服務關聯性導向的服務品質計算與網路服務組合

沈惟嘉
凌網科技
eaglet.lang@msa.hinet.net

吳秀陽、馬啟銘
國立東華大學資訊工程學系
showyang@csie.ndhu.edu.tw

摘要

本文提出以服務關聯性為導向的服務品質計算機制，作為在網路服務組合執行時，動態選擇網路服務的依據；首先使用圖形理論來表述與分析服務之間的關聯性，並以極大獨立集合成功的解決服務關聯性而造成的衝突；針對選擇最適當的網路服務部分，設計了最多關聯性、最大覆蓋、最多折扣及最大服務品質等關聯性選擇策略。考量工作流互斥與平行的執行對於服務組合所造成的影響，在工作流互斥執行部分，設計權重計算方式來評估各元件被執行之可能性以輔助服務品質計算；在工作流平行執行部分，採用漸進最佳化的方法來改善服務品質。實驗顯示，考量服務關聯性的服務品質計算機制可以更精準挑選出適合的網路服務，對於以網路服務為主之跨企業電子商務系統，提供更堅實的理論基礎與應用的實務價值。

關鍵詞：網路服務、網路服務組合、工作流、服務品質、服務關聯性

一、前言

網路服務(Web Services)的應用對於以網際網路為基礎的電子商務系統已經是一門成熟的技術，當單一的網路服務無法滿足複雜的資訊系統需求時，組合現有的網路服務成為一個全新且功能更強的組合稱為網路服務組合(Web services composition)。網路服務組合的出現為企業的電子商務帶來了新的應用思維，現階段的研究採用了工作流(workflow) 來表示網路服務組合的流程，並利用服務品質(QoS, Quality of Service) 計算，經由評量挑選出最適合使用者喜好的網路服務。

結合網路服務與工作流自動化是跨企業電子商務系統在建置過程中非常重要的選項，加入以服務關聯性為基礎的服務品質計算，在挑選網路服務時可以更符合電子商務環境的實際需求。但現階段網路服務挑選只考慮到單一網路服務的服務品質計算，並未考量到服務關聯性而影響到整體的服務品質計算，因此產生了研究的動機，提出以服務關聯性為導向作為服務品質計算的依據；首先探討服務與服務之間的關聯性，說明因工作流的特性所造成的影響，對於活動控制圖(AC-Diagram)[1]的每個元件加入權重的設計，用以評估每個網路服務被執行的可能性，並設計了四種挑選服務關聯性的策略，最後以漸進式最佳化的方式找到全域最佳化的解。

為達成上述目標，本文將分以下步驟來進行，包括：1、使用活動控制圖作為網路服務組合的塑模工具；2、進一步分析活動控制圖的語意特性以計算網路服務組合裡各元件在服務品質上的權重；3、設計服務品質模型；4、明確的定義服務關聯性的服務品質計算；5、.探討因服務關聯性產生可能的衝突，設計不同特性之關聯性挑選策略及以服務關聯性為基礎之網路服務組合；6、針對工作流的平行執行特性採用漸進最佳化方法達到全域最佳化的目標；7、系統實作與效能評估。

二、相關研究

在網路服務組合研究上，目前所面臨的最大的問題是如何在動態連接的階段由眾多功能相近的網路服務中挑選出最適合使用者的元件，常見的研究是利用服務品質作為挑選網路服務的依據；Liu[2]利用，例如:服務的價值、執行的時間、使用者評價、可靠性等功能性質當作品質標準(Quality Criteria)來評估一個網路服務的服務品質，再依據使用者對於品質標準的喜好給予權重，重新計算出網路服務上的服務品質分數，作為動態連接時網路服務元件的選擇依據，但以上的作法未考量服務關聯性而影響到整體服務品質的計算；Zang[3]提出了區域與全域服務品質最佳化的觀念，Zang 利用整數線性規劃(Integer Linear Programming)找出最適合的網路服務組合元件，Canfora[5]則使用基因演算法(Genetic Algorithm)來挑選網路服務，但是也未加以討論服務之間的影響。

觀察現實生活，當網路服務成長為一種商業行為時，會因為策略聯盟而使得兩個不同的網路服務必須同時使用，服務之間的相互影響就產生了，進而影響到整體的服務品質計算，以上的現象命名為服務關聯性(Service dependency)；以現實生活為例，當某些商品合在一起購買時商家才提供折扣，部分的消費者會因為考量到折扣而增加購買的商品；同樣的道理，當兩個以上的網路服務同時北使用時，彼此間產生影響，應該將關聯性所帶來的影響列入網路服務選擇的考慮因素中。

工作流(Workflow)是目前最常被使用在網路服務組合上的技術，其中 WfMC(Workflow Management Coalition)在技術上的發展及規格的制定[4]，Wu[1]在樣板的建立及研究都趨於成熟，至於網路服務組合塑模語言，如 BPEL4WS[6] 及 BPML[7]等，Aalst[9]曾對眾多的塑模語言做過詳細的介紹，並以工作流樣板(Workflow Template)的角度來比較網路服務組合塑模語言的優劣。

本文採用活動控制圖做為分析與塑模網路服務組合的基礎，但因為活動控制圖本身並沒有考量互斥執行造成的影響，也沒有服務品質的計算方式，所以我們除了分析活動控制圖各種元件的語意特性及設計出權重計算的方式之外，也定義了一個完整的服務品質模型來探討服務間的關聯性。

三、活動控制圖塑模網路服務組合

(一) 活動控制圖

完整的活動控制圖必須是一個 bipartite graph 包含一組活動圖件、一組控制圖件和一組關係來表示活動與控制之間的流程與相互關係。現存的流程描述工具只有通用的活動圖示，而未提供類似活動圖件的機制。在工作流的設計上須明確訂出活動的類別、語意性質，特別是資訊傳輸流向，有助於使用者的商務流程模式化的過程及系統的自動化分析和處理。

活動圖件用來描述商務活動，我們採用如圖 3-1 活動圖件，展示的一般活動，讓企業的商務活動描述更有彈性。

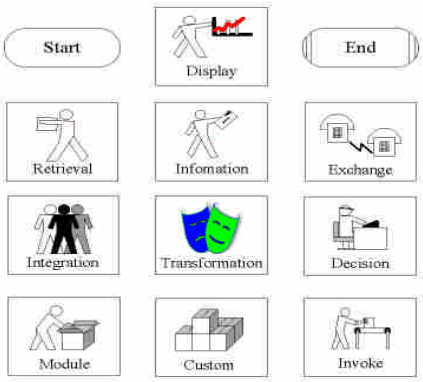

圖 3-1 活動圖件

圖 3-2 控制圖件，顯示的控制圖件可用來描述各種流程定義中所需要用到的控制結構，完整的表達了商務活動之間的複雜關係。在完成商務程序之活動控制圖設計後，商務模式化系統可以依據正規語意，自動的將活動控制圖轉換成 XML 工作流規範(XML workflow specification)，成為跨企業代理人交易系統之輸入。交易系統的任務，正是將此一商務流程在網際網路上順利執行，同時保證流程執行的交易特性。

