第二章 文獻探討

本章節主要先針對國內外對於訂單分配相關之研究，以瞭解目前訂單分配的現況，由於本研究針對基因演算法中對於限制式之處理進行探討，因此將對最佳化問題中於限制式處理之概念進行介紹，以及比較各種對於限制式之處理方式，並且介紹本研究中採用的幾種演算法：基因演算法、禁忌搜尋法、層級分析法。

2.1 訂單分配與訂單管理

Yangm and Sum (1994)將分配規則可分為兩大類：第一、以時間為基礎的規則(Time-based Rules)－如依據工件交期(Job due dates)、剩餘處理時間(Remaining processing time)、目前流程時間(Current shop times)等進行分配。第二、以成本為基礎的規則(Cost-based Rules)－如以工件價值(Job monetaryvalues)、剩餘處理時間(Remaining processing time)等為評量依據。

依據上述的概念，我們對於國內外之文獻進行探討，以瞭解國內外目前對於訂單分配方面的研究現況為何。

林慈傑(2002)針對多廠訂單分配問題進行研究，建立一個考慮到多間工廠在生產不同產品種類的訂單時，各工廠所需付出的製造成本、設置成本、運輸成本與因故未能如期完成的訂單交期延誤成本，且包含各工廠在規劃期間內其產能所能承受負荷限制的問題模式。研究除建立此訂單分配問題的混合整數規劃數學模式外，並提出特殊的類運輸問題(Quasi-Transportation Problem, QTP)模式以模式化此多廠訂單分配問題。研究以基因演算法(Gnentic Algorithm, GA)使用三種染色體編碼方式(矩陣型(matrix)、基變數型(basic-variable)、及擴展樹型(spanning trees))作為多廠訂單分配問題的求解方法，並針對不同編碼方式的特性作分析比較，進而尋找出滿足訂單交期且花費較少生產成本的多廠訂單分配結果。

李志勇(2002)針對多製造工廠在全面訂單管理的情況下，對於製造工廠的跨廠訂單分配模式進行研究，建立一個考慮多製造廠在接收大量訂單時，如何依據產品的市場銷售特性、訂單交期、工廠間的生產排程、訂單利潤及工廠產能負荷度等等限制，建立一多廠整合型生產指派與排程系統的決策模式。該研究利用AHP法則、斐氏理論(Petri Net)的概念，建立一訂單指派模式，再以基因遺傳演算法求算訂單指派至各廠的生產排程與各廠的產能平衡，以決定訂單的最佳配置，進而依據所指派的結果作為各製造工廠進行生產計畫與排程規劃的依據。

Chan等學者(2004)提出一種混合式基因演算法，針對多廠區供應鏈模型下之生產與配送問題。供應鏈問題通常牽涉到多目標決策(multi-criterion decision-making)，例如操作成本(operating cost)、服務等級(service level)與資源利用度(resources utilization)等等多個目標考量，為了將這多個目標組織起來，此研究提出以多層級分析法(Analytic Hierarchy Process)來達到此目的，並且提供決策者設定幾個目標間相互的權重關係，最後介紹基因演算法(Genetic Algorithm)來求解這個問題，將其與多層級分析法進行結合，提出一混合式的演算法來求解此問題。

劉珮伶(2004)提出一考慮產品配送下之多廠區訂單分配模式，以外、內兩個門檻值接受法，以兩階段求解的方式決定第一階段之多廠區訂單分配決策，並將訂單分配決策交與第二階段決定產品配送路徑，考量兩階段決策互相影響下，求解多廠訂單分配中的工廠生產成本，及包含生產成本、產品配送成本及配送距離成本之總成本，再將第二階段之結果回饋於第一階段進行決策，以反覆影響的方式同時降低生產成本及配送成本，得到總成本最小之多廠區訂單分配決策。

