支援多重服務品質考量的教育雲服務選擇使用基因演算法

唐東暘　　朱正忠　　黃建華
東海大學資訊工程學系
g99350038@go.thu.edu.tw

摘要

「教育雲」的概念是利用雲端服務去支援教育與學習，主要是以服務導向架構的方式提供服務給使用者使用，在這個架構下，軟體經由網路提供功能性的服務給其他軟體使用，為了實現更多包含許多個別服務的複雜工作或商業程序，我們必須使用服務組合(service composition)的概念去幫助我們完成這個工作，但是如何在眾多功能相近的服務中挑選出最適當的服務組合以維持良好的服務品質（QoS, Quality of Service）則是一個挑戰，要達到良好的服務品質必須考慮包含花費(cost)、回應時間(Response time)等多種因素，因此基於 QoS 的服務組合選擇問題是一個 NP-Hard 問題。

本研究探討如何解決服務選擇優化的問題，並提出一個基於基因演算法(GA)的方法，針對多種服務類型的 QoS 特性，藉由基因不斷的演化過程，找出較優的服務組合，以確保服務品質。

關鍵詞：服務品質、組合優化、基因演算法

Abstract

The concept of education cloud is making use of cloud service to support education and learning. Providing services based on service-oriented architecture in recent years. The cloud application can be implemented by service composition in this architecture. An important challenge is how to select a service for each task involved in a composite service such that the overall QoS for the composite service is optimal. This includes customer focused elements such as response time, cost, reliability and availability. QoS-based service selection is a combination problem and is a kind of NP-Hard problems.

This study investigated how to optimize the selection problems and propose a genetic algorithm-based method which solving the problem by considering many QoS attributes in the gene. By the evolution process, the gene will be more suitable solution for services to ensure service quality.

Keywords:Quality of Service；　Service Selection；　Genetic Algorithm

1. 介紹

目前雲端運算偏向以服務導向(Service oriented)架構的方式提供雲端服務給使用者使用，而一套完整的雲端服務系統各個元件可能是由多個分散的雲端服務所構成，這些原本分散的服務藉由介面的溝通與綁定組合成了一個新的服務，如教育雲可能有郵件服務、課程服務、儲存服務等多個服務組成。

同一功能類型的服務可能有多個選擇並且彼此可以互相取代，在眾多服務中挑選符合需求的服務可以根據服務品質（QoS）的屬性去選擇，有些人可能要的是價錢最便宜的服務，有些人要的是最快的服務，有些則是要介於兩者之間的，因此 QoS 的屬性必須根據使用者的非功能性需求去制定服務指標，服務提供商可根據這些 QoS 屬性制訂服務等級協議(Service Level Agreement, SLA)，保證服務的品質可以滿足使用者的需求。

如何在眾多服務組合間挑選出符合 QoS 服務品質的組合是一個組合優化的 NP-Hard 問題[1]，必須還要考慮各種 QoS 限制條件，本論文提出一個基於基因演算法的方法，藉由基因不斷演化的過程，淘汰不好的基因留下好的基因組合，再進行交配、突變等過程，不斷的重複直到找到最好的選擇。

2. 相關研究

2.1 服務組合

雲端服務的一個吸引人的特性是現有的服務可以整合在一起，創建新的價值服務以滿足使用者的需求，整合建立新的服務的過程稱為服務組合[13]。服務組合的用意是取得現有的服務，並且結合這些服務形成新的服務能力，也就是將現有的服務加以組合，可以解決更複雜問題的服務這也讓服務可以應用在企業流程。以流程為基礎的網路服務組合，是一種新興的服務方式，可運用在組織內，或跨組織的自動化企業流程方法。

服務組合對於系統開發來說，是很重要的變革。傳統的系統開發是從無到有,必須依據系統開發流程，從需求訪談、系統分析與設計開始，一個元件一個功能辛苦地開發，而現今的方式是藉由現有服務的搜尋，配合特殊需求功能的服務開發，共同組合成一個新的系統。這樣的開發方式不僅可以減少系統開發的時間，並且可以增加網路服務的可再利用性。

