第二章 文獻回顧

2.1 最佳化設計簡介

傳統上，一項工程設計的進行係由工程師遵循規範與其他相關規定，依其專業素養與經驗判斷作出符合該工程功能要求之設計。如果一個設計符合所有應滿足的限制條件，即為一個可行設計。而絕大多數狀況下一項工程均有多種甚至無限多種可行設計，這些眾多可行設計之間各有異同，亦有優劣之分，而所謂的最佳化設計就是利用有系統的方法，在眾多的可行設計中尋求其最佳者。

結構設計的最佳化概念，最早是由 Michell [1]在 1904 年時所提出的桁架理論中呈現；近年來隨著電腦的快速發展與有限元素分析能力的成熟，各種 CAD、CAE 的輔助，讓結構最佳化設計的進展更加有效率，應用層面也愈廣。結構最佳化設計一般而言可分為結構的尺寸最佳化設計(Size optimization design)、形狀最佳化設計(Shape optimization design)和拓樸最佳化設計(Topology optimization design)這三個部分，隨著技術之發展，現亦有將拓樸最佳化設計和形狀最佳化設計相互結合等之相關研究。而這三個領域裡，尺寸最佳化設計首先被發展，在 1960 年由 Schmit [2]所提出；顧名思義即是對一個已知外型的結構尺寸作最佳化設計，譬如透過改變其寬度、厚度、截面積等等，在滿足最大應力、變形、位移等限制條件下，以達成結構具最小重量之設計目標。

幾年來，最佳化理論已有長足的發展，最常用的包括有多種數學規劃法，如循序線性規劃法(SLP)、循序二次規劃法(SQP)、整數規劃法(IP)..等等。近年來亦有許多新的方法如遺傳演算法(GA)、模擬退火法(SA)、螞蟻演算法(Ant-algorithm)...等等都可用來求解最佳化問題。這些方法各有其優缺點，使用者應選擇最適合的方法求解以提昇效率。

2.2 桁架斷面尺寸最佳化設計

對於一個桁架結構斷面尺寸的最佳化設計而言，其目標函數就是求桁架總重的最小值，其最主要的限制條件就是：桿件所受的應力需在容許壓、拉應力的範圍之內，以及節點位移需在容許位移之內。對於一個 n 根桿件有 m 個自由度的桁架而言，其目標函數可表成如(2-1)所示，假設其所有桿件及所有節點均須滿足上述條件，則限制可表示成(2-2)、(2-3)式：

目標函數：

$$\min \quad f(A) = \sum_{i=1}^{n} \rho_i A_i L_i \tag{2-1}$$

限制函數：

$$\sigma_i \le \sigma_{allow} \quad i = 1 \ \text{to} \ n \tag{2-2}$$

$$\delta_j \le \delta_{allow} \quad j = 1 \ \text{to} \ m \tag{2-3}$$

其中:

ρ_i ：第 i 根桿件材料密度。

A_i ：第 i 根桿件斷面積。

L_i ：第 i 根桿件長度。

σ_i ：第 i 根桿件所受應力。

σ_{allow}：桿件之容許壓、拉應力。

δ_j ：第 j 個自由度所受位移。

δ_{allow}：自由度之容許位移。

而傳統的數學最佳化或遺傳演算法，對於桁架斷面最佳化設計就是針對式(2-1～3)，將目標函數與限制函數做結合，建立成目標函數，再利用 2.1 節所提到的各種最佳化方法進行搜尋求得最佳解。

2.3 滿載應力法(Fully Stressed Design，FSD)

滿載應力設計(Fully Stress Design，FSD)的觀念是基於簡單的應力比逼近，使構件的應力儘量趨近於本身所能承受之容許應力，使構件均能發揮其最大作用以達最佳化設計目標，由於其方法的簡易性，因此廣泛的被應用在土木、機械、航太等領域。

其應用在桁架斷面最佳設計上，由於滿載應力設計出來的斷面只能滿足桁架的應力限制要求，對於容許位移並不能同時滿足，因此FSD方法只適用於無位移限制之桁架結構問題。基於此缺點 S.N.Patnaik(1986)[3]提出 FUD(Fully Utilized Design)方法，針對FSD不能滿足位移限制的缺點做改善，其方法分兩步驟：

(1)先行FSD獲得滿足應力限制之斷面。
(2)依照最大位移需符合容許位移而等倍放大各斷面而成FUD設計。

由於FUD為等倍放大設計，因此最終結果雖然能滿足位移限制，但是卻過於保守。

其後S.N. Patnaik(1997)[4]更進一步提出FMUD(Modified fully utilized design)，其方法為在每次迭代過程中，利用力法觀念導出桁架結構要同時滿足應力限制與位移限制的條件，使斷面尺寸在迭代過程中，在滿足應力限制下且各節點位移也可以逐漸滿足位移限制。

