以遺傳編程為基礎之多步先網路服務品質預測

Genetic Programming-based N-Step-Ahead QoS Prediction

黃偉倫[1]，許揚[2]，范姜永益[1]
輔仁大學資訊工程學系[1]，國立臺北科技大學資訊工程學系[2]
b2266369@gmail.com, a29066049@gmail.com, yyfanj@csie.fju.edu.tw

摘　要

在軟體工程領域中，利用歷史服務品質(如回應時間等)資料預測未來服務品質數值，其預測結果能夠支持後續服務使用規劃以及相關決策。現今網路和雲端服務的使用與應用越來越熱門，已有許多這些服務的動態服務品質預測相關研究存在，常見的預測方法包括 ANN(Artificial Neural Network)，ARIMA(Autoregressive Integrated Moving Average model)等。本研究則是利用遺傳編程 (Genetic Programming, GP)，其精準度以及多樣性適合作為多步先預測的預測模型產生/搜尋方法。本研究提出並嘗試三種不同預測模式，預測結果優劣則是以預測精準度為基準，並且與傳統預測方法做比較。

關鍵字：遺傳編程、多步先預測、多預測器

1. 介紹

本研究專注於網際服務(Web Services, WSs)的動態服務品質(Quality of Service, QoS)[1][3][6] 預測。在這個軟體服務越來越流行與普及的時代，使用者是否會使用一個網際服務，其動態 QoS 像是回應時間(Response time)等會是一個主要的考量因素，也是本研究希望藉由預測方法進行分析與預測的目標。

網際服務的動態 QoS 數值並非固定不變的;相反的，動態 QoS 數值會隨著時間不斷上下變動(如圖 1)，因此必需要對其進行分析與預測，以期能夠預知其在未來的可能數值為何。

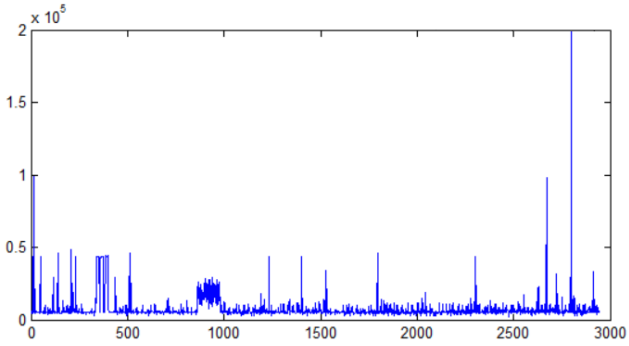

圖 1. Amazon 回應時間

常見的預測方法包括有 ANN、ARIMA 以及 GP 等，藉由輸入歷史時間序列資料、進行訓練，最後產生可用之預測器。若要預測的是單一未來下一時間點的資料，稱為一步先預測(One-Step-Ahead)。若預測的是未來多個時間點之資料，則需要進行多步先預測(N-Step-Ahead)。在多步先預測模式下，若只有產生單一預測器，則會需要重複使用此預測器後向推進產生多個預測數值。

本研究則是利用 GP 進行動態 QoS 的多步先預測。在本研究中，我們除了實作單預測器預測外，另外還會以每個未來時間點都有其專屬預測器的架構進行預測，希望藉此提高多步先預測的預測精確度，此為多步先多預測器(N-Step-Ahead N-Predictor)預測。由於多預測器 GP 需要同時演化生成 N 個預測器，計算成本遠較單預測器 GP 高，但是其目的在於能產生比單預測器更準確的預測結果。多預測器預測將會與各種多步先單預測器(N-Step-Ahead Single-Predictor)預測方法做比較，探討其使用是否值得。

在本研究中預測結果好壞的量化量度採用

MAE(Mean Absolute Error)以及 MAPE(Mean Absolute Percentage Error)[1]。利用 MAE 我們可以知道預測器生成的預測值與實際值的平均絕對誤差;而透過 MAPE 則可以知道絕對誤差值與實際值的比例，作為各預測方法與模式優劣比較之客觀依據。

本文的剩餘章節安排如下。第二章簡短回顧現有方法與研究；第三章定義研究問題與預測精確度量化測量方法；第四章針對所提出之 GP 方法與架構進行詳細的闡述；第五章描述實驗設計以及實驗結果；第六章則對實驗結果進行深入分析與探討;最後第七章則是我們的結論以及未來工作。

2. 相關工作

[1]對於網際服務品質與目前所面臨的問題有相當充足的介紹，並且定義了時間序列與預測器之間的關係。在[1][2][3]中則是介紹了許多現有時間序列預測方法，像是指數平滑、ARIMA 以及 ANN 等，同時提供了預測精確性測量的方法參考。