依據使用者的需求,產生和執行組合計劃的能力,這樣的系統所提供的價值,明顯地超過個別網路服務所提供的價值，因此網路服務組合的必要性，在於它不僅可以將企業的流程，依據新的需求自動化產生，並且能快速地提供服務，一個服務組合的程序流程如圖 1 所示，其中的規範定義了結構化的活動來管理整個流程以及基本的活動，包括本身與服務互動的過程。

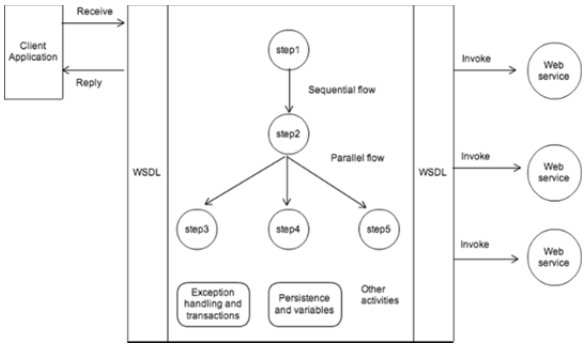

圖 1　BPEL-WS 服務程序流程[11]

大多數基於 Web 的雲端服務應用必須透過服務組合來建立，在開發雲端服務的應用程式的時候，考量 QoS 的服務組合是必須的，Jager[10]在他的論文中定義了 QoS 的服務組合主要的步驟

雲端服務的組合與選擇是一個建立基於服務導向的應用程式必要的過程，導入 QoS 的雲端服務組合面臨了許多挑戰，例如：如何建立服務的 QoS 模型和設計 QoS 的即時監控(Monitoring)框架，此外，進一步的挑戰是如何選擇服務去組成完整的服務，並考慮最佳的 QoS 因子，後者面臨的挑戰是所謂的基於 QoS 的服務選擇問題，在現實中，基於 QoS 的服務選擇通常不是一個單機操作，選擇應該考慮服務之間的倚賴性和衝突性等種種限制條件以保證其正確性，Zeng[5]在研究中提出了使用整數線性規劃的方法去解決這個問題，Yu 和 Lin[12]則提出了基於動態規劃的演算法去解決這個問題，也有學者用基因演算法(GA)[14][15]去克服這個問題。

2.2 基因演算法

基因演算法最原始的概念源自於 1859 年達爾文(Charles Darwin)在物種源起(The Origin of Speices)一書中對於演化的理論的「物競天擇，適者生存」觀念，而在 1975 年美國密西根大學 J. H. Holland 與他的同事及學生發表的研究[6]中首次被提出，其精神是以模擬生物演化過程中擇優汰劣的自然法則，保留母代的優良特性，一代一代找出適應性最強的子代，從而得出最佳解。

基因演算法適合用來解決全域最佳化問題(Global Optimization Problem)，例如時間表排程的問題等，步驟如下：

STEP 1　產生初代族群(Population)
STEP 2　針對初代族群的每一個體計算其適應性
STEP 3　根據交配(Crossover)或稱交叉操作產生下一代
STEP 4　根據突變(Mutation)機率產生突變的個體
STEP 5　計算此世代每一個體的適應性
STEP 6　測試是否滿足終止條件，若滿足則結束，否則回到 STEP 3

族群是染色體(Chromosome)的集合，而染色體是由基因所組成，針對問題設定一適性函數，用來評估各個體對於問題的適應性，也就是每個解對於該問題適合的程度，並且對染色體做編碼產生母群體，接著進行複製、交配與突變等操作產生新的族群，並對每個個體評估其適應性與測試是否達到終止條件，基因演算法的參數說明如下：

