", *Operational Research Quarterly*, Vol. 17, No. 2, pp161-170, 1966.

[39]Michalewicz, Z., *Genetic Algorithm + Data Structures = Evoluation Programs*, Springer-Verg Berlin Heidelberg, 1994.

[40]Michel, R. and Middendorf, M., "An island based ant system with lookahead for the shortest common supersequence problem" in: A. E. Eiben, T. Bäck, M. Schoenauer, H.-P. Schwefel(Eds.), *Proceedings of the Fifth International Conference on Parallel Problem Solving from Nature, Lecture Notes in Computer Science*, Vol. 1498, Springer, Berlin, pp692–708, 1998.

[41]Min, H. S., Y. Yih and C. O. Kim, "A competitive neural network approach to multi-objective FMS scheduling", *Int. J. Prod. Res.,* Vol. 36, No. 7, pp1749-1765, 1998.

[42]Min, L. and Cheng, W., "A genetic algorithm for minimizing the makespan in the case of scheduling identical machines" *Artificial Intelligence in Engineering*, Vol. 13, pp399-403, 1999.

[43]Murata, T. and H. Ishibuchi, "Performance evaluation of genetic algorithms for flowshop scheduling problems", *Proceddings of the First IEEE Conferernce on Evolutionary Computation*, Vol. 2, pp812-817, 1994.

[44]Murata, T., H. Ishibuchi and H. Tanaka, H., "Multi-objective genetic algorithm and its applications to flowshop scheduling", *Computers and Industrial Engineering,* Vol. 30, No. 4, pp957-968, 1996.

[45]Nawaz, M., "A Heuristic Algorithm for the 'M'-Machine, 'N' Job Flow-Shop Sequencing Problem" *Management Science*, Vol. 11, No. 1, pp91-95, 1983.

[46]Neppalli, V. R., C. L. Chen and J. Gupta, "Genetic algorithms for two-stage bicriteria flowshop problem", *European Journal of Operational Research*, Vol. 95, pp356-373, 1996.

[47]Nowicki E. and Czeslaw S., "The Flow Shop with Parallel Machines：A Tabu Search Approach" *European Journal of Operational Research*, Vol. 106, pp226-253, 1998.

[48]Petty, C. B., M. R. Leuze and J. J. Grefenstette, "A parallel genetic algorithm", *Proceedings of the Second International Conference on Genetic Algorithm,* pp155-161, 1987.

[49]Piersma, N. and Van Dijk, W., "A local search heuristic for unrelated parallel machine scheduling with efficient neighborhood search", *Mathematics and Computer Modeling*, Vol. 24, No. 9, pp11-19, 1996.

[50]Pinedo, M., *Scheduling : theory, algorithms, and systems,* pp118-141, Prentice Hall, Inc., 1995.

[51] Santos, D. L., Hunsucker J. L. and Deal. D. E., "Flowmult: Permutation Sequences for Flow Shops with Multiple Processors", *J. Inform. Optim. Sci.*, 16, pp351-366, 1995.

[52] Satty, T. L., *The Analytic Hierarchical Process*, McGraw-Hill, Inc., 1977.

[53] Serafini P., "Scheduling jobs on several machines with the job splitting property", *Operations Research*, Vol. 44, No. 4, pp617-628, 1996.

[54] Sivrikaya-Serifoðlu, F. and Ulusoy, G., "Parallel machine scheduling with earliness and tardiness penalties", *Computers and Operations Research*, Vol. 26, No. 8, pp 773-787, 1999.

[55] Sridhar, J., Rajendran, C., "Scheduling in A Cellular Manufacturing System: A Simulated Annealing Approach" *International Journal of Production Research*, Vol. 31, No. 12, pp2927-2945, 1993.

[56] Srivastava, B., "An effective heuristic for minimizing makespan on unrelated parallel machines", *Journal of the Operational Research Society*, Vol. 49, pp886-894, 1998.

[57] Stützle, T. and Dorigo, M., "ACO algorithms for the quadratic assignment problem", In D. Corne, M. Dorigo and F. Glover, editors, *New Ideas in Optimization*, Mc-Graw Hill, pp33–50, 1999.

[58] Stützle, T. and Hoos, H. H., "The MAX – MIN ant system and local search for the traveling salesman problem" in: T. Bäck, Z. Michalewicz, X. Yao (Eds.), *Proceedings of the IEEE International Conference on Evolutionary Computation (ICEC'97)*, IEEE Press, Piscataway, USA, pp309–314, 1997.

[59] Stützle, T., "An Ant Approach to the Flow Shop Problem", *In Proceedings of the 6 European Congress on Intelligent Techniques & Soft Computing (EUFIT'98)*, Vol. 3, Verlag Mainz, Aachen, pp 1560–1564, 1997.

[60] Suresh, V. and Chaudhuri, D., "Bicriteria scheduling problem for unrelated parallel machines" *Computer and Industrial Engineering*, Vol. 30, No. 1, pp77-82, 1996.

[61] Tamimi, S. A. and Rajan, V. N., "Reduction of total weighted tardiness on uniform machines with sequence dependent setups", *6th Industrial Engineering Research Conference Proceedings*, pp181-185, 1997.

[62] Wellman, M. A. and D. D. Gemmill, "A genetic algorithm approach to optimization of asynchronous automatic assembly systems", *International Journal of Flexible Manufacturing Systems,* Vol. 7,

pp27-46, 1995.

[63]Zwaan, S. V. D. and Marques, C., “Ant Colony Optimisation for Job Shop Scheduling” *Proceedings of the Third Workshop on Genetic Algorithms and Artificial Life (GAAL 99)*, 1999.