本研究提出使用 GP 進行多步先多預測器動態 QoS 預測的概念，並且將其與常見之時間序列預測方法做比較。透過[1][2][3]，我們選擇 ARIMA 及 ANN 作為所提出方法的主要比較對象，並且另外以平均值預測法的預測結果作為預測精準度的最低底線。

[4]首先提出使用多步先多預測器 GP 進行金融時間序列的塑模與預測的概念，而在本研究中我們則將此概念應用至多步先動態 QoS 的預測上。在[5]，作者區分了 SSP 以及 ISSP 兩種不同時間序列預測的模式， 此兩種模式也都將包含在我們的研究與實驗中。

除反映時間外，一個網際服務所會有的其他動態 QoS 屬性還包括了吞吐量與價格等。[6]利用了這些同服務內其他QoS時間序列所內含之資訊進行多變量多步先預測，結果顯示多變量時間序列預測可以得到比單變量更好的結果。

3. 問題定義與預測精確度測量

本研究的問題主要分為兩個維度。首先，傳統的預測方法採用的是一步先預測(One-Step-Ahead)，利用歷史時間序列資料預測未來下一時間點之數值。另一種則是多步先預測(N-Step-Ahead)，在目前時間點同時產生對未來多個時間點之預測值。理論上，預測目標離目前時間點越遠(預測原點)，預測之誤差會越大（越難預測)。在本研究中，提升多步先預測的精準度是主要的目標之一。

另一個維度則是單預測器(Single-Predictor)與多預測器(N-Predictor)之差別 。在多步先單預測器的模式下，單預測器需要負責產生所有未來時間點之預測，而在這裡我們則是希望每個未來時間點都能有一個專屬之預測器來負責產生其預測值，藉此提高多步先預測的精準度。

3.1. 時間序列符號定義

本研究專注於網際服務的回應時間預測。假設一回應時間時間序列指標集合(Index Set) 為 $T=\{0,1,2,3,\ldots,n\}$，此時間序列上共有 $n+1$ 個離散時間點。每個離散時間點所紀錄之回應時間 QoS 數值所構成之集合序列則為 $X=\{x_0,x_1,x_2,x_3,\ldots,x_n\}$。

本研究想預測的是未知的未來回應時間 QoS 數值，根據過去歷史資料產生對未來數值的預測;而在此問題結構下的一個重要指標是"目前時間點"。我們從時間序列中取一時間點做為目前時間點(cur)，並將 X 中的 x_{cur} 假設為目前可獲得的最後（最近）觀察值。則可將時間序列 X 分為 $X_{0\ldots cur}=\{x_0,x_1,x_2,x_3,\ldots,x_{cur}\}$ 過去歷史資料序列，以及測試用資料序列 $X_{cur+1\ldots n}=\{x_{cur+1},x_{cur+2},\ldots,x_n\}$ 。

3.2. 預測器

傳統的預測方法以單預測器預測下一步，為一步先單預測器(One-Step-ahead Single-Predictor)，其預測架構為 Single-Step-Prediction (SSP)[5]。利用單一預測器進行多步先預測時，預測器所需要的資料輸入範圍也會逐步往前移動，超出已知實際資料的範圍。文獻[5]的 Iterated Single-Step-Prediction (ISSP)架構，實際資料銜接每一次的預測結果成為未來時間點預測所需的輸入資料，讓一步先單預測器可以進行多步先預測。

預測器表示為 Predictor，多預測器則可表示為 PredictorN，預測器生成的未來預測值表示為 F_{cur+n}，N 以及 n 為大於等於 1 的正整數，X 序列為實際 QoS 資料。未來 N 個時間點的預測值可以表示成(1)：

$$
\begin{aligned}
F_{cur+1} &= \text{Predictor} \quad (x_0, x_1, ..., x_{cur}) \\
F_{cur+2} &= \text{Predictor} \quad (x_1, x_2, ..., x_{cur}, F_{cur+1}) \\
F_{cur+3} &= \text{Predictor} \quad (x_2, x_3, ..., x_{cur}, F_{cur+1}, F_{cur+2}) \\
&\vdots \\
F_{cur+n} &= \text{Predictor} \quad (x_{n-1}, ..., F_{cur+n-2}, F_{cur+n-1})
\end{aligned} \tag{1}
$$

單預測器使用 ISSP 架構，預測越後面的時間點將使用越多預測值，實際資料的比例會越來會少，甚至都採用預測值。

ISSP 也能應用在多預測器，與單預測器不同的即為每個時間點使用不同的預測器，生成的預測值將會把多預測器以迭代關係串在一起。除此之外也能夠輸入相同的實際資料進行預測，由於預測器的不同能夠產生不同的預測值，一方面能夠針對未來某個時間點進行預測並且不需要先預測中間未知的時間點，預測器之間也互不影響。使用不同預測器與相同實際資料可表示成(2)：

$$
\begin{aligned}
F_{cur+1} &= \text{Predictor1} \quad (x_0, x_1, ..., x_{cur}) \\
F_{cur+2} &= \text{Predictor2} \quad (x_0, x_1, ..., x_{cur}) \\
F_{cur+3} &= \text{Predictor3} \quad (x_0, x_1, ..., x_{cur}) \\
&\vdots \\
F_{cur+n} &= \text{PredictorN} \quad (x_0, x_1, ..., x_{cur})
\end{aligned} \tag{2}
$$