高敏純(2005)提出一訂單分配決策支援系統，其中分為二階段：(1)工廠之灰關聯排序：其針對影響訂單工廠排序之定性因素，以「客戶核廠」、「客戶喜好」、「工廠專精」與「工廠技能度」等作為排序因子；(2)訂單分配線性規劃模式：針對可量化因素並以最總成本最小為目標，將訂單作最適工廠分配。最後以一個案來作為演練，希望研究的成果將可提供成衣產業在全球運籌管理下的運作方

式、營運指標及訂單分配模型參考。

2.2 最佳化問題中對於限制式之處理

Michalewicz and Fogel 等學者在探討最佳化問題中，對於限制式之處理方式時，發現在大多數的現實問題中，大多都很難去設計一個運算子，使其皆產生可行解（feasible solution），而且當不可行解的適合度（fitness）高於可行解時，是該保留此解？修正它？或者直接捨棄它呢？這些都影響到演算法的設計策略，我們可以就下面幾個方向來探討：

- 去除不可行解（Rejecting infeasible solution）

  這是最常見的一種處理方式，將不可行解剔除，當然這也是最簡單的一種方式，不過這會有某些缺點發生，Michalewicz,Z.[1] 指出在多數的例子中，若可以跨過不可行解區，將可以更快找到最佳解，加速求解速率。

- 改良不可行解（Repairing infeasible solution）

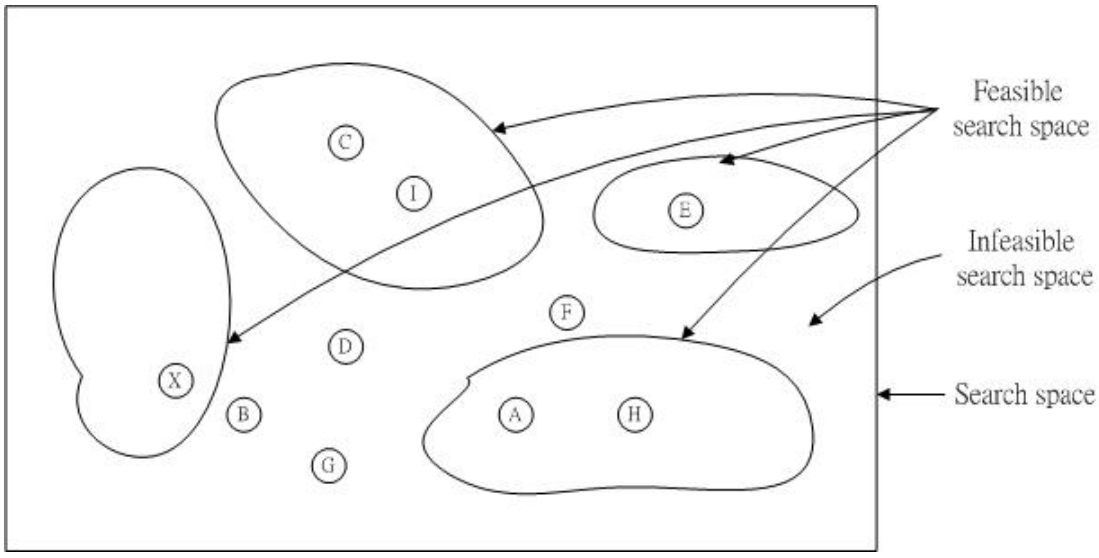

圖 2- 1 求解空間示意圖

圖 2-1 為可行解空間與不可行空間的示意圖，其中的多個點(A,B,C…)為各個解的個體(individual)，個體可能位於可行解空間，也可能位於不可行解空間，而 X 點為最佳解，我們由圖中可以發現，B 點雖位於不可行解空間中，

但是它距離最佳解卻是非常接近的，若是可以改善 B 點，則有很高的機率能夠直接找到最佳解。

改良不可行解就是根據這個概念而生，在群體（Population)中的個體，若為不可行解，則有可能使用區域搜尋（Local Search）找另一可行解取代。

- 使用懲罰函數（Methods based on penalty function）

此方式的主要概念是建立一個懲罰函數（penalty function)，去降低不可行解（infeasible solution）的權重，使得在求解過程中，此解比較不容易被選取，而懲罰函數的設計，則可分為動態與靜態的不同，可視問題需求而去設計。