- 編碼(Encoding)：編碼型式依不同問題而有所不同，可依問題的類型與結構做選擇，常見的編法方式有二進位法、整數編碼、實數編碼等。
- 族群數(Population Size)：族群數會影響結果的產生，若設定一較小的值，雖然搜尋速度快，但會導致過早收斂，可能只得出區域最佳化的解而忽略了全域更好的解；反之，若設定較大可以在更大的物種多樣性中尋找解答，但會需要更多的計算時間。
- 交配機率(Crossover Probability)：介於 0 到 1 之間，用以控制不同基因組合的個體產生的速度，若太大會收斂過快，陷入區域最佳解；反之，則需要更長久的繁衍才會收斂。
- 突變機率(Mutation Probability)：介於 0 到 1 之間，但一般來說都設得很小，因為設太大與隨機選取無異，用以防止可能更優良的基因組合無法進入族群。
- 適應性函數(Fitness Function)：用來評估各個體的適應程度，透過適應程度的差異淘汰次佳個體，加速搜尋最佳解的依據。在不同的問題結構與特性下，可能需要尋找在某一範圍內之最大化或最小化當成解的適應性佳的趨勢。
- 終止條件(Termination Condition)：用以結束演化循環的控制條件，實作中，繁衍終止的條件很多，例如達到繁衍世代次數限制、繁衍的後代適應性達到事先訂定的標準或是繁衍出來的後代適應性皆收斂至同一水準而得到最佳解等。

而產生新的子代族群時有「整代取代」或是利用「菁英政策[8]」挑選，菁英政策基因演算法[9]是在產生子代時，挑選出適應性較佳的個體組成新的子代族群，加速搜尋最佳解。

本研究將探討如何使用基因演算法(GA)，以解決基於 QoS 的服務組合問題，基於 QoS 的 Web 服務組合優化，找出可行、可靠和高品質的複合 Web

服務，以完成客戶交付的目標。

3. 問題定義

多重服務品質的的服務選擇關係到組合後是否能保證服務品質的成功與否，選擇的結果直接決定了組合後的服務的 QoS，因此如何選擇適合的組合是非常重要的。

這個問題是典型的組合優化問題，本研究將問題定義為：

1. 一組服務組合內的抽象服務或工作的集合 $A\{A_1, A_2, \ldots, A_n\}$，n 為在這個組合內的抽象服務數量。
2. 在抽象服務 Ai 中擁有不同數量的具體服務 $C_i\{c_{i1}, c_{i2}, \ldots, c_{im}\}$,m 為在抽象服務 A_i 中擁有的具體服務數量。
3. 每個具體服務擁有自己的 QoS 指標數值，包含回應時間、花費價格、可靠性、可用性等。
4. 每個 QoS 指標的權重數值不同，
5. 組合後的服務可以根據這些 QoS 指標數值與 QoS 指標權重以及使用者的需求求出較優的服務組合

4. Qos 服務品質指標

第一件必須做的事是定義各種服務品質的屬性，雲端服務品質的概念主要是在網路、即時性、中介軟體等方面的屬性，在過去的研究中多注重如何定義服務品質的模型，目前還沒有任何標準技術去定義服務品質模型，不同的組織和研究者使用不同的服務品質分類去定義，三種常見的服務品質定義模型如表 1 所示。

表 1 各種 QoS 屬性的定義規範模型

UML QoS-Profile[3]	Ran's QoS Model[4]	Zeng's QoS Model[5]
throughput	throughput	throughput
latency	response time	response time
efficiency	cost	cost
availability	availability	availability
reliability	reliability	reliability
security	security	reputation

Qos 品質指標代表服務某方面非功能性的質量屬性，本研究參考表 1 各種 QoS 屬性的定義規範模型，在 UML 的 QoS profile 中缺少了花費(cost)這項指標，但是考慮到在雲端服務模式下 cost 是一項重要的指標，因此 cost 是必要的，而安全性(security)也是雲端的必須考慮的一個重要的因素，但是吞吐量(throughput)通常是特定的應用必須同時容納大量使用者的同時存取才必須考慮，綜合各點決定使用以下六個服務指標定義服務品質：

1. 費用(Cost, C)

費用反映服務使用的經濟成本，為提供商給出的使用該服務所需的價格，單位為元，負指標。

2. 反應時間(Response Time, RT)

反應時間反映服務的運作速度和需要使用者等待的時間，包括網路延遲和執行時間，透過呼叫服務的歷史資料所得到結果的平均時間計算，單位為秒，負指標。

3. 可靠性(Reliability, R)

可靠性反映服務的可靠程度，透過呼叫服務後在規定的時間內成功執行的次數佔總呼叫次數的比例計算，無單位，正指標。

4. 可用性(Availability, A)

可用性反映服務能夠被正常呼叫的機率，透過在過去一段時間內服務可用的時間佔總時間的比例計算，無單位，正指標。

5. 安全性(Security, S)