能夠直接預測較遠的時間點，而不用預測較近的時間點，在架構上比較簡單，必要時也能夠節省時間。

3.3. 測量

量化量度採用 MAE(Mean Absolute Error)計算實際數值與預測值之間的誤差，並利用 MAPE(Mean Absolute Percentage Error)計算誤差與實際數值的百分比。在 GP 的演化過程我們使用 MAE 作為個體在物競天擇的籌碼，能夠得到曲線較為平滑的 MAPE。MAPE 為進行實驗時，作為預測器精準度的量化量度，數值越低表示越精準。

在計算上(3)，*xi* 為實際數值，*yi* 為預測值 F_{cur+i}，*n* 為訓練或是測試時預測器運算的次數。

$$
MAE = \frac{1}{n}\sum_{i=1}^{n}\left(|xi - yi|\right) \quad MAPE = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{|xi - yi|}{xi}\right) \tag{3}
$$

4. 設計

在此章節將從單純 GP 的架構，延伸到多步先單預測器(N-Step-Ahead Single-Predictor)以及多步先多預測器(N-Step-Ahead N-Predictor)的架構，後者包含兩種架構、問題以及解決方式。

4.1. GP 的架構

最簡單的 GP 為一個運算元，可以是實際的資料或是經過統計之後得到的結果。使用運算子對運算元進行操作，則可堆疊成樹狀結構(圖 2)。將該樹狀結構攤開來，便是數學運算式，兩者可以互相轉換。

在這裡，定義一個數學運算式，也就是一個 GP 架構為一個染色體(chromosome)，也就是預測器。當一個個體(individual)只包含一個染色體，我們稱之為單預測器 GP；如果包含 N 個染色體(圖 3)，我們稱之為多預測器 GP，N 為大於等於 2 的正整數。

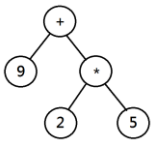

圖 2.GP 染色體

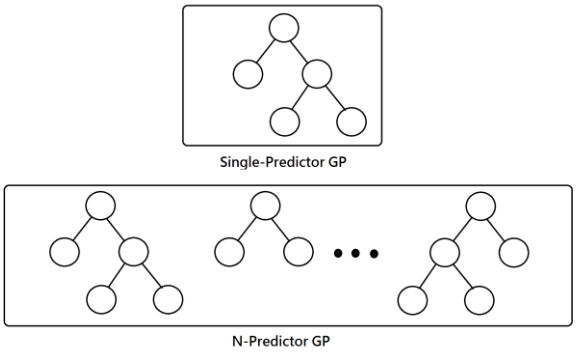

圖 3.N-Predictor 示意圖

多數個體組成人口(Population)進行演化，進行物競天擇，適應度(Fitness)高的個體將留下，反之淘汰。藉由一代又一代的演進，GP 的演化將針對問題提高其適應度，得到一個最好的個體來處理問題。

在本研究的 GP 染色體，分為 Function Set 以及 Terminal Set。Function Set 包含{+ ,- ,* ,/ ,pow ,exp ,log ,inc}；Terminal Set 則包含實數、QoS 資料以及統計方法{Average ,Maximum, Minimum, Standard Deviation , Variance }。

4.2. GP 的演化操作

4.2.1. 選擇 selection

選擇的操作是選出優秀的基因留到下一代，透過適應函數(Fitness function)針對每個個體進行適應度

計算，在這裡我們使用 MAE 作為適應函數。

4.2.2. 配對 crossover

配對的操作是將群體隨機挑出兩個體進行配對，並隨機交換基因的部分片段，希望能產生出更高適應度的個體。

4.2.3. 突變 mutation

突變的目標是造成意想不到的改變來適應環境，隨機改變染色體的一部分成為新個體。在 GP 的突變中，因為樹狀結構有分成運算子和運算元，要針對角色進行特定的突變才不會讓運算出錯。

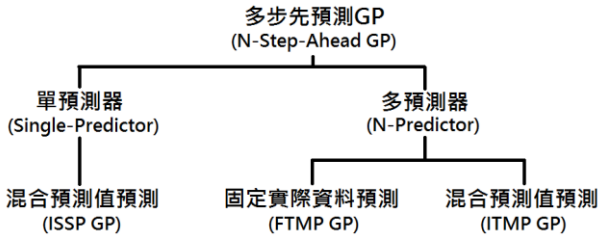

圖 4. 三種多步預測 GP

4.3 架構

本研究在設計上將多步先預測 GP 分成如(圖 4)的架構，在多預測器有分成兩種架構：使用固定實際資料與 ISSP 架構。

4.3.1 GP 各種預測器的架構

4.3.1.1 多步先單預測器 GP(N-Step-Ahead Single-Predictor GP)