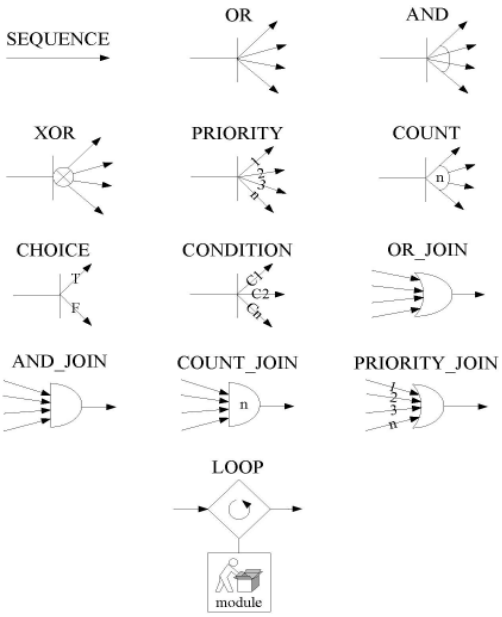

圖 3-2 控制圖件

(二) 網路服務組合之塑模

一個工作流程被定義為 (A,C,R)

$A=\{a_1,a_2,...,a_n \mid \forall i, a_i \in \mathrm{A}\}$ ，

代表所有活動圖件的集合。

$C=\{c_1,c_2,...,c_m \mid \forall i, c_i \in \mathrm{C}\}$ ，

代表所有控制圖件的集合

$R=\{(I,c,O) \mid I \subseteq A, O \subseteq A, c \in C\}$ 。

代表所有關係的集合。

在本文中，我們將網路服務組合視為一個工作流，所有的活動圖件都視為網路服務呼叫，但是網路服務組合裡面所呼叫的網路服務是在執行時才動態決定的，所以須重新定義網路服務組合為 (T,C,R)，其中 $T=\{t_1,t_2,...,t_n\}$ 是網路服務組合裡需被執行的任務集合，安排相對應的網路服務來執行網路服務組合裡的任務，稱之為網路服務組合之執行計劃，定義 p 為一個網路服務組合的執行計劃：

$$p=\{<t_1,s_1>,<t_2,s_2>,...,<t_n,s_n>\} \text{ ，}$$

其中 $t_1,t_2,...,t_n$ 為網路服務組合裡所有的任務之集合，且 $s_1,s_2,...,s_n$ 分別為執行任務 $t_1,t_2,...,t_n$ 所安排之網路服務，有了執行計劃後便可以將之轉換為真正的工作流程來執行。

(三) 活動控制圖之權重分析

在工作流程裡面，由於存在互斥性質的元件而導致每個元件被執行到的機率不同，因此需評估所有元件被執行到的機率，如此才能判斷不同的元件會對整體的工作流造成的影響程度，但是活動控制圖跟傳統工作流語言的差別在於，傳統工作流語言是一個分支(Split)對應一個結合(Join)，而活動控制圖允許，如圖 3-3. 彈性之分支結合結構，來源相

同之分支連接至不同的結合的情況發生，所以無法用機率來判斷元件被執行的可能性，因此我們改以權重的概念來模擬機率的發生，作法是每個元件都被分配到一個權重 w，其中 $0 \le w$，當權重的值越大代表元件影響整體工作流的程度越大，我們分別定義 $w_c(c)$ 、 $w_e(e)$ 為元件 c 及連結 e 的權重。

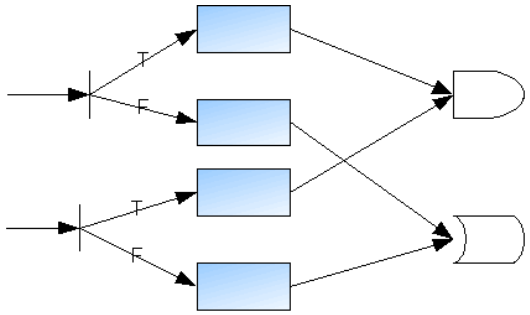

圖 3-3. 彈性之分支結合結構

我們將活動控制圖裡的活動圖件跟控制圖件分為三大類型來說明，並列出其權重的計算方式，其中 c 代表圖件本身：

■循序(Sequence)類型：

此類型包含了 SEQUENCE、LOOP 及所有的活動圖件，特徵是只有一個輸入及輸出的連結，其中 i 代表 c 輸入的連結， o 代表 c 輸出的連結。

對 SEQUENCE 圖件及其他的活動圖件來說，因為沒有分支也沒有結合，所以我們定義這類型圖件輸出的連結之權重及圖件本身之權重都等於輸入的連結之權重：

$$w_c(c) = w_e(i) \text{ ，}$$

$$w_e(o) = w_e(i) \text{ 。}$$

對 LOOP 圖件而言，因為所包含的模組圖件可能會被執行多次，所以我們根據過去執行的經驗來得出平均執行其模組圖件的次數，我們為其記錄一個表格，格式為 $<number, times>$，$number$ 代表曾經發生過的連續執行其模組的次數， $times$ 為連續執行 $number$ 次模組之次數，我們假設此表格為 NT 且 $NT = \{<1, times_1>, <2, times_2>, ..., <n, times_n>\}$，$n$ 為目前為止發生過的最多連續執行模組次數，相關權重計算方式如下：

$$w_c(c) = \frac{\sum_{i=1}^{n} i * times_i}{\sum_{i=1}^{n} times_i} * w_e(i) \text{ ，}$$

$$w_e(o) = w_e(i) \text{ 。}$$

我們得出平均模組被執行之次數之後再乘上輸入連結之權重來成為這個 LOOP 元件之權重，輸出連結之權重則跟輸入連結之權重一樣。LOOP 元件本身的權重是所有權重計算中唯一有可能超過 1 的元件。

■分支(Split)類型：

此類型包含了 AND、OR、XOR、CHOICE 、CONDITION、COUNT 及 PRIORITY 等控制圖件，其中 i 為 c 輸入的連結， $O = \{o_1, o_2, ..., o_n\}$ 為 c 所有輸出連結之集合。

對 AND 圖件而言，此圖件會對所有的分支平等對待且平行地執行，所其權重設定為：

$$w_c(c) = w_e(i) \text{ ，}$$
$$w_e(o) = w_e(i), \forall o \in O \text{ 。}$$

每個輸出分支之權重都等於輸入連結之權重，代表所有的分支均會被執行。

對 OR 及 XOR 圖件而言，這兩個圖件都會選擇任一分支往下執行，若所選擇的分支執行失敗，OR 會嘗試執行其他分支，但是 XOR 則不會。對這兩個圖件的權重設定為：

$$w_c(c) = w_e(i) \text{ ，}$$

$$w_e(o) = w_e(i)/n, \forall o \in O \text{ 。}$$

因為每個分支被選到的機率是相等的，表示每個分支被執行的機率為 $1/n$ 。

對 CHOICE 圖件而言，其元件上只有兩個分支，分別代表此圖件要求的條件是成立或是失敗而執行，在此我們利用過去的執行經驗來判斷權重，首先我們先定義 p_e 為分支 e 在此網路服務組合中到目前為止的成功執行機率：

$$0 \le p_e \le 1 \text{ 且 } \sum_{e \in O} p_e = 1 \text{ 。}$$

相關的權重設定如下：

$$w_c(c) = w_e(i)$$

$$w_e(o) = w_e(i) * p_o, \forall o \in O$$

用機率來決定權重以符合我們 CHOICE 的語意。

對 CONDITION 圖件而言，只要目前的執行狀況能夠符合分支的執行條件，分支即可往下執行，跟 CHOICE 最大的不同是 CONDITION 的分支沒有互斥的性質。在此也是使用過去的經驗來判斷分支的權重，首先定義 p_e 為分支 e 目前為止的成功執行機率，其中 $0 \le p_e \le 1$，其權重的設定為：

$$w_c(c) = w_e(i) \text{ ，}$$
$$w_e(o) = w_e(i) * p_o, \forall o \in O \text{ 。}$$

因為 CONDITION 元件只要分支的條件成立即會繼續執行下去，所以我們藉著過去經驗來求出所有分支各自成功的機率，然後再將輸入連結之權重乘上成功執行機率來計算分支的權重。