安全性反映服務的安全程度，透過安全級別（0～10）去衡量，無單位，正指標。

6. 企業名聲(Corporate Reputation, CR)

反映使用者對該服務提供商的滿意程度，透過用戶對服務的評分的平均值計算，無單位，正指標。

5. 多重服務考量的服務選擇

5.1 演算法流程

本研究中基因演算法應用於組合選擇優化的概念主要如下：將個別可能會使用服務類別以字串組合的方式表示成基因編碼，基因的族群表示一群候選服務組合的集合，適應性代表對於基因優劣量測後的屬性，用於之後的演化過程篩選，主要是以服務組合的 QoS 指標條件屬性計算適應性，以 QoS 為準則的服務選擇演算法過程如下：

1. 產生初始族群

以亂數的方式產生隨機編碼的基因初始族群，基因編碼為可能的服務組合索引字串形式，根據編碼的組合 QoS 屬性對每個基因計算適應性。

2. 選擇

根據基因適應性的分數對族群排序，把分數較低的基因移除，留下適應性分數較高的基因。

3. 交配

從經過選擇後適應性較高的族群中，隨機挑選任意兩個不同的父母基因，進行基因的交配，產生新的基因組合直到達到需要的族群數量為止。

4. 突變

將交配產生的新基因進行突變，根據設定的突變機率隨機找尋要突變的部分進行基因突變。

5. 疊代

反覆的進行以上選擇、交配、突變的動作產生新的子代，直到滿足收斂條件為止。

演算法流程如下所示：

Algorithm : A QoS-based genetic algorithm for the service selection problem

1. initialize population with random candidate solutions
2. evaluate the fitness of each individual based on the fitness function

defined in Equation 5.3 which contains QoS attributes

infeasible individuals

3. **while** termination condition is not true **do**
4. select fit individuals for reproduction
5. probabilistically apply the crossover operator to generate offspring
6. probabilistically apply the mutation operator to offspring
7. evaluate the fitness of each individual based on the fitness function defined in Equation 5.3
8. **end**

5.2 染色體編碼

為了將染色體表示成一組問題的解，首先必須將問題編碼為染色體基因組合,本研究中以整數陣列編碼代表染色體組合，以 0-based 陣列的索引代表服務種類的順序，內容的值代表該項服務在該服務種類的索引編號，根據此索引編碼規則對應服務組合。

以下列圖 2 的 chromosome 為例子：有三個服務類型(service_type0, service_type1, service_type_2)，分別擁有不同個數的服務,染色體編碼為:{1,3,0}，index 0 的數值對應到 service_type_0 編號 1 的成員，以此類推。

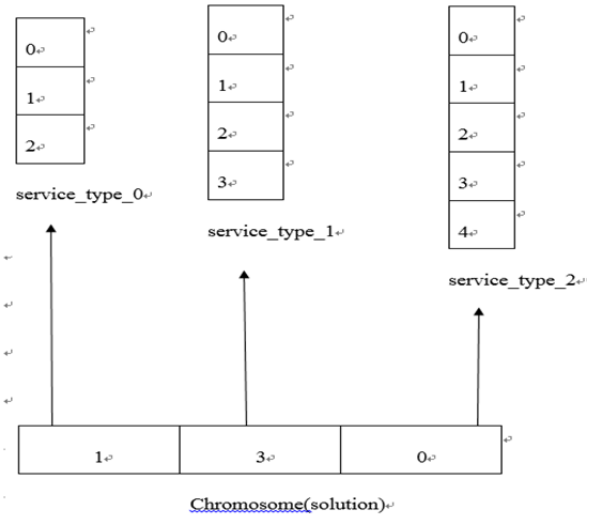

圖 2 基因編碼

5.3 適應性函數

根據前面定義的 QoS 指標定義 fitness function，用來評估解的優劣，並以此為依據判斷個體是否被淘汰，而當問題存在某些限制條件、或者服務存在相依性或者衝突性時,必須加入新的 fitness function 判斷是否符合或者不符合限制條件，若不符合條件，則從族群中移除該個體。