- 使用解碼器（Methods based on decoders）

此方式的概念是，某些問題的解，我們可以建立特殊的編碼，使其計算過程中，不會產生不可行解，例如背包問題中，若是我們將其解的編碼形式以二進位表示，則其可轉為「取或不取」的問題，因此無論如何運算皆為可行解。

表 2- 1 限制式處理比較表

處理方式	在本研究中的缺失
刪除不可行解	不可行解未必是不好的解，可能僅需部分的修改即可
使用懲罰函數	懲罰函數的概念是降低不可行解被選取的機會，但在本研究中不可行解未必不接近好的解
使用解碼器	解碼器的設計方式，主要是利用解碼器將解的編碼轉換為一僅出現可行解的編碼，一般用於背包問題類型的選取問題，因此不適於本研究。

表 2-1 是各種對於不可行解的處理方式比較表，本研究擬採用「改良不可行解」之方式，原因在於，如上述中所提，不可行解可能離最佳解很近，改良此不可行解能加速求解速度與品質，Orvosh 和 Davis 等學者提出「百分之五法則」（5-percent rule)，表示在被取代的不可行解中的百分之五，可能就是最佳解。而其它的方式，如刪除不可行解（求解品質與效率較改善不可行解為差)、使用

懲罰函數（不可行解未必不會接近好的解）、使用解碼器（不適用於此類行問題）都有其不適宜之處，因此我們選擇「改良不可行解」的方式來探討。

在改善不可行解的討論中，最常見的方式就是找尋另一可行解來取代它，而如我們前面所提，不可行解可能很接近最佳解，因此若我們可以對此不可行解的周圍進行搜尋，則有機會找到最佳解，即便不是最佳解，也可以此取代原先之不可行解，此方式將有效改善求解的品質與效率。在下一節中，我們將選擇一適合的演算法，來對不可行解進行鄰近的搜尋。

2.3 演算法探討

本小節介紹幾種常見的演算法以進行比較，並選取一適合的演算法來對限制式之處理進行修補，以及幾種研究中使用到的演算法之介紹。

2.3.1 啟發式演算法比較

Glover and Laguna 對幾種常見啟發式演算法做了以下的比較：

表 2- 2 啟發式演算法比較表

啟發式演算法	**分類**
基因演算法(genetic algorithm)	A/S/P
模擬退火法(simulated annealing)	A/S/1
繖布搜尋法(scatter algorithm)	A/N/P
禁忌搜尋法(Tabu Search)	M/N/1

表 2- 3 名詞解釋表

代號	特徵代號	特徵說明
X	M	使用適度記憶（adaptive memory），紀錄過程中搜尋過的解。

	A	無記憶功能。
y	N	有系統化的方式搜尋鄰近解
	S	隨機產生鄰近解
z	1	以單一解進行搜尋
	P	以群體解進行搜尋

演算法特性說明：

基因演算法：在搜尋過程中，不紀錄過程中搜尋過的解，而產生鄰近解的方式是以運算子隨機產生，搜尋過程則是以群體為單位產生鄰近解。

模擬退火法：在搜尋過程中，不紀錄過程中搜尋過的解，而產生鄰近解的方式是以運算子隨機產生，不過在搜尋過程中是以單一個體為單位產生鄰近解。

撒布搜尋法：在搜尋過程中，不紀錄過程中搜尋過的解，其產生鄰近解的方式有其特殊的運算子產生，在搜尋過程中是以群體為產生鄰近解的單位。

禁忌搜尋法：在搜尋過程中，會紀錄過程中搜尋過的解，其產生鄰近解的方式有特殊的運算子產生，在搜尋過程中是以單一個體為單位產生鄰近解。

在上述幾種啟發式演算法的比較表中，我們發現禁忌搜尋法有幾個特點很適合我們這個問題：

1. 以單點進行搜尋，正好可針對不可行解進行鄰近搜尋，進行替代動作。

2. 進行鄰近搜尋，而非隨機搜尋。

基於以上幾點，我們可以發現禁忌搜尋法很適宜於我們的這個問題，而觀察禁忌搜尋法之演算法架構發現，禁忌搜尋法之結構簡單、搜尋快速，避免為了改善不可行解而拖慢整體速度，因此本研究採用禁忌搜尋法來搜尋不可行解周圍的鄰近解進行替代動作。