對 COUNT 圖件而言，當執行到此圖件時會任選 COUNT 指定數目的分支往下執行，假設 COUNT 指定執行的數目為 m 的話，權重為： $w_c(c) = w_e(i)$ ，任一分支會被選擇繼續執行的機率為： $\frac{C_{m-1}^{n-1}}{C_m^n}$ ，相關權重計算方式如下：

$$w_e(o) = w_e(i) * \frac{C_{m-1}^{n-1}}{C_m^n}, \forall o \in O \text{ 。}$$

為符合了 COUNT 圖件的語意，每個分支的權重都等於輸入的權重乘上 $\frac{C_{m-1}^{n-1}}{C_m^n}$ ，。

對 PRIORITY 圖件而言，會根據優先權的高低來一一執行分支直到有任一分支成功為止，在此也用過去的執行經驗來判斷權重，首先先定義 p_e 為分支 e 到目前為止使得 PRIORITY 能夠繼續往下執行的機率，其中

$$0 \le p_e \le 1 \text{ 且 } \sum_{e \in O} p_e = 1 \text{ ，}$$

權重設定的如下：

$$w_c(c) = w_e(i) \text{ ，}$$

$$w_e(o) = w_e(i) * p_o, \forall o \in O \text{ 。}$$

如同 CHOICE 一樣，用機率的方式來判斷我們該如何切割權重。

■結合(Join)類型：

此類型包含了 AND_JOIN 、 OR_JOIN 、 COUT_JOIN、PRIORITY_JOIN 等控制圖件，其中 $I = \{i_1, i_2, ..., i_n\}$ 為 c 所有輸入連結之集合，o 為 c 的輸出連結。

對 AND_JOIN 圖件而言，此圖件必需等到所有的輸入連結都到達才可以繼續往下執行，其權重的設定為：

$$w_c(c) = \min(w_e(i_1), we(i_2), ..., we(i_n)) \text{ ，}$$
$$w_e(o) = \min(w_e(i_1), we(i_2), ..., we(i_n)) \text{ 。}$$

對所有權重取最小值是因為 AND_JOIN 這個圖件包含了必須等待所有輸入完成的語意，所以我們找出所有輸入連結權重的最小值，作為 AND_JOIN 的權重以符合圖件的語意。

對 OR_JOIN 圖件而言，此圖件只要有任一輸入完成，即可繼續往下執行，其權重的設定為：

$$w_c(c) = \max(w_e(i_1), we(i_2), ..., we(i_n)) \text{ ，}$$
$$w_e(o) = \max(w_e(i_1), we(i_2), ..., we(i_n)) \text{ 。}$$

對所有權重取最大值是因為 OR_JOIN 只要有任一輸入完成即可往下執行，所以我們取最大值的權重以符合這個圖件的語意。

對 COUNT_JOIN 圖件而言，此圖件必需等待至少指定的數目之輸入完成才能繼續往下執行，相當於是 AND_JOIN 跟 OR_JOIN 的結合，假設 COUNT_JOIN 至少要完成的輸入之數目為 m 的話，那麼會有 C_m^n 種組合情況讓 COUNT_JOIN 判斷執行，對這 C_m^n 種情況單獨分開來看的話，相當於是必須等待 m 個輸入的 AND_JOIN，再將這 C_m^n 種情況結合起來的話，相當於是一個有 C_m^n 個輸入的 OR_JOIN ，根據上面我們為 AND_JOIN 跟 OR_JOIN 所定義的權重設定，推出對 COUNT_JOIN 的權重設定應該是先對 C_m^n 種情況各自取最小值之後再取其中之最大值，式子可化簡為：

$$w_c(c) = \text{the m largest of } (w_e(i_1), w_e(i_2), ..., w_e(i_n))$$
$$w_e(o) = \text{the m largest of } (w_e(i_1), w_e(i_2), ..., w_e(i_n))$$

或

$$w_c(c) = \text{the (n-m+1) smallest of } (w_e(i_1), w_e(i_2), ..., w_e(i_n))$$
$$w_e(o) = \text{the (n-m+1) smallest of } (w_e(i_1), w_e(i_2), ..., w_e(i_n))$$

因為 C_m^n 種情況都要取最小值，這時所有權重中比第 m 個大的權重都會被忽略掉，而 C_m^n 種情況裡的最小值之中最大的剛好就是第 m 大的權重，所以再取最大值的時候就可找出第 m 大的權重。

對 PRIORITY_JOIN 圖件而言，因為 PRIORITY_JOIN 的語意是除了優先權最高的輸入到達之後才可直接往下執行，其他的則是要等到 PRIORITY_JOIN 本身的等待時間結束後來會根據優先權由高向低檢查是否有輸入完成，有的話才會根據成功的輸入連結資料繼續往下執行，不過等待的性質跟 OR_JOIN 圖件一樣，只要有任一成功的輸入即可，其權重的設定為：

$$w_c(c) = \max(w_e(i_1), we(i_2), ..., we(i_n)) \text{ ，}$$
$$w_e(o) = \max(w_e(i_1), we(i_2), ..., we(i_n)) \text{ 。}$$

跟 OR_JOIN 一樣，取所有輸入權重的最大值。

四、服務關聯性模型

用來評估服務品質所使用的品質標準(quality criteria)，分成基本網路服務(Elementary Web service)及組合網路服務(Composite Web service)兩部分，分別說明服務品質的計算方式。

(一) 基本網路服務之品質標準

我們使用了五個品質標準，分別是：執行價格(Execution price)、執行時間(Execution duration)、使用者評價(Reputation)、可靠性分數(Reliability)及可得性(Availability)分數[10]，其定義及計算方式分述如下：

■執行價格：

定義 $q_{price}(s, op)$ 為呼叫網路服務 s 中的 op 功能所要支付給服務提供者的費用。

■執行時間：

定義 $q_{duration}(s, op)$ 為呼叫網路服務 s 中的 op 功能所需要花費的平均時間，時間的計算是從送出呼叫訊息開始，到接收到回傳訊息為止的時間，計算方式如下：

$$q_{duration}(s, op) = \frac{\sum_{i=1}^{n} T_i(s, op)}{n} \text{ ，}$$

計算出服務 s 中的 op 功能每次呼叫平均花費的時間，成為服務 s 執行時間之品質條件。

■使用者評價：

設定使用者評價的分數是從 1 到 10 的整數分數，最高評價分數是 10，最低是 1。定義網路服務 s 的使用者評價計算方式如下：

$$q_{reputation}(s) = \frac{\sum_{i=1}^{n} R_i}{n} \quad ,$$

其中 n 為目前為止網路服務 s 被呼叫過的次數，R_i 表示第 i 次的評價。

■可靠性分數：

設定分數的範圍從 0 到 10，最高分數是 10 分，最低是 0 分。網路服務 s 的可靠性分數計算方式定義為：

$$q_{rscore}(s) = \frac{N_c(s)}{k} * 10 \quad ,$$

其中 N_c 為目前為止網路服務 s 被呼叫過且成功的次數，k 表示目前為止被呼叫過的總次數。

■可得性分數：

設定分數的範圍從 0 到 10，最高分數是 10 分，最低分數是 0 分。網路服務 s 的可得性分數計算方式定義為：

$$q_{ascore}(s) = \frac{T_a(s)}{\theta} * 10 \quad ,$$

其中 T_a 為最近 θ 秒內網路服務 s 正常回應秒數，其中 θ 的設定可以隨著網路服務類型的不同以及對服務要求的不同來調整，例如當網路服務具迅速變動的特性時，θ 可設為小一點的值，如即時的查詢服務。