本研究單純以 QoS 指標條件進行尋找全域式的最佳化，計算出來的值越大，代表該個體的適應性越好，服務組合擁有更好的服務品質。

$$f(c) = \frac{\omega_A \prod_{i=0}^{n-1} A_i + \omega_R \prod_{i=0}^{n-1} R_i + \omega_S \sum_{i=0}^{n-1} S_i + \omega_{CR} \sum_{i=0}^{n-1} CR_i}{\omega_C \sum_{i=0}^{n-1} C_i + \omega_{RT} \sum_{i=0}^{n-1} RT_i} \quad (5.3)$$

上列式子是對於有 N 種服務的服務組合進行適應性函數計算，對於可用性(Ai)與可靠性(Ri)屬性做乘積，安全性(Si)、名聲(CRi)、花費(Ci)、回應時間(RTi)等屬性做加總，暫時不考慮服務之間的相依性或衝突性的計算方式，以正指標的屬性放在分子，負指標的屬性放在分母，並乘上使用者對於不同 QoS 的權重(Weight)$\omega_i (\omega_i \geq 0)$，求出來的值越高代表 Qos 品質越好。

5.4 族群數

族群數會影響尋找最佳解的搜尋或是否因為太小而陷入區域最佳解，因此必須考慮族群數對於運算時間所造成的影響,規模較小、較簡單的問題，族群數若設得太大與窮舉法無異，雖然可能在較少的演化世代數得到最佳解，但會造成搜尋長度太大而降低演算法效能，若設得太小會造成個體多樣性不足，容易陷入區域最佳解；而規模較大、較複雜的問題，族群數若設得太小，也需要演化更多世代才可能找到最佳解，而設得太大更加使得演算法效能低落。假設有 N 種服務類型，每種服務類型擁有

的服務數量不一定，而數量總共有 M 個，最佳的族群數設定並非為一個定值，因此本研究中的族群數設定為：

$$\text{Population Size} = \lceil \sqrt{N \cdot M} \rceil \qquad (5.4)$$

此式用以反應問題的規模大小與複雜程度，使得基因演算法可以不失其精神且不會造成演算法效能的過度影響，族群數大小分佈如圖 3 所示。

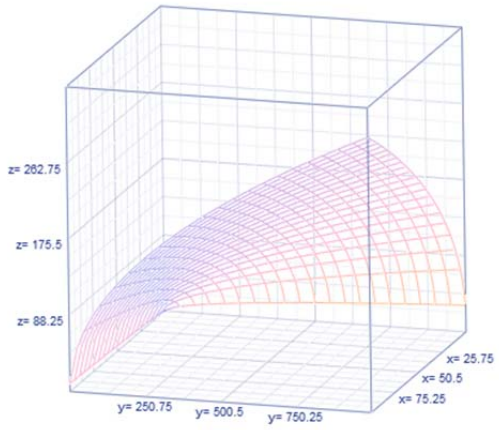

圖 3　族群數分佈

5.5 適應性選擇與世代的再生

在適應性比例選擇（fitness proportionate selection）的第一個步驟是根據染色體的優劣分數（score）去做降冪排序，之後根據分數比重(weight)大於某個值進行決定此染色體組合是否存活下來。

存活下來的染色體組合經過交配和突變後再達到相同的族群大小，並反覆的執行產生新子代，直到產生一定數量的子代(達到收斂條件)才停止。

經過不斷的將染色體族群進行分群篩選較優良的存活下來，並進行交配、突變等重複動作，當達到設定的收斂條件(產生第幾個世代、達到某個適應性分數)後，程式才停止運行，最後會得到一組適應性最高的服務組合的基因編碼，根據編碼的值對應到具體的服務。

6. 實驗分析

由於一般在做基因演算法求解時，族群數、交配機率與突變機率等參數的設定上沒有一定規則，即使在設定上沒有特別針對個別問題做設定，依然可以找到所需要的解，故於第四章中針對本研究之問題設定族群數以及交配機率，以期能夠有更好的效能。