雖然FSD方法已廣泛被接受且應用在各領域，但是在設計重量上的最佳卻缺少明確的數學證明，因此 Surya and Patnaik(1998)[5]則嘗試利用拉格朗函數(Lagrangian function)，以數學的方法來說明FSD在重量上的最佳化原理。

Gil, Lluis，Andreu and Antoni(2001)[6]對數個平面桁架做形狀(Shape)與斷面尺寸的最佳化設計，其中桿件面積尺寸是以FSD方法所設計，而形狀則是以共軛梯度法去搜尋最佳的節點配置。

2.4 遺傳演算法(Genetic Algorithm method)

遺傳演算法係由 John Holland 於 1975 年首度發表。經過了多年的發展，遺傳演算法被証明為一有效的最佳化的搜尋方法。在最近幾年，許多學者投入這個領域繼續對演化式計算做更深一層的探索。由達爾文進化論的觀點，物種靠不斷的演化而產生最適合生存的物種。遺傳演算法即是由此一論點出發，模擬自然界的演化方式，對既定問題求最佳解。許多的實驗証明，遺傳演算法係一兼重效率與效能的搜尋方法，且可被廣泛的應用在許多的問題上。

其根據達而文的「物競天擇，適者生存，不適者淘汰」的理論，即一個族群為求適應生活環境不致被淘汰。在遺傳演化的過程中如圖 2.1 所示(亦即基因複製、交配、突變)中，較劣的個體隨著演化的進行而逐漸被淘汰，如此在演化達許多世代後之子個體均為較佳個體。基於這個理念，可以將遺傳演算法應用於各種最佳化問題。而近年來應用在桁架結構斷面最佳化設計已非常廣泛與成熟，例如：

Adeli and Cheng(1993)[7]指出梯度法在搜尋最佳解時的缺點，亦即當有多個局部最佳解存在時，搜尋的結果與搜尋的起點選擇有關，而且不易尋得整體最佳解。因此提出以遺傳演算法作結構最佳化設計。此外 Adeli and Cheng(1993)[8，9]也應用遺傳演算法作大型結構最佳化設計。

Adeli and Kumar(1995)[10]提出應用平行運算來解決遺傳演算法大量的重複運算，利用區域網路進行平行遺傳演算法的實做，並設計一大型的空間鋼結構。結果指出遺傳演算法對於平行化計算有高度的適用性。

Rajan(1995)[11]則更進一步應用遺傳演算法於桁架結構幾何形狀最佳化問題。

Sarma KC and Adeli(2000)[12]指出傳統的最佳化演算法其限制往往是滿足一明確(crisp)的容忍值，而實際工程實務上，其限制的估算往往是包括很多不精確與近似的來源。因此當設計需要很準確的滿足限制條件時，將有可能錯過全域的最佳解，所以提出利用模糊理論(Fuzzy)來改善其限制條件的估算，並提出利用 Fuzzy GA 可以降低最佳重量與減少電腦計算的時間。

Nicholas Ali, Kamran Behdinan and Zouheir Fawaz(2003)[13]探討遺傳演算法與有限元素軟體(ANSYS)結合的適用性與可行性，並對多個數值桁架案例做斷面尺寸與形狀的最佳化設計，並提出此方法富有彈性且適用於各種結構。在結論也提出遺傳演算法雖然可以成功地收斂在全域最佳解空間，但是卻需要花費很多電腦計算時間來完成。

總言之，遺傳演算法相對於傳統數學方法有以下優點：

1. 遺傳演算法屬於多點的搜尋，非一般的單點搜尋，其搜尋是具有高度平行性的，其一次處理一組親代，也就是處理很多個設計點，傳統方法一次處理一個設計點，容易陷入局部最佳解，遺傳演算法採取群的搜尋方式，可以有充分的機會接觸到每一個極點，使其解具有全域性。

2. 直接作用在參數的編碼字串上，而非原始參數本身。其他方法需要大量輔助工具，如需要一階函數、二階函數。若函數式不易微分，或是不易獲得其他輔助工具，往往限制該方法適用範圍。遺傳演算法有別於其他方法直接處理設計變數，其間接處理設計變數，也就是編碼後的字串，不受限函數連續與否，導函數是否存在、容易與否，運算也簡單多了。

3. 遺傳演算法僅需要目標函數(Object function)，而不需要其他的資訊。一般傳統的最佳化方法往往要大量的表列數據。

4. 遺傳演算法的轉移規則(transition rule)是隨機性的(probabilistic)，而非決定性的(deterministic)，因而較能符合各種不同類型的最佳化問題。

而遺傳演算法的缺點，主要在於當我們對所要解決的問題不了解時，所進行的世代演化次數會有過多或過少的現象。過多，則浪費太多時間在進行重複的的迭代運算；過少，則演算法太早收斂，以至尚未到達真正的近似最佳解。同時，染色體適應度函數的設計，也是影響演算法對問題求解的重要因素之一。