根據上面對所有品質條件的定義，對每個網路服務 s 及所要呼叫的功能 op 而言，會有一個服務品質的向量，此向量為 $q(s,op)$，其定義如下：

$$q(s,op) = (q_{price}(s,op), q_{duration}(s,op), q_{reputation}(s), q_{rscore}(s), q_{ascore}(s))$$

(二) 組合網路服務之品質標準

網路服務組合在本文中被視為一個樣板，裡面並沒有包含網路服務，真正定義的服務品質計算方式其實是針對網路服務組合的執行計劃，在此先假設 p 為網路服務組合 wsc 的執行計劃:

$$p = \{< t_1, s_n >, < t_2, s_n >, ... < t_n, s_n >\}$$

相同於基本網路服務之服務品質條件，在網路服務組合中亦區分為執行價格、執行時間、使用者評價、可靠性分數及可得性分數[10]等五項條件：

■執行價格：

定義 $q_{price}(p)$ 為網路服務組合 wsc 之執行計劃 p 的執行價格，並定義計算方式如下：

$$q_{price}(p) = \sum_{i=1}^{n} q_{price}(s_i, op(t_i)) * w_c(t_i)$$

其中 $op(t_i)$ 為任務 t_i 所呼叫之功能，算出每個網路服務各自的價格乘上各自的權重並加總之後來成為此執行計劃之價格。

■執行時間：

定義 $q_{duration}(p)$ 為網路服務組合 wsc 之執行計劃 p 的執行時間，因為執行時間的評估需要考量到工作流的特性，所以用傳送時間的方式來評估執行時間。從 START 元件開始，根據不同的元件來傳送時間給下一個連接的元件直到到達 END 為止，在此定義 $q_{cptime}(c)$ 為元件 c 上的傳送時間、$q_{eptime}(e)$ 為連結 e 往下傳送的時間。

根據不同元件的特色來改變時間傳送的方式，例如 AND_JOIN 需要等待所有的輸入都完成，所以傳送給下一個元件的時間即是所有輸入工作流裡最長的時間；而 OR_JOIN 只需要一個輸入完成即可，但是在此取得是平均值而不是最小值，這是因為所有的時間都有乘上相對應的權重，如果取最小值的話，那便有可能取到權重很小但是執行時間很長的工作流之時間而失去客觀，所以用平均值來代替最小值；對 COUNT_JOIN 而言，先為 C_m^n 種情況各自取最大值，然後再對這 C_m^n 個最大值再取平均值來當作 COUNT_JOIN 的傳送時間，$\{mtime_1, mtime_2, mtime_{C_m^n}\}$ 代表我們對 C_m^n 種情況各自取最大值的結果。

此外除了 JOIN 類型的控制元件，其他的控制元件都沒有等待的性質，所以直接傳送原先輸入的時間給下一個元件。當執行時間傳送到 END 元件 end 之後，時間傳送便停止。在此定義：

$$q_{duration}(p) = q_{cptime}(end)$$

END 元件上的 q_{cptime} 即是執行計劃的執行時間。

■使用者評價：

定義 $q_{reputation}(p)$ 為網路服務組合 wsc 之執行計劃 p 的使用者評價，並定義計算方式如下：

$$q_{reputation}(p) = \sum_{i=1}^{n} q_{reputation}(s_i) * w_c(t_i)$$

跟執行價格一樣，計算出各自的評價分數乘上權重並加總。

■可靠性分數：

定義 $q_{sescore}(p)$ 為網路服務組合 wsc 之執行計劃 p 的可靠性分數，並定義計算方式如下：

$$q_{rscore}(p) = \sum_{i=1}^{n} q_{rscore}(s_i) * w_c(t_i)$$

跟執行價格一樣，計算出各自的分數乘上權重並加總。

■可得性分數：

定義 $q_{ascore}(p)$ 為網路服務組合 *wsc* 之執行計劃 *p* 的可得性分數評價，並定義計算方式如下：

$$q_{ascore}(p) = \sum_{i=1}^{n} q_{ascore}(s_i) * w_c(t_i)$$

跟執行價格一樣，計算出各自的分數乘上權重並加總。

根據網路服務組合上之執行計劃的服務品質條件之定義，對每個網路服務組合之執行計劃 *p* 可得出一個服務品質的向量，此向量為：

$$q(p) = (q_{price}(p), q_{duration}(p), q_{reputation}(p), q_{sescore}(p), q_{ascore}(p))$$

但是在本文中所設計的服務挑選演算法並沒有利用到網路服務組合之服務品質的計算方法，不過為了能跟其他方法挑選出來的執行計畫做一個客觀的比較，所以在此仍要定義出網路服務組合在各個服務品質之計算方式。

(三)服務品質之服務相關性

根據現實世界的情況，我們發現許多不同種類不同性質的關聯性，除了影響價格的方式不同之外，也有的關聯性要求要全部條件都符合，例如指定的商品都要購買才有折扣，也有的關聯性只需要部分符合即可，例如四種商品只要任購三種即可發生關聯性，下面為不同種類之關聯性的介紹：

■相同折扣率：

當商品發生關聯性時，所有的商品都會根據相同的折扣率打折，例如商品 A 跟商品 B 有關聯性，同時購買符合這個關聯性之時，兩種商品都會有九折的優惠。

■相異折扣率：

當商品發生關聯性時，不同的商品有不同的折扣率，例如商品 A 跟商品 B 有關聯性，但是商品 A 的折扣率是九折，商品 B 則是八折，兩種商品的折扣率不同。

■根據使用次數的不同折扣率：

當某消費者對同一種關聯性使用次數累加到一種程度之後，折扣率會改變，例如商品 A 跟商品 B 有關聯性，一開始的折扣率為九折，但是只要同一位消費者使用關聯性的次數超過三次以上之後，從下次開始折扣將會變成八折。

■指定價格：

當商品發生關聯性時，所有商品的價格之總和會被指定為一個較便宜之價格，例如商品 A 的單價為一百元，商品 B 的單價為兩百元，兩個一起購買只需兩百五十元。

■額外附贈：

當滿足關聯性之時，會直接贈送其他商品或是給予某種滿足關聯性的證明，下次即可使用此證明來得到某種優惠，例如同時購買商品 A 及商品 B，可以得到商品 C 的八折優待卷，下次購物時便可直接使用。也有其他不同情況的例子，例如商品 D 可以買二送一等。

其他應該還有許多不同情況之關聯性，而且不止侷限在價格方面的影響，如果兩個不同的服務是同一家公司所提供的話，那麼一起使用它們便可節省網路上傳遞的時間，這就是執行時間方面的關聯性，但是在本文中，現階段只考慮最常見的價格影響，其他服務品質方面的影響則不列入考慮。

上面雖然列舉了不同影響價格方式的關聯性，不過在本文中只考量相同折扣率以及指定價格，而且只考量整體性的關聯性，並不考量部分性質的關聯性，也就是假設所有的關聯性都會要求指定的網路服務全部都要符合才行。在此定義 *UD* 代表相同折扣率，*FP* 代表指定價格，接著定義 $D = \{UD, FP\}$ 為這兩種關聯性之集合，然後定義服務關聯性 *sd* ：

$$sd = < S, dt, d >$$ ，

$$S = \{< s_1, op_1 >, < s_2, op_2 >, ..., < s_n, op_n >\}$$ 。

為發生關聯性 *sd* 所必需包含的所有網路服務及對應的功能之集合，*dt* 為關聯性的種類且 $dt \in D$，*d* 為關聯性相關的變數，如果 $dt = UD$ 的話便代表折扣率，$dt = AP$ 的話便代表指定價格。定成定義關聯性之後，得重新定義具關聯性之價格品質條件的計算方式，當一個關聯性 $sd = < S, dt, d >$ 且 $S = \{< s_1, op_1 >, < s_2, op_2 >, ..., < s_n, op_n >\}$ 發生的條件成立時，定義 $q_{dprice}(s, op)$ 為網路服務 *S* 上的功能 *op* 在發生關聯性情況下之價格，其中當 $dt = UD$ 時：$q_{dprice}(s, op) = q_{price}(s, op) * d, \forall < s, op > \in S$ ，當 $dt = FP$ 時：$q_{dprice}(s, op) = d / n, \forall < s, op > \in S$ 。