以單預測器 GP 實行多步先預測，在 ISSP 架構下，除了第一個預測時間點的輸入資料都使用實際資料，其他時間點預測時都會需要較早時間點的預測值作為輸入資料(圖 5)。

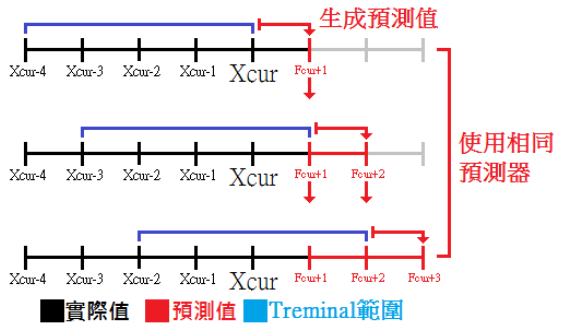

圖 5. N-Step-Ahead Single-Predictor GP

4.3.1.2 Fixed Terminal Multiple Predictor GP (FTMP GP)

不以預測值作為 GP 預測器的輸入，以相同的實際資料對未來不同時間點進行預測。因為不同時間點有著各自的預測器，輸出的結果將不相同(圖 6)。

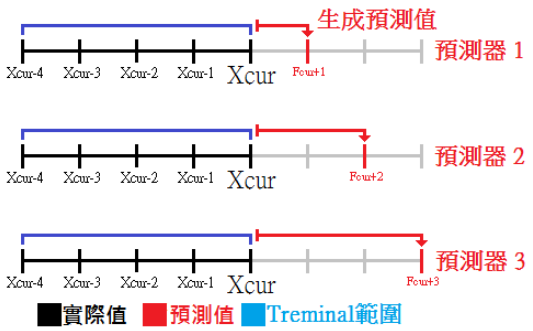

圖 6. FTMP GP

4.3.1.3 Iterated Terminal Multiple Predictor GP (ITMP GP)

在 ISSP 架構下，以多預測器實行多步先預測 (圖 7)，預測器因為使用預測值作為輸入資料，預測器之間存在著迭代關係。

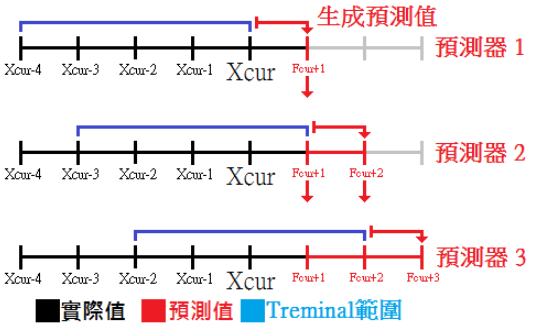

圖 7. ITMP GP

4.3.2 多步先多預測器 GP 的改良

在這裡我們將採用 ISSP 架構的多步先多預測器稱為 Iterated Terminal Multiple Predictor (ITMP)(圖 7)，越後面的預測器將會使用越多的預測值作為預測器輸入資料(Terminal)。另一方面，多步先多預測器也能讓不同預測器使用相同的實際資料作為輸入，預測器之間不存在迭代關係，實行基因工程不會發生問題，並且可以直接預測後面的時間，在這裡稱為 Fixed Terminal Multiple Predictor (FTMP)(圖 6)。

在多步先單預測器 GP，所有的預測值為同一個預測器所生成，該預測器的優劣即影響個體在物競天擇中的表現。

然而在多步先多預測器 GP，個體內的預測器優劣不一(圖 8)。進行個體適應度計算時，優良預測器會被個體中的劣質預測器拖累，個體表現不佳，使得優秀預測器被埋沒。越多的預測器存在同一個體，越容易發這種狀況，並造成多步先多預測器進化緩慢。

圖 8. 預測器優劣不一

面對上述問題，為了確保優良基因的存續，進行基因工程(Genetic Engineering, GE)，將每代群體中最優秀的預測器挑出來組成菁英(圖 9)，作為這一世代的代表，也成為下一代的父母。從(圖 10)可以看出有無進行基因工程對多步先多預測器 GP 的差異，透過取得優秀的預測器組成菁英個體，加速進化過程。