本研究將不同種類的服務定義反應時間、可用性、花費等 QoS 指標數據，可透過此演算法計算服務等級，找出最佳的服務組合，針對不同的實驗參數進行演算法的效能評估。

本研究中基因演算法中定義的參數包含下列幾個參數：

- 服務種類(variety of services)-服務種類代表抽象化服務的數量，每個抽象化服務中擁有不同數量的具體服務與 QoS 屬性。
- 族群大小(Population size)-族群大小代表每次世代產生的族群數量
- 染色體存活率(Proportion of survivors in each generation)-染色體存活率代表每次進行適應性選擇時存活下來的個體佔原本總個體數量的比例。
- 交配機率(Crossover chance)-交配機率代表有多少個體會進行交配。
- 突變機率(Chance of random mutation)-突變機率代表個體基因中某段基因可能會突變的機率。
- 世代數(Number of generations)-世代數代表總共要產生的世代數量。

基因演算法在運算時間和產生的解答好壞取決於服務選擇問題的大小與複雜度，而問題的大小是由兩個參數決定：(1)在工作流中的抽象服務的數量(2)每個抽象服務所擁有的具體服務數量，當數量越大，問題的複雜度也越大。

實驗中設定 10 組不同的抽象服務種類，每個抽象服務擁有不同數量的實體服務，而每個實體服務也具有各種不同的 QoS 屬性，根據不同的突變機率執行後，如圖 4、圖 5 所示。

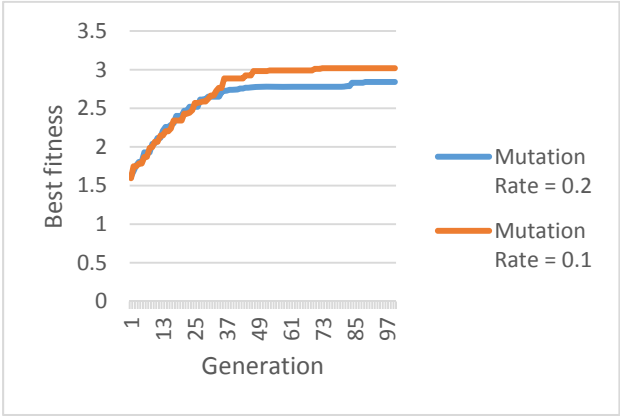

圖 4　最佳 fitness score

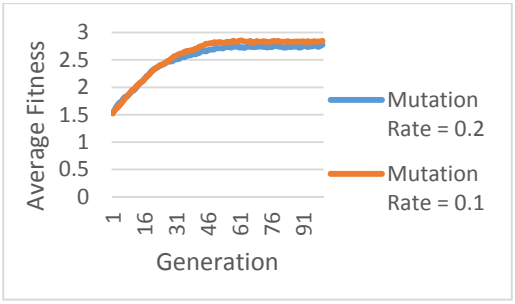

圖 5 平均 fitness score

從實驗數據中可以看出經過世代的演化後，族群組合的 fitness score 呈現上升的趨勢，代表路徑解答逐漸呈現最佳化，曲線逐漸收斂的範圍值與設定條件的最佳解的區間符合，而在本演算法在計算服務選擇的問題時，突變機率 0.1 明顯效率優於突變機率 0.2 的設定，是由於突變機率越大時，可能會改變的基因片段越多，帶來較差的結果導致，但是針對不同的案例或問題規模，比較小的突變機率不一定是最好，必須針對案例去調整適合的值。

本實驗中設定 10 組不同的抽象服務類型，分別有 10、20、30…100 種服務類型，每種服務類型擁有不同數量的實體服務與 QoS 屬性，測試不同數量的服務對於演算法效能的影響，詳細運算時間如圖 6 所示。

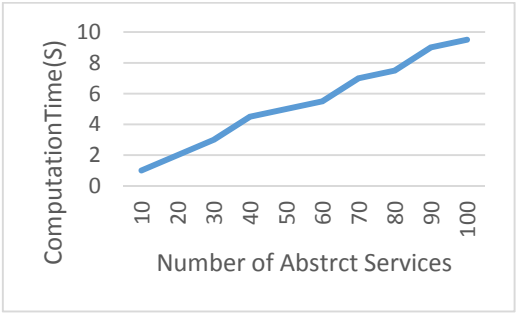

圖 6 不同數量的抽象服務的計算時間

當抽象服務類型越多，或者每一個抽象服務所擁有的具體服務數量越多，所要花費的計算時間相對地也越長，為線性成長的曲線分佈，在基因演算法的族群數設定上，如 5.4 節中所述是以$\sqrt{N \cdot M}$根據不同數量的服務動態地調整，不會發生問題規模不大，但是卻因為產生的族群數太多，導致運算時間變長的情形發生。