2.3.2 禁忌搜尋法介紹

禁忌搜尋法是由 Glover and Laguna 所提出一啟發式演算法，適於處理各種最佳化或排列組合之問題，其主要發展的概念是源自登山演算法(Hill-Climbing)之演算過程，以搜尋鄰近解中之最佳解為基礎，不斷反覆尋找最佳解，並且改善登山演算法易於陷入區域最佳解(Local optimal)的缺點，加入適度記憶(adaptive memory)的架構，利用禁忌名單(Tabu list)，紀錄搜尋過的點，避免搜尋僅限於部分區域中。流程架構參考圖 2-2:

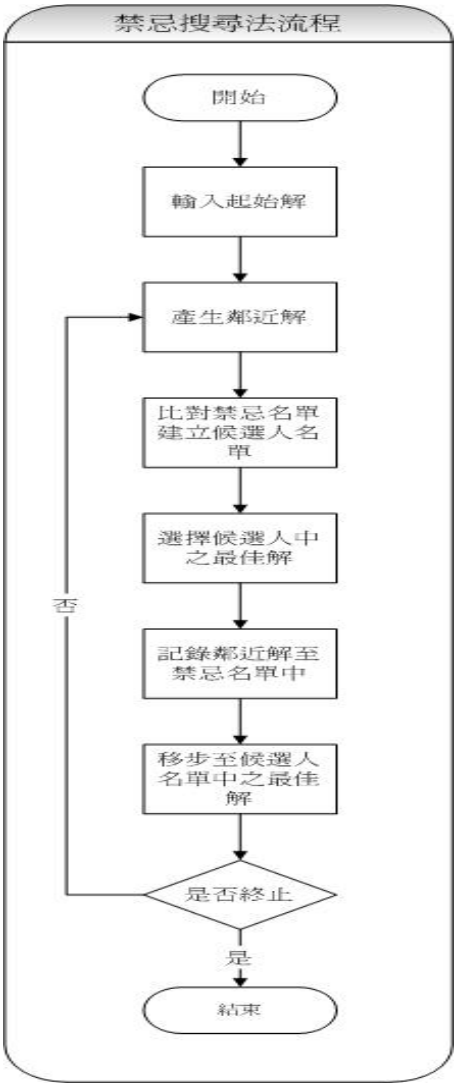

圖 2- 2 禁忌搜尋法流程圖

禁忌搜尋法的記憶策略分為短期記憶(long-term memory)與長期記憶(short-memory)，短期記憶主要是記錄搜尋過的點，以避免重覆的軌跡，而長期

記憶則是透過短期記憶，以尋找適合的方向找尋最佳解。

一般禁忌搜尋法的架構分為下列幾項：(一) 起始解(initial solution)、(二) 移步(move)、(三) 禁忌名單(tabu list)、(四)候選者名單(candidate list)、(五) 破禁原則(aspiration criterion)、(六) 停止條件(stopping criterion)

(一)起始解(initial solution)：

起始解為開始搜尋之點，也可說是禁忌搜尋法之輸入值，一般的作法是採用亂數產生，也有部分研究使用啟發式演算法求出之解當作起始值。

(二)移步(move)：

移步也就是產生鄰近解 (neighbor solution) 的方式，一般常見的有交換(Swap)、插入(Insert)、加入/消去 (Add/drop) 和增加/減少 (Increase/decrease) 等方式。

(三)禁忌名單(tabu list)：

此為用來記錄搜尋過程中之軌跡，名單的大小有各種的研究，名單長雖然可記錄的點較多，但也相對會拖慢搜尋的過程，一般採用 Glover 提出的魔術數字 (magic number) 7 為禁忌名單大小。