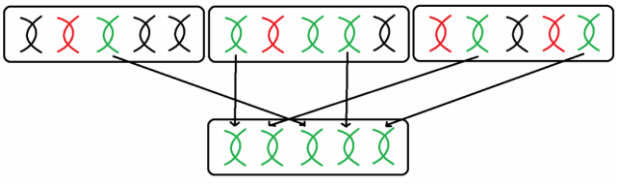

圖 9. 基因工程

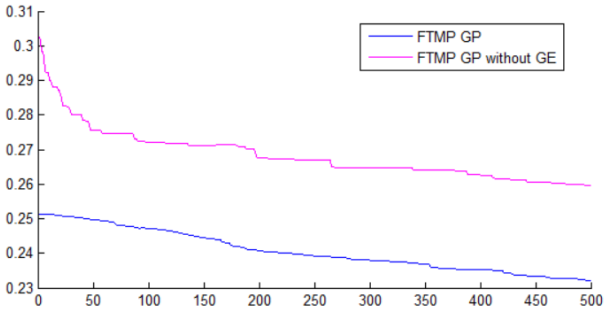

圖 10. 多步先多預測器 GP 有無進行基因工程比較

多步先單預測器 GP 可以延伸成多步先多預測器 GP，包含其 ISSP 架構的使用(圖 7)，也需要使用基因工程加速進化。在多預測器中使用 ISSP 架構，預測器產生的預測值將會往後影響，此迭代關係可以提升預測精度，進行基因工程將打破此關係。

假設一組最優秀的預測器(同一個體)，其一預測器被該時間點更好的預測器所取代，如果只看到這個時間點的確得到更好的 MAE，但是生成的預測值卻導致後面的預測器出現無法運算、更高的 MAE 或是沒比群體中、甚至上一代的好。

如果說 FTMP GP 是將基因工程作為演化的必須手段，那麼在考慮上述的問題後，我們希望他成為額外手段，加速 ITMP GP 進化，一但基因工程出問題或是沒有比自然進化的群體中好，將採用群體中最好的預測器(圖 11)。

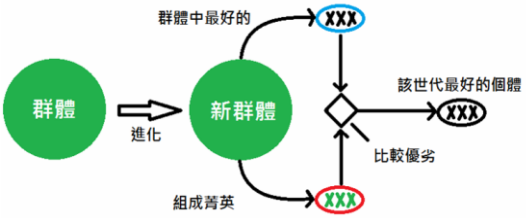

圖 11. ITMP 利用基因工程加速

4.4. 錯誤控制

利用訓練資料進行多步先 GP 的訓練，每一代進化完成後得到最優秀的個體，將該個體以測試資料進行測試並予以紀錄，藉此比較不同架構的多步先 GP 在進化時的優劣。但是訓練資料並不一定適合訓練好的預測器，測試中可能會產生無效預測，運算出無限大或是無效數值的狀況，並且因為傳宗接代的關係使得這種狀況接連出現。

在這裡我們將測試時有問題的個體以上一代通過測試的個體取代，並且依照不同架構處理。多步先單預測器 GP 以及 ITMP GP 發生狀況，由於預測器的迭代關係只能以上一代個體取代；但是在 FTMP GP 中，可以直接置換有問題的預測器(圖 12)。

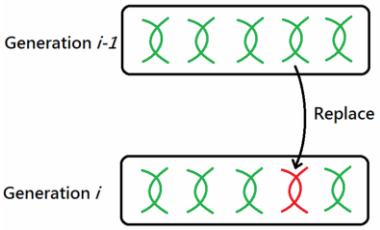

圖 12. FTMP 的錯誤控制

5. 實驗

在這個章節，進行各種預測方法在解決多步先預測問題的實驗與精準度比較。多步先單預測器包含單預測器 GP、ARIMA、ANN 以及平均值(Average)方法；在多預測器則為第四章提到的 FTMP 和 ITMP 兩種多預測器 GP。

5.1. 實驗環境與數據

ARIMA、ANN 和平均值使用程式語言 R 與 forecast 7.1[7]套件進行多步先預測；多步先單預測器、FTMP 和 ITMP 這三種 GP 使用程式語言 Java 與 Jgap 3.6.2[8]套件進行多步先預測的實作；輸出的結果使用 Matlab 進行彙整及繪圖。

實驗中所使用的服務[3]有 Amazon、Google、StockQuotes 以及 FastWeather，在 2006 年 7 月中到 11

中記錄的資料，每筆資料間隔一小時(FastWeather 間隔兩小時)，使用回應時間(Response time)作為 QoS 資料序列，各分 10 個片段進行實驗。在 ANN 以及三種 GP 架構，相同訓練資料會取得不同的結果，需要進行多次實驗取平均，所以上述的方法每個片段各跑 5 次。

每個片段包含 500 筆訓練資料以及後方接續著的 109 筆測試資料。在本實驗中預測未來 10 個時間點(N = 10)，進行 100 次測試，最後一次測試參與誤差計算的測試資料即為第 100 到 109 筆。