藉著重新定義價格品質條件的計算方式，可以明確地知道當發生關聯性時這些服務的價格該怎麼根據關聯性的種類來變動，以便在服務挑選期間，發生關聯性之時可以計算出正確的服務價格。

(四)網路服務組合架構

我們介紹了網路服務組合塑模工具—活動控制圖，針對活動控制圖的語意特性設計了權重計算，接著是所設計的服務品質模型以及服務關聯性模型等，這四個部分的關係如圖 4-1 模型架構圖所示。

我們先用活動控圖設計網路服務組合之樣板，接著每當有任何使用者要求呼叫這網路服務組合時，同時這個使用者會給予他對各個服務品質喜好的權重設定以及喜好之關聯性挑選策略，然後即時地計算出網路服務組合裡頭每個元件對整體服務所應具有的權重之後，並根據使用者的喜好以及使用者所喜好之關聯性挑選策略，利用服務品質的

計算找出最符合使用者所要求的特性之網路服務，例如最省錢或是執行速度最快等，最後組合成一個真正的網路服務組合，將之轉化為活動控制圖引擎可執行之格式，執行完之後將結果回傳給使用者。至於相關的服務挑選演算法以及關聯性挑選策略，將在下一節說明。

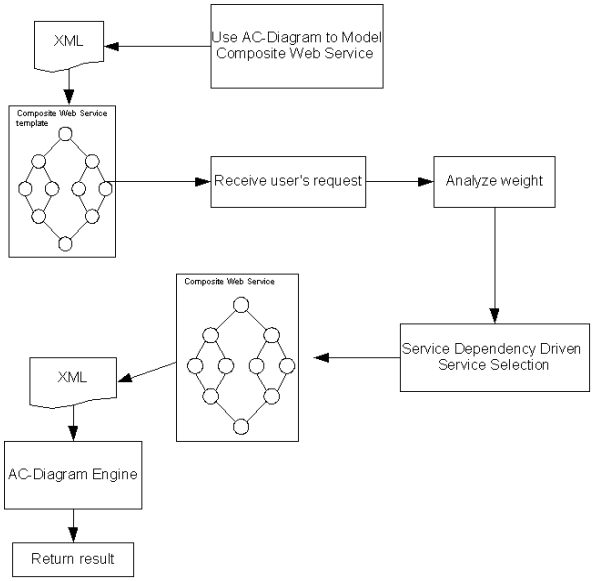

圖 4-1 模型架構圖

五、網路服務的選擇

(一) 目標設計與相關假設

我們的方法特色在於提供處理服務關聯性的機制，考量網路服務組合裡各自網路服務的權重而且兼顧全域最佳化的議題。以下列出希望達成的目標及使用的方法：

- 針對各服務被執行之不等機率而評估出對整體網路服務組合影響之權重。

透過對活動控制圖的分析，設計出一套權重計算的方法，每當有使用者要求使用網路服務組合時，便即時地算出各元件該佔有之權重，再將權重列入服務品質計算的考量之中。

- 能夠處理服務關聯性。

為了能夠處理服務關聯性，除了分析及了解關聯性之間是否會產生衝突之外，另外也設計了四種關聯性的挑選策略，讓使用者能夠根據自己的喜好來決定關聯性挑選的方式。

- 考量全域最佳化的概念。

提出了漸進最佳化的想法，分析有可能產生全域最佳化議題的平行執行工作流，並找出可改善服務品質之網路服務來替換。

此外定義一些研究假設假設，以釐清問題的範圍，以下列出相關的假設：

假設 1、語意相符之網路服務：

假設已經建立了完善的網路服務註冊機制，能夠準確查詢出所有網路服務組合裡各任務所對應語意相符之網路服務，供我們的演算法來從中挑選，本文所提的方法是建立在知識基礎已經建立完成的假設之上，所以不會發生任何語意不符合的情況。

假設 2、服務關聯性的註冊機制

假設已經建立良好之服務關聯性的註冊機制，為網路服務組合挑出一群候選的網路服務時，能夠同時查詢到這些對應至不同任務的網路服務之間，哪些是有服務關聯性存在及相關的特性或資料等。

(二) 區域最佳化

區域最佳化顧名思義即是針對各自的任務挑出一個網路服務，最後整合成網路服務組合，在挑選的過程中不會考量到其他任務的網路服務；至於如何從眾多網路服務中挑選出最適合使用者喜好的網路服務，就要靠服務品質分數之計算。

我們為每個網路服務計算出一個服務品質的總分，最後挑選出分數最高的網路服務。在本文中使用之前介紹的服務品質模型並假設 $S=\{s_1,s_2,...,s_n\}$ 為候選的網路服務集合，q_i 為網路服務 s_i 的服務品質向量，將所有網路服務的服務品質向量建立成一個二維的矩陣 Q 且 $Q=(Q_{i,j};1\le i\le n,1\le j\le 5)$，第 i 列為網路服務 s_i 的服務品質向量，因為只有五個服務品質條件，所以 j 必須大於等於 1，小於等於 5。並不是每種服務品質條件都是越大越好或是越小越好，而且不同的服務品質條件的數字範圍又不相同，所以我們必須對 Q 做轉向及一般化的動作，使得所有的值都能介於 0 到 1 之間，且值越大代表越好。

下面是轉向及一般化的公式，假如服務品質 j 是逆向性質，也就是越小越對使用者有利，例如執行時間，可利用下面的式子為其做轉向及一般化：

$$V_{i,j}=\frac{Q_{i,j}-Q_j^{\min}}{Q_j^{\max}-Q_j^{\min}} \; if \; Q_j^{\max}-Q_j^{\min}\neq 0 \text{ ，}$$

$$V_{i,j}=1 if \; Q_j^{\max}-Q_j^{\min}=0 \text{ 。}$$

完成上述的工作之後會得到一個矩陣 V，$V=(V_{i,j};1\le i\le n,1\le j\le 5)$，再根據使用者對這五個服務品質所提供的權重設定 W，W_j 代表對第 j 個服務品質的權重:

$$W_j\in[0,1] \text{ 且 } \sum_{j=1}^{5}W_j=1 \text{ ，}$$

接著使用下面的公式來計算出每個網路服務的服務品質分數：

$$Score(s_i) = \sum_{j=1}^{5} V_{i,j} * W_j \text{ 。}$$

其中 $Score(s_i)$ 代表網路服務 s_i 的區域服務品質分數，當所有網路服務的服務品質分數都計算出來之後，挑選其最高分來成為這個任務所需之網路服務。

(三) 服務關聯性衝突

不同的服務關聯性在同一個網路服務組合裡，可以混用也有會產生衝突的情況，在此我們使用了圖形的概念來解決這個問題，首先將每個關聯性都當成是圖形裡面的節點，假如兩個關聯性之間只要有一各自的網路服務是對應到網路服務組合裡相同的任務且對應到相同任務之兩個網路服務並不是同一個網路服務的話，那就稱這兩個關聯性有衝突關係。