7. 結論與未來工作

本論文探討了多重服務考量的雲端服務選擇問題，根據具體服務本身的 QoS 屬性，透過基因演算法找出符合使用者需求的服務組合，主要的貢獻是設計了一個有效率的方法，可以幫助使用者在眾多的服務中，找到滿足客戶需求的服務組合，由於有些人可能要的是價錢最便宜的服務，有些人要的是最快的服務，有些則是要介於兩者之間的，因此 QoS 的屬性必須根據使用者的非功能性需求去制定服務指標，根據服務指標的屬性找出最適合的服務組合，由於服務的種類數量龐大與種種 QoS 條件的限制根據，如果要找出符合服務品質的服務組合是相當困難的，本研究使用 GA 幫助我們更容易的去選擇符合限制條件的組合，確實可以幫助在多個抽象服務中找出適合的服務，以保證服務的品質。

本研究僅針對特定服務選擇問題去設計與實作，未來可針對不同的條件限制去設計不同的演算法，由於服務選擇有不同的策略因素需要考慮，有時候要考慮到服務的相依性與衝突性，或許可以將演算法以 API/ Library 的方式方便使用者針對不同的情況使用。

參考文獻

[1] Michael R. Garey and David S. Johnson., "Computers and intractability: a guide to the theory of NP-completeness," W. H. Freeman, 1979.

[2] Jorge Cardoso, Amit Sheth, John Miller, Jonathan Arnold, and Krys Kochut, "Quality of service for workflows and web service processes," Web Semantics: Science, Services and Agents on the World Wide Web, 1(3):281 – 308, 2004.

[3] Object Management Group. UML Profile for Modeling QoS and Fault Tolerance Characteristics and Mechanisms., http://www.omg.org/spec/

[4] Shuping Ran, "A Model for Web Services Discovery with QoS," SIGecom Exch., 4(1):1–10, 2003.

[5] Liangzhao Zeng, Boualem Benatallah, Anne H.H. Ngu, and Marlon Dumas, "QoS-aware middleware for web services composition," IEEE Transactions on Software Engineering, 30(5):311–327, 2004.

[6] Holland, J. H., "Adaption in natural and artificial systems," The University Michigan Press, Ann Arbor , 1975

[7] GAUL: Genetic Algorithm Utility Library, http://gaul.sourceforge.net/intro.html(July 1st)

[8] Srinivas, M., "Genetic algorithms: a survey," *Computer*, Vol. 27, Issue 6, Pages 17-26, JUNE 1994

[9] Chang W. A., "Elitism-based compact genetic algorithms," *IEEE Transactions on Evolutionary Computation*, Vol. 7, Issue 4, Pages 367-385, AUG 2003

[10] Michael C. Jager, "Optimising Quaility-of-Service for the Composition of Electronic Services," PhD thesis, Berlin Unveristy of Technology, 2006.

[11] Poornachandra Sarang Matjaz B. Juric, Benny Mathew, editor, "Business Process Execution Language for Web Services," Packt Publishing, 2006.

[12] T. Yu and K. Lin, "Service Selection Algorithms for Composing Complex Services with Multiple QoS Constraints," LECTURE NOTES IN COMPUTER SCIENCE, vol. 3826, p. 130, 2005.

[13] Liangzhao Zeng, Boualem Benatallah, Anne H.H. Ngu, and Marlon Dumas, "QoS-aware middleware for web services composition," IEEE Transactions on Software Engineering, 30(5):311–327, 2004.

[14] Gerardo Canfora, Massimiliano Di Penta, Raffaele Esposito, Maria Luisa Villani, "An approach for QoS-aware service composition based on genetic algorithms," conference on Genetic and evolutionary computation, Page 1069-1075, 2005.

[15] Zhen Ye, Xiaofang Zhou, Athman Bouguettaya, "Genetic Algorithm Based QoS-Aware Service Compositions in Cloud Computing," Database Systems for Advanced Applications Lecture Notes in Computer Science Volume 6588, pp 321-334, 2011.