5.2 模型訓練

ARIMA 與 ANN 需要進行預測模型訓練，並且以相同筆數的輸入進行測試，讓模型能夠正常運行。進行的 100 次測試，第 1 次測試所需的輸入即為訓練資料；訓練出多步先預測模型後，隨即進行第一次的預測並計算 MAE 與 MAPE，接著使用相同模型預測之後的 99 次預測並計算 MAE 與 MAPE。

平均值方法不用進行模型訓練，但是以 500 筆的資料作為輸入，精確度非常的低，在這裡改用與 GP 相同的輸入筆數(與訓練資料筆數不同)，以 15 筆資料作為平均值方法的輸入，並且得到較高的精確度。

GP 的模型(預測器)訓練包含物競天擇以及演化的過程，訓練資料並不是一次輸入。在本實驗中，單一 GP 模型的輸入資料為 15 筆，總長度 500 筆的訓練資料必須先取前 15 筆做為第一次運算的輸入資料，總共運算 500-15 次，取平均得到 MAE 作為單一 GP 個體在物競天擇中的籌碼。

GP 在進化上的設定包含：人口數、世代數、選擇機率、配對機率以及突變機率。人口越多越能進化出多樣的個體，進化的世代數越多越能進化出越好的個體，但是進化時間也隨著兩者呈正比增，在實驗我們將兩者數值皆設為 500。進化操作的選擇與配對機率這裡採用 Jgap 預設的 0.1 與 0.9，突變機率則設為 0.02，主要進化手段即為配對。

在訓練結束後，以最優秀的 GP 個體進行測試，如同其他預測方法一樣運算 100 次並平均結果進行比較。GP 經過每一代的訓練都是越來越精準，但是在測試的時候卻會發生過度適應(Over Fit)，導致測試的 MAE 與 MAPE 提高許多甚至無法運算，前者導致其他預測方法結果曲線無法分辨時只能重作，後者以章節 4.4 提到的錯誤控制處理。紀錄其每一世代的數值，可以明確的分辨出進化狀況，就過度適應發生的機率做比較，FTMP GP 最高、多步先單預測器 GP 其次、ITMP GP 最不易發生。該狀況推測與架構有關，FTMP GP 每個時間點都要生成毫無關連又各自發展的龐大數學式，測試時發生問題的機率最高；多步先單預測器 GP 則是必須同時滿足多組測試資料；ITMP GP 進化到最後，數學式的發展會集中在前方的預測器，後方預測器會選擇沿用前方預測器的預測值或是稍作調整，不需要每個預測器都發展成龐大的數學式，面對問題有著令人意外的可塑性。

本研究的 GP 以 MAE 進行訓練，以實際誤差進行演化所得的 MAPE 相對平滑，預測器的優劣比較則需要 MAPE 的量度。所有預測方法使用測試所得的 MAPE 進行比較，以釐清預測方法的優劣。

5.3 實驗結果

所有預測方法完成 10 片段 10 步先預測的 100 次測試，將 MAPE 匯集取總平均，分不同服務進行比較。實驗結果 Y 軸為 MAPE，與預測精度成反比；X 軸為 GP 進化的世代數，其他預測方法由於沒有進化過程，將其結果以直線畫出(圖 13-16)。