為這兩個關聯性的節點加上連結，最後會得到一個服務關聯性的衝突圖形(Conflict Graph)。只要找出這個衝突圖形所有的極大獨立集合(Maximal Independent Set)，每一個極大獨立集合裡的所有關聯性便代表可以同時使用在同一個網路服務組合裡而不會導致有衝突發生，找出衝突圖形請參考表 5-1 極大獨立集合演算法。

表 5-1 極大獨立集合演算法

```
MIS( G )
 M ← ∅   // M is a set of set of vertex
for each vertex v ∈ V   // V is set of vertex
    do I ← V − {v}
       // Adj[v] is adjacency-list of vertex v
       for each u ∈ Adj[v]
           do I ← I − {u}
       if I = ∅
          then S ← {v}  // S is a set of vertex
               M ← M ∪ {S}
       else for each vertex i ∈ I
            S ← I   // S is a set of vertex
            for each u ∈ Adj[i]
                do S ← S − {u}
            S ← S ∪ {v}
            M ← M ∪ {S}
return M
```

(四) 關聯性挑選策略

解決關聯性衝突的問題後，接著便要決定關聯性挑選的方式，考量執行的效率，我們直接從所有關聯性極大獨立集合中挑選其一，並全部將關聯性代入網路服務組合裡面。在此設計了四種挑選策略，分別介紹如下：

■ 最多關聯性策略(Maximal Dependency Strategy)

從所有極大獨立集合當中，挑選出內含數量最多的項目，亦即關聯性最多的項目，代表在網路服務組合之中可能地使用最多的關聯性。

■ 最大覆蓋策略 (Maximal Coverage Strategy)

從所有極大獨立集合當中，挑選出所有關聯性能夠處理最多網路服務組合裡的任務之項目，如果能夠處理的任務數量一樣多的話，我們則會選擇關聯性數量的項目，意即完成最多任務中使用最少關聯性的項目。

■ 最多折扣策略 (Maximal Discount Strategy)

從所有極大獨立集合當中，挑選出所有關聯性都使用的情況之下，能節省最多價格之項目。作法是將所有極大獨立集合裡的關聯性都套用的情況之下，找出所有關聯性裡的網路服務各自節省下的價格乘上其對應的任務之權重後並加總。

■ 最大改善服務品質策略 (Maximal Improvement Strategy)

在所有極大獨立集合當中，挑選出所有關聯性都使用的情況之下，能夠改善區域最佳化最多服務品質之項目，做法是針對每個極大獨立集合中所有的關聯性都使用的情況之下，求出其改善區域服務品質乘上相對應的任務之權重並加總，跟最大節省策略的作法幾乎一樣，只是從價格上的考量變為所有服務品質的考量。

當使用關聯性挑選策略選擇出服務關聯性並對應至網路服務組合後，可能無法兼顧到所有的任務，因為一個網路服務組合裡所有的任務不太可能完全被包含在所有的關聯性之下，也就是說扣掉使用關聯性決定的網路服務，還會有許多網路服務組合裡頭的任務尚未被決定對應的網路服務，因此採用區域最佳化來處裡這些尚待挑選的任務，以找出區域內最適當的網路服務。

(五) 全域最佳化改善服務品質

當使用關聯性選擇策略挑選出所要使用之關聯性之後，也用區域最佳化來補上其他空著的任務所對應之網路服務後，還可以再針對 Zeng[3]所提出因為工作流平行執行而使得區域最佳化無法成為全域最佳化的問題做改善。

改善的方式根據現實的情況來選擇要不要對關聯性做過濾，接著對已過濾後的關聯性或是沒過濾的關聯性做衝突上的判斷，然後根據使用者喜好的策略來做關聯性的挑選，再來是利用區域最佳化找到網路服務，以滿足剩餘沒被關聯性包含到的任務，最後可以選擇要不要針對全域最佳化來做漸進式的改善。

六、系統實作與效能分析

(一) 發展環境介紹

我們選擇 Java 程式語言作為系統開發工具，實驗中採用 AMPL(A Modeling Language for Mathematical Programming)[9]來實作使用整數線性規劃之全域最佳化網路服務挑選。AMPL 為數學問題塑模軟體，可搭配其他 Solver 處理如線性規劃、整數線性規劃以及其他的排程問題，本研究的實驗是在 Pentium 4 3.0 GHZ、512MB RAM，作業系統為 Gentoo Linux 2006 上執行。

(二) 實驗假設

我們使用了圖 6-1.旅遊服務圖所示之旅遊服務來做實驗，實驗中設計讓每個任務能夠挑選的網路服務從 5 個成長到 50 個，接著設計訂房和訂機票服務，以及訂機票與租車服務會產生關聯性，關聯性數量的期望值為每個任務可挑選之網路服務數乘上 0.1，且這兩種關聯性都設定為固定折扣，折扣為八折，除此之外關聯性會固定在區域最佳化服務品質分數排名前 75%的範圍內發生，藉此模擬現實世界中有關聯性的情況下。

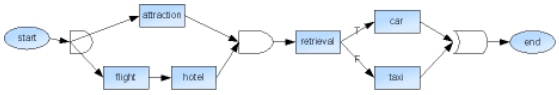

圖 6-1 旅遊服務圖

(三) 服務品質分數分析比較

首先先比較使用不同策略挑選關聯性及使用整數線性規劃之全域最佳化和區域最佳化，討論這幾種方法所找出的網路服務在服務品質上的分數表現。圖 6-2.QoS 分數比較圖 1，比較結果為四個關聯性挑選策略在不做過濾但是使用了漸進最佳化的情況下，其中最多關聯性、最大覆蓋、最多折扣的表現初期都比整數線性規劃來得好，但是到後期因為可選擇的網路服務越來越多，所以這三種比較屬於啟發式的挑選方法表現變得比整數線性規畫差，而最大改善策略則是一直表現得比所有的策略以及其他的方法好。

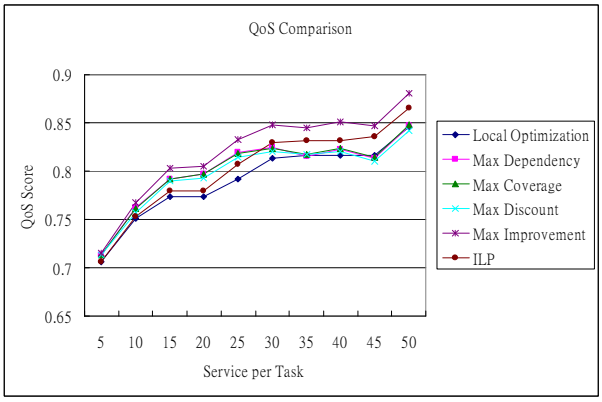

圖 6-2. QoS 分數比較圖 1

圖 6-3. QoS 分數比較圖 2，為所有的關聯性加上了區域最佳化過濾的動作所得出的結果，本來三種策略到後期會表現得比整數線性規劃還差，但是加上過濾的動作之後，到後期仍然能跟整數線性規劃表現得差不多，而最大改善策略則是一直表現的很好，過濾動作似乎對此策略沒什麼太大的改善。