(四)候選者名單(candidate list)：

所謂的候選者名單指的是，扣除禁忌名單與破禁原則後的鄰近解，由此名單中選出該移步的點。

(五)破禁原則(aspiration criterion)：

此為解除禁忌名單之條件，在某些問題中，若達到這些條件，釋放部分禁忌名單中的點，可改善求解的品質。

(六)停止條件(stopping criterion)：

指的就是演算法的停止迴圈數，一般分為兩種：最大迴圈數(max iteration)、最大解無改善回圈數 (max iteration with no improvement)，前者指的是搜尋的最多次數，後者是指在此圈數下，若解仍無改善，則停止。

2.3.3 基因演算法介紹

基因演算法(Genetic Algorithm, GA)由 Holland 於 1975 年提出，主要概念源自達爾文的進化論，利用一個群體（population）進行演化，群體中包含多個染色體，利用複製（reproduction）、交配（crossover）、突變（mutation）等等運算逐漸接近最佳解，已被應用至最佳化、工程、排序、電腦等多個領域上。

基因演算法以群體為架構，每個群體中包含多個染色體，而每個染色體包含多個基因，每個基因代表一種屬性，架構可參考圖 2-3，對應至最佳化問題中則，群體代表一組解，染色體代表一個解，基因代表解上的某一個變數，利用此方式即可解最佳化之問題。

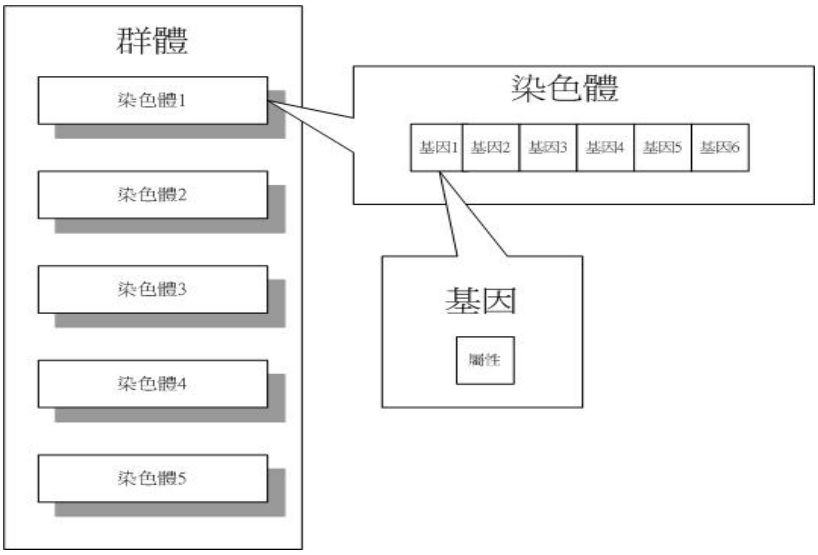

圖 2- 3 染色體架構圖

基因演算法不同於一般的演算法在於，它並非採用單點搜尋，而是多點搜尋，當解空間很大時，僅在起始解附近開始搜尋最佳解，效率較差，這也是他逐漸被應用至多個領域的原因。基因演算法的架構，主要包括下列幾個部分，(一)起始解、(二）編碼、(三）適合度、(四）複製、(五）交配、(六）突變、(七）停止條件，其流程請參考圖 2-4