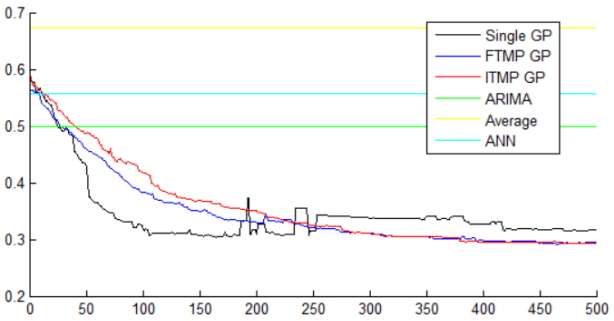

圖 13. Google

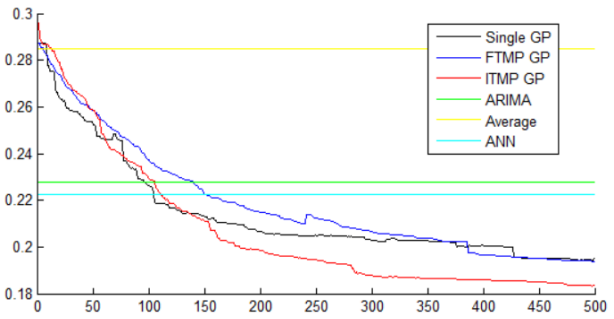

圖 14. Amazon

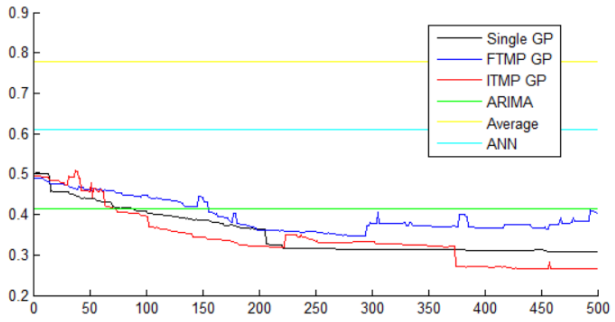

圖 15. StockQuotes

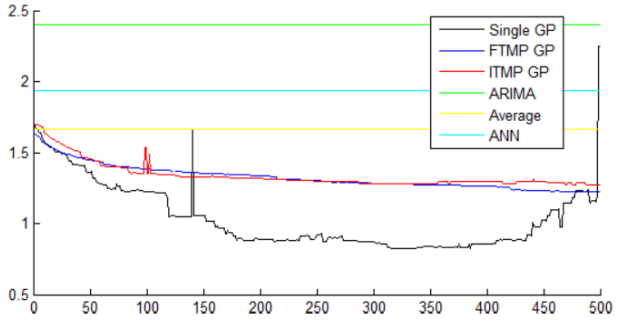

圖 16. FastWeather

6. 討論

實驗結果三種 GP 都比傳統的方法精準，同時實驗結果也展示出並沒有最好的多步先 GP 架構，觀察四種服務的進化過程以及結果，如果不考慮因為過度適應導致結果反轉，以最低的 MAPE 來看，多步先單預測器 GP 以及 ITMP GP 各別在兩個服務取得最佳預測。在相同服務的情況下，多步先多預測器 GP 所花費的時間是多步先單預測器 GP 的兩倍以上。

表 1. 服務標準差

服務	訓練資料 標準差	測試資料 標準差
Google	5.7393e+003	3.0756e+003
Amazon	5.7600e+003	4.7216e+003
StockQuotes	6.2858e+003	3.7711e+003
FastWeather	3.0369e+004	1.4236e+004

各種預測方法在四種服務表現都有差異，在這裡針對資料的離散程度使用標準差，並分為訓練以及測試兩組數值(表 1)。在訓練標準差不高的情況下(Amazon、Google、StockQuotes)，ITMP GP 能夠比多步先單預測器 GP 訓練的更好；標準差相當高的情況下(FastWeather)，多步先單預測器 GP 的訓練會比較好，並且在測試時出現最優秀的預測精度，只是在最後發生了嚴重的過度適應。在標準差相對較高的情況下，傳統預測方式表現的精準度也和較低標準差的服務有差別，可見離散程度對預測精度影響非常大。

另外在 Google 服務的部分，雖然在訓練時為 ITMP GP 最優秀，但是在測試階段卻發生多步先單預測器 GP 在進化途中出現全世代最低的 MAPE，如果沒有發生過度適應，多步先單預測器 GP 反而比較好。分析 Google 服務時，將全部實驗分成前後各五片段，前半片段(圖 17)是 ITMP GP 在測試時表現最好；多步先單預測器 GP 在後半片段(圖 18)得以先行降低 MAPE，雖然最低數值與 FTMP GP 相近，但是世代差距讓多步先單預測器 GP 在後半片段明顯優秀。其他的服務沒有發生片段差距甚大的問題，這部分的原因可能跟服務的資料分佈、片段或是雜訊有關，需要進一步研究。

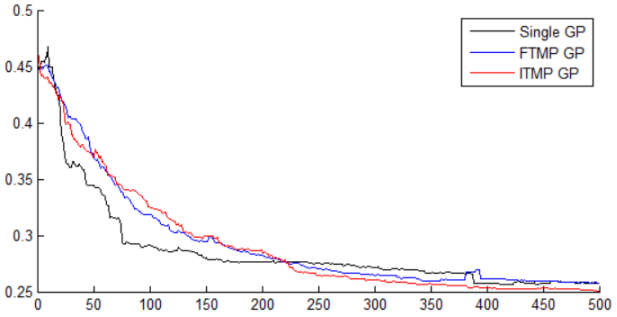

圖 17. Google 服務前半片段

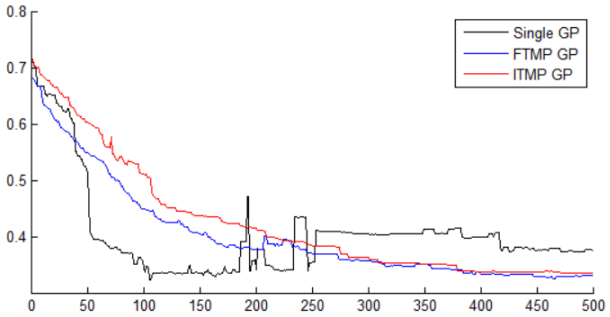

圖 18. Google 服務後半片段

多步先預測 GP 的多預測器與單預測器之間的差異，可能問題在於訓練上，訓練的目的便是以足夠的資料訓練出更精準的預測器，通常越多的訓練資料越好。對單預測器 GP 而言，多步先預測提供了數倍的訓練資料；而在多預測器 GP，個別預測器只用專屬於它的資料進行訓練。數倍的訓練資料使得單預測器 GP 能夠應付高標準差的資料序列甚至雜訊等問題，而多預測器 GP 反之。