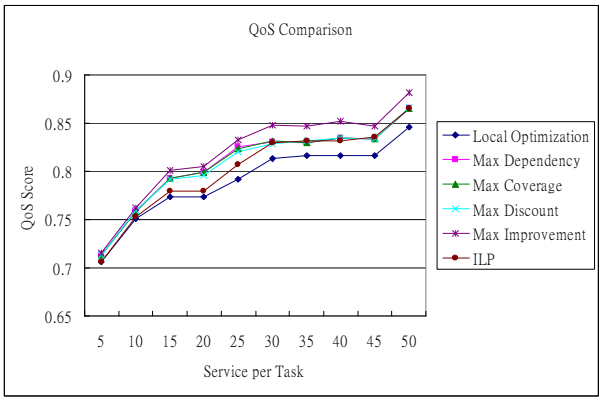

圖 6-3. QoS 分數比較圖 2

(三) 演算法效能評估

此小節將比較演算法的效能，分別比較區域最佳化、全域最佳化以及本文所提出的方法，實驗環境延續上一小節的設定，但是每個任務能夠挑選之網路服務則是從 12 成長到 60。

圖 6-4. 演算法效能比較圖 1，，實驗結果發現本文提的方法除了能夠挑選服務關聯性之外，效能的表現上也比使用整數線性規劃之全域最佳化來得好。另外值得注意的當為四種策略加上過濾的動作之後，反而會比沒有過濾來得快，那是因為過濾後關聯性變少了，所以在計算所有極大獨立集合時省下的時間比過濾的時間還要多，因此效能表現比較好；另外在四種策略之中，因為最大改善服務品質的策略要對所有的極大獨立集合做服務品質分數上詳細的比較，所以跟其他的策略比起來，很明顯的比較慢。

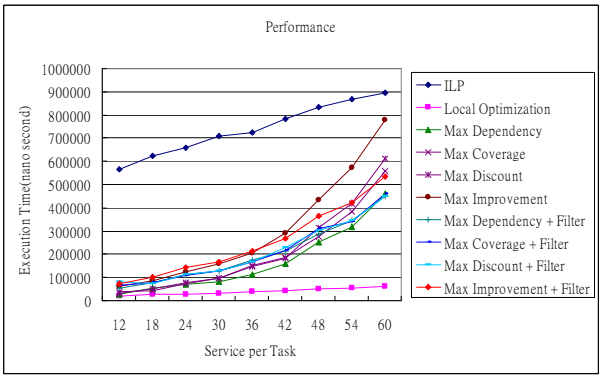

圖 6-4. 演算法效能比較圖 1

從現實世界的觀點來看，關聯性要發生變動的頻率不大，每個關聯性都是維持一段時間之後才會有所改變，所以每次在使用者要求網路服務組合時才即時地計算極大獨立集合是不適當的，極大獨立集合的問題應該先被預先處理，當關聯性有變動的時候才再重新算過。

圖 6-5. 演算法效能比較圖 2，預先處理極大獨立集合之後的結果，沒有做過濾的關聯性挑選策略效能非常的好，處理速度非常接近區域最佳化，甚

至最多關聯性策略還有發生比區域最佳化還快的情況;可是有過濾動作的關聯性挑選策略卻因為要即時的過濾，不能預先處理關聯性衝突的問題，所以在效能的表現上便比沒過濾但有預先處理極大獨立集合問題的策略差，不過仍然比使用整數線性規劃的全域最佳化要來得快很多。

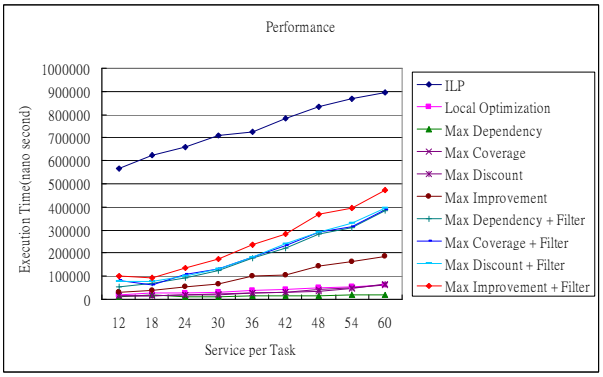

圖 6-5. 演算法效能比較圖 2

(四) 關聯性挑選策略分析比較

在此針對本文所提出的四種關聯性挑選策略，用實驗的方式來找出不同的影響之下，何種策略表現較佳，以及各種參數對關聯性的影響。首先我們先設計了三個實驗，利用這三個實驗來分析出各策略的特性。

實驗一：在這個實驗中設定小型的服務關聯性在價格上之折扣會比大型的服務關聯性來得高，亦即關聯性所包含之網路服務越多折扣率越差。所有的服務關聯性會在區域最佳化排名過後之前 75% 的地方發生，用意是在模擬現實世界中有關聯性的情況大部分都會比沒關聯性來得好。

圖 6-6. 關聯性挑選策略比較圖 1，實驗結果顯示最多關聯性及最多折扣表現的比最大覆蓋要來得好，服務品質的分數都比最大覆蓋來得高，由此可知在小型關聯性折扣較佳以及大部分關聯性都能比單獨服務好的情境之下，最多關聯性及最多折扣找出的關聯性會比最大覆蓋好，而最大改善服務品質之策略永遠都會比其他三個策略來得好。

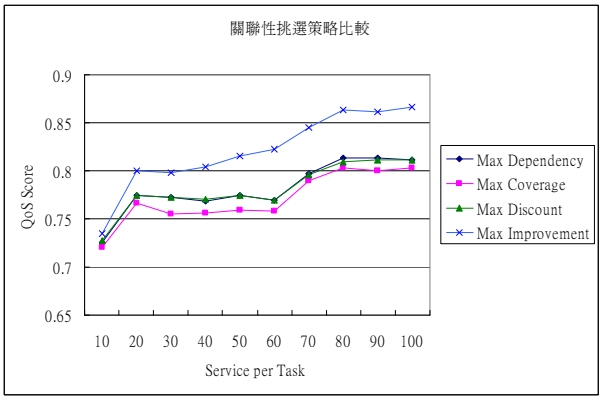

圖 6-6. 關聯性挑選策略比較圖 1

實驗二：在這個實驗中，設定跟上一個實驗相反的情況，越大型的關聯性折扣會越好，亦即關聯性包含的網路服務越多，折扣率就越高。

圖 6-7. 關聯性挑選策略比較圖 2，實驗結果顯示最大覆蓋跟最多折扣表現的比最多關聯性策略來得好。由此可知在越大型關聯性越有利以及大部分關聯性都還不錯的情況之下，最大覆蓋跟最多折扣都會表現得比最多關聯性好，而最大改善服務品質一直會都有很不錯的表現。

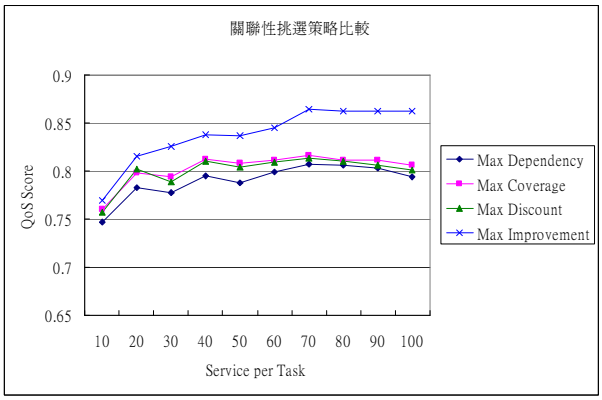

圖 6-7. 關聯性挑選策略比較圖 2

實驗三：在此實驗中，固定每個任務對應到的網路服務各為 100 個且所有的關聯性無論包含多少網路服務，折扣率都差不多。但是關聯性發生的範圍會改變，先用區域最佳化排名之後，從原先的 100%一直拉大到前 20%，讓發生關聯性的服務從可能大部分會比單獨服務差到最後可能大部分的關聯性會比單獨的服務來的來。