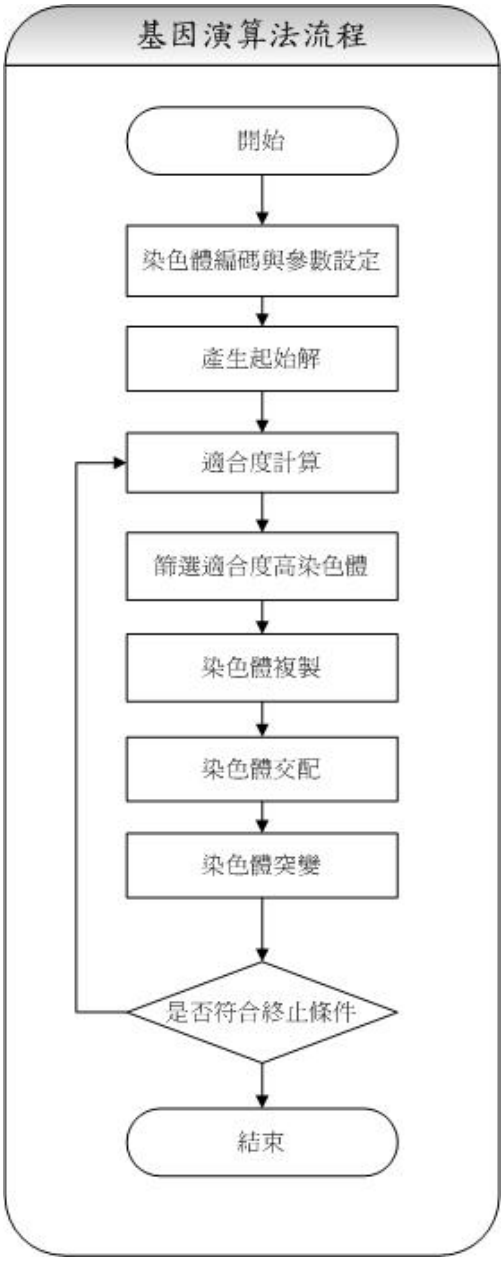

圖 2- 4 基因演算法流程圖

（一）起始解:

產生起始解時，會先設定群體的大小，群體的大小影響求解的速度，大的群體雖然涵蓋的範圍較廣，但是所耗費的時間也較多，此為需要考量之處。

（二）編碼:

染色體的編碼，必須依照問題的解的類型來決定，一般常見的編碼有為二元數值(Binary)、實數(Real)、整數(Integer)、及符號(Symbol)，必須依照問題特性來選擇。

（三）適合度:

適合度是基因演算法中用來衡量染色體優劣的方式,而染色體的優劣則會影響其存活的機率，適合度越高者，存活機率越高，而適合度函數的設計，則需依照問題特性決定。

（四）複製:

複製指的就是從群體中挑選適合度高的染色體保留至下一代,一般常見的方法有輪盤法（Roulette Wheel Selection），就是在一輪盤上，面積代表被選中的機率,因此越高的適合度有越大的面積,也表示越容易被選中至下一代。

（五）交配:

當完成複製階段後,群體中的染色體則有機會進行交配,但並非每條染色體都會進行交配，而是會根據交配率(crossover rate)，選擇偶數條染色體進行交配，而交配的方式有單點交配、雙點交配、均勻交配、算術型交配、及矩陣型交配法，而本研究之染色體編碼為矩陣型編碼，因此我們介紹由Vignaux＆Michalewicz提出的矩陣型交配法：

假設染色體 X 為一矩陣型編碼染色體，而 X 展開後為

$$X=\begin{bmatrix} x_{11} & x_{12} & x_{13} & \Lambda & x_{1n} \\ x_{21} & x_{22} & & & \mathrm{M} \\ x_{31} & & \mathrm{O} & & \mathrm{M} \\ \mathrm{M} & & & \mathrm{O} & \mathrm{M} \\ x_{m1} & \Lambda & \Lambda & \Lambda & x_{mn} \end{bmatrix}$$

Xmn ： 基因值

步驟 1:染色體交配

若欲交配之兩染色體分別為 x1 和 x2,則在交配過程中會產生兩矩陣 y1, y2

$$y1=\mathrm{int}(\frac{x1+x2}{2})$$

$$y2=(x1+x2)\bmod 2$$

步驟 2:產生新的染色體

將 y2 隨機拆成兩個矩陣，並使得 y2=y21+y22

則，新產生的染色體 x1’ 和 x2’ 為

x1’ ＝y1+y21

x2’ ＝y2+y22

交配結束。

（六）突變:

若交配時僅有交配運算，則易陷入局部最佳解，因此為了避免這種情形，加入了突變運算，以增加搜尋的廣度。演化過程中，依照突變率決定是否突變，通常突變率都小於0.1，而不同的編碼採用不同的突變法，一般常見的突變有交換式(interchange)突變、取代式(displacement)突變法。交換式突變的作法是任選兩個基因，然後對其進行交換；取代式突變法的作法則是任選一個基因，然後取一合理數值進行取代。本研究採用後者，也就是取代式突變，前者較適合排序問題。

（七）停止條件:

常見的停止條件有下列幾種

(1)設定演化的世代數

(2)設定電腦運算的時間

(3)若經過了x世代，而最佳的適合改善不到y%

2.3.4 層級分析法介紹

層級分析法（Analytic Hierarchy Process, AHP）(Saaty,1980)於1971年由Thomas L. Saaty學者提出，主要應用在不確定的情況下具有多個評估準則的決策問題上。AHP架構，主要就是將問題予以系統化分析，由不同的層面給予層級分解，並透過量化的判斷，提供給決策者

AHP的建立層級結構主要包含最終目標、決策準則、可行方案，接著將同一層級的元素兩兩比對其相對重要性，最後並結算出整體的重要性，最佳可行方案便由此來判斷。其主要包括以下三個階段(Saaty,1980; 鄧振源，曾國雄,1989)：

第一階段：建立層級結構處理複雜問題時，利用建立層級結構將問題予以分解，可以有效的描述問題的形式，以及各要素之間、高層要素對於低層要素的影響程度,進而可以更有效的達成工作。AHP的層級結構如圖2-5。

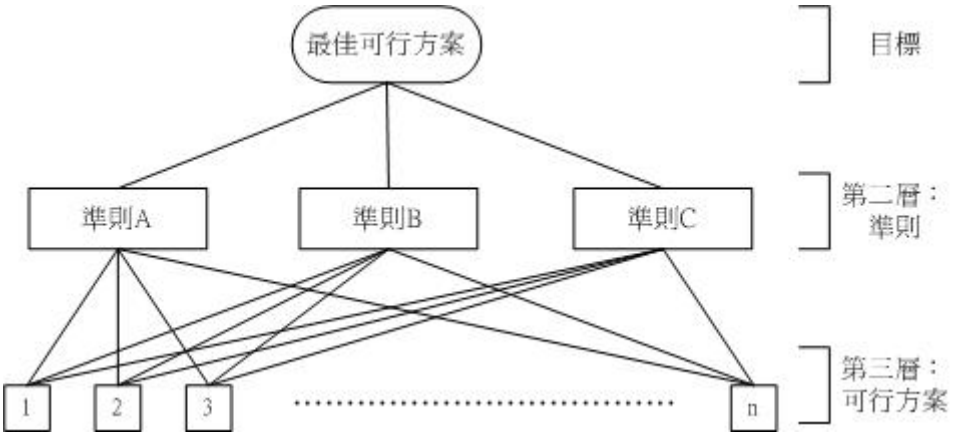

圖 2-5 AHP 層級架構圖

第二階段：各層級要素間權重的計算。此一階段可區分為三個步驟：

（1）建立成偶比對矩陣

某一層級的要素，以上一層級某要素作為評估基準下，進行要素間的成對比較。將n 個要素的比較結果，用數值置於矩陣A的上半部，下半部則為它的倒數。例如：此為一3x3成偶比對矩陣

$$X = \begin{bmatrix} 1 & 2 & 8 \\ 1/2 & 1 & 4 \\ 1/8 & 1/4 & 1 \end{bmatrix}$$

（2）計算特徵值及特徵向量

獲得成偶比對矩陣後，即可計算各層級要素的權重。並使用數值分析中常使用的特徵值與特徵向量解法，求出特徵向量或稱優勢向量（Priority vector）。

（3） 一致性的檢定

決策者在做成對比較時，未必填答時能夠前後一致。因此需進行一致性的檢定，作成一致性指標（Consistency Index, CI），以檢查從決策者所獲得的成對比較矩陣，是否符合一致性。

第三階段：整體層級權重的計算。

各層級要素間的權重計算後，再進行整體層級權重的計算，即可得到最佳的決策。