本研究使用 GP 實作多步先預測，尤其 ITMP GP

能在標準差約 6000 左右達到最好的訓練，在測試的階段能夠比多步先單預測器 GP 更好。至於使用固定實際資料作為輸入的 FTMP GP，在實驗中的預測精度不穩定，同時發生過度適應的機率也異常的高。透過實驗可以發現多步先多預測器 GP 使用 ISSP 架構有助於提升預測精度與穩定性。但是除了標準差以外，資料序列還存在著許多因素影響預測精度，仍然有許多可以改進的空間。

7. 結論與未來工作

目前來看，多步先多預測器 GP 可以通過許多方式改良，像是使用多變量時間序列[6]、資料的預先處理或是加強預測器之間的關係[4]。多步先多預測器 GP 的演化也相當耗費資源，但是如果能夠進行平行處理，這方面的問題能夠減少。

在資料序列方面，實驗所用的資料是 QoS 的原始回應時間，並沒有將資料進行處理，像是平滑、差分或是去除雜訊等。是否能夠透過去除雜訊提高多預測器 GP 的預測精度，會是個很不錯的研究議題。

在設計方面，理想的多預測器位於同一個體內，而不是多個預測器個別計算誤差值選出最好的預測器組合在一起。多預測器 GP 的基因工程有些類似於後者，但也僅限於預測器之間沒有迭代關係的時候，此迭代關係存在於個體內，對預測精度有助益。

以基因工程解決多預測器進化緩慢，但是 FTMP 與 ITMP 的多預測器基因工程有一個差別。FTMP 沒有預測器的迭代關係，能夠取得該時間點實際最佳的預測器；ITMP 取得的是有條件的最佳預測器，其條件便是建立在前方預測器的預測值，評比標準不一。

進一步的構想是再改進 ITMP GP 的軟體工程，準備一個獨立空個體，群體進化完成後，依序放入預測器，預測器優劣評選所用的預測值輸入，為該獨立個體已有的最佳預測器配置。當然，計算成本相當高。

另一方面，保留上一代的個體，部分置換成新世代的預測器，透過排列組合是否能夠得到更好的個體？GP 本身預測精度相較傳統方法好，同時也有著相當程度的可塑性，如何挖掘它的優點會是未來的目標。

參考文獻

[1] Y. Syu, Y.-Y. FanJiang, J.-Y. Kuo, and S.-P. Ma, "Applying Genetic Programming for Time-aware Dynamic QoS Prediction", in 2015 IEEE International Conference on Mobile Services, New York, NY, USA, pp. 217 - 224, 2015.

[2] N. Wagner, Z. Michalewicz, M. Khouja, and R. Roy McGregor, "Time Series Forecasting for Dynamic Environments: The DyFor Genetic Program Model", IEEE Transactions on Evolutionary Computation, pp. 433-452 , 2007.

[3] B. Cavallo, M. Di Penta, and G. Canfora, "An Empirical Comparison of Methods to support QoS-aware Service Selection", in PESOS '10 Proceedings of the 2nd International Workshop on Principles of Engineering Service-Oriented Systems, New York, NY, USA, pp. 64 - 70, 2010.

[4] M. Santini, and A. Tettamanzi, "Genetic Programming for Financial Time Series Prediction", in EuroGP '01 Proceedings of the 4th European Conference on Genetic Programming, Springer-Verlag London, UK, pp. 361-370, 2001.

[5] A. Agapitos, M. Dyson, J. Kovalchuk, and S. M. Lucas, "On the Genetic Programming of Time-Series Predictors for Supply Chain Management", GECCO '08 Proceedings of the 10th annual conference on Genetic and evolutionary computation, New York, NY, USA, pp. 1163-1170, 2008.

[6] Z. Ye, S. Kumar Mistry and Hai Dong, "Long-term QoS-aware Cloud Service Composition using Multivariate Time Series Analysis", IEEE Transactions on Services Computing, 2014.

[7] Hyndman RJ (2016). *forecast: Forecasting functions for time series and linear models*. R package version 7.1, http://github.com/robjhyndman/forecast.

[8] Meffert, Klaus et al.: JGAP - Java Genetic Algorithms and Genetic Programming Package. URL: http://jgap.sf.net