圖 6-8. 關聯性挑選策略比較圖 3，實驗結果一開始關聯性較差的時候，最大折扣策略表現得比較差，當隨著關聯性可能會越來越好的時候，最多折扣的表現跟最多關聯以及最大覆蓋互有輸贏，由此實驗可知當不能確定關聯是否保證一定會有優勢的時候，最大折扣策略並不適用。

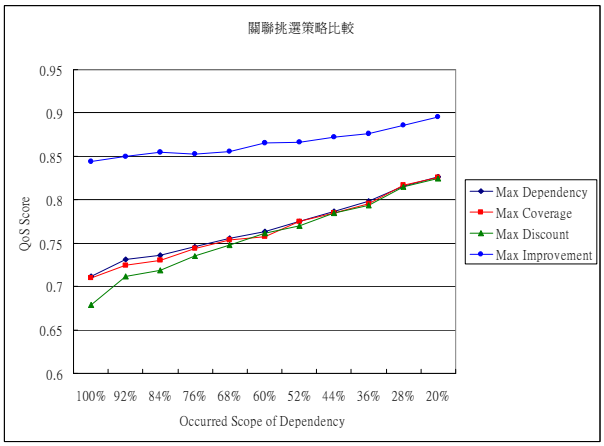

圖 6-8.關聯性挑選策略比較圖 3

(五) 最佳化之正確性測試與效能評估

在此將針對漸進式最佳化做測試，為了能夠更明確地看出漸進式最佳化的效果，在這個實驗之中不考慮服務關聯性，只針對區域最佳解的結果做漸進式最佳化，然後跟區域最佳解以及全域最佳解的結果來做比較。因為漸進式最佳化是屬於啟發式的演算法，並不保證一定會是最佳解，所以我們預測

漸進式最佳化所求出之服務品質的分數應該大於等於區域最佳解，以及小於等於全域最佳解。

圖 6-9.全域最佳化 QoS 比較圖是服務品質方面的實驗結果，可以看到大部分的情況漸近式最佳化都能找出跟全域最佳化同樣的結果，而且範圍也在我們預測的界於區域最佳化以及全域最佳化的中間，證明漸進最佳化理論是正確的。

圖 6-10. 全域最佳化效能比較圖，因為我們所使用的 AMPL 版本有變數數量上的限制，無法處理超過 300 個變數，所以差不多只能測到每個任務各 60 個左右的網路服務，而且 AMPL 只能測量到千分之一秒的單位，在時間測量上不是很準確，所以測試出來的時間不是很準確而且數據的震盪很大，但是由實驗中可以觀察到漸近最佳化和區域最佳化會比整數線性規劃的全域最佳化要來得要有效率多了。

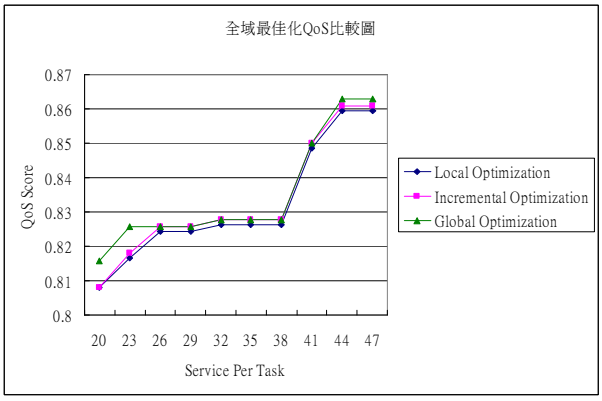

圖 6-9. 全域最佳化 QoS 比較圖

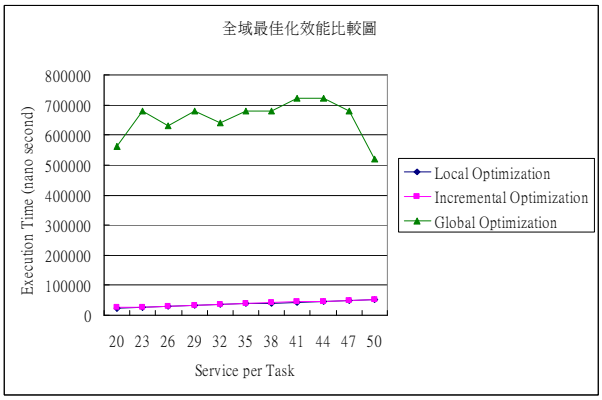

圖 6-10.全域最佳化效能比較圖

七、結論與未來展望

本文的研究目標在於探討服務關聯性所引起之服務品質的改變，並利用服務關聯性帶來的好處，來幫助挑選適當的網路服務，組合現有的網路服務成為全新且功能更強的網路服務組合，除此之外，還要能夠兼顧網路服務組合裡的網路服務被執行之可能性，因為被執行的可能性不相同會對服務品質造成不同的影響程度。研究中，利用圖形理論的極大獨立集合的概念來解決關聯性可能會產生衝突的情況，提出四種關聯性挑選策略使得網路服務組合的服務品質能夠更好。針對工作流互斥執行的可能性，分析了活動控制圖的語意特性，並使用權重的概念來評估所有基本網路服務對整體服務品質的影響程度，解決了之前相關研究在這方面的缺失。在全域最佳化方面，我們也分析了活動控制圖的特性，找出可能造成全域最佳化比區域最佳化更好的情況，並使用漸進最佳化的方法來改善服務品質，使其能夠有效率地逼近全域最佳化的結果，不使用的暴力法或是效率不佳的整數線性規劃。

經由這一次的研究，發覺到值得進一步研究的議題，包括：定義更完整的關聯性類型，找尋各種服務品質相關的關聯性，設計更有效率的極大獨立集合演算法，並針對特定結構之工作流進行更有效率之網路服務挑選及建構完善之網路服務架構。

八、參考文獻

[1] Wu, S.-y. and K.-C. Lin, “Cross Enterprise Business Modeling with AC Diagrams and Workflow Patterns.” *Proceedings of the Seventh IEEE International Conference on E-Commerce Technology (CEC 2005)*, pp. 110-117, July 2005.

[2] Liu, Yutu, Anne H. Ngu and Liang Z. Zeng, “QoS Computation and Policing in Dynamic Web Service Selection.” *Proceedings of the 13th International World Wide Web Conference on Alternate track papers & posters*, pp. 66-73, May 2004.

[3] Zeng, Liangzhao, Boualem Benatallah, Anne H.H. Ngu, Marlon Dumas, Jayant Kalagnanam and Henry Chang, “QoS-Aware Middleware for Web Services Composition.” *IEEE Transactions on Software Engineering*, **30**(5):311-327, May 2004.

[4] WfMC(Workflow Management Coalition). http://www.wfmc.org/.

[5] Canfora, Gerardo, Massimiliano Di Penta, Raffaele Esposito and Maria Luisa Villani, “An Approach for QoS-aware Service Composition based on Genetic Algorithms.” *Proceedings of the Conference on Genetic and Evolutionary Computation (GECCO '05)*, pp. 1069-1075, 2005.

[6] Business Process Execution Language for Web Services. http://www-128.ibm.com/developerworks/webservices/library/specification/ws-bpel/.

[7] BPML. http://bpmi.org/.

[8] Aalst, Wil M.P. van der, Marlon Dumas and Arthur H.M. ter Hofstede, “Web Service Composition Languages: Old Wine in New Bottles?” *Proceedings of the 9th Euromicro Conference*, pp. 298-305, Sept. 2003.

[9] AMPL(A Modeling Language for Mathematical Programming). http://www.ampl.com/.

[10] Benatallah, Boualem, Quan Z. Sheng, and Marlon Dumas, “The Self-Serv Environment for Web Services Composition.” *IEEE Internet Computing*, **7**(1):40-48, Jan.-Feb. 2003.