國立交通大學

資訊學院 資訊學程

碩士論文

網站內網頁之區塊等級分析

Block-level Ranking for Intra-Website Pages

研 究 生：姚文鋒

指導教授：吳毅成 博士

中華民國九十六年六月

網站內網頁之區塊等級分析

Block-level Ranking for Intra-Website Pages

研 究 生：姚文鋒　　Student: Wen-Feng Yao

指導教授：吳毅成 博士　　Advisor: Dr. I-Chen Wu

國 立 交 通 大 學

資訊學院 資訊學程

碩 士 論 文

A Thesis

Submitted to College of Computer Science

National Chiao-Tung University

in partial Fulfillment of the Requirements

for the Degree of Master of Science

in

Computer Science

June 2007

Hsinchu, Taiwan, Republic of China

中華民國九十六年六月

網站內網頁之區塊等級分析

學生：姚文鋒　　指導教授：吳毅成 博士

國 立 交 通 大 學　　資 訊 學 院　　資 訊 學 程 碩 士 班

摘　　要

依據統計資料，截自 2007 年 6 月為止全世界的網頁數量有超過 140 億個之多，面對這樣龐大的資料庫，如何有效地使用是一件很重要的事。對於未知路徑的資料，通常尋求搜尋引擎的協助來正確定位資料；對於已知路徑的資料，為了增加使用效率，則會使用資料萃取的技術。

本實驗室所開發的 BODE (Browser Oriented Data Extraction)系統即是一套網頁資料萃取系統，使用者可以透過人性化的操作介面點選所要萃取的資料，再由系統產生萃取所需的腳本(BODE script)，並進行萃取的動作。

然而在建構 BODE script 的過程中，使用者必須要對 BODE script 語法、XPath 及 HTML Tag 有一定程度的了解才能順利進行。因此為了降低 BODE 系統的使用門檻，本論文提出了自動辨識單一網站內有用資料區塊的演算法，以便協助達成自動建立 BODE script 的目標。

Block-level Ranking for Intra-Website Pages

Student: Wen-Feng Yao Advisor: Dr. I-Chen Wu

Degree Program of Computer Science
National Chiao Tung University

ABSTRACT

According to the statistical data, there are more than 14 billion web pages in whole world by June of 2007. It's a important thing that how to use this huge database efficiently. For the information that we do not know its location, we usually use search engines to help us to find it out. And for the information that we do know where it is, we use data extraction to increase the efficiency.

BODE (Browser Oriented Data Extraction), developed by our laboratory, is such a web data extraction system. Its GUI can be used to indicate the data they want to retrieve, and the system will generate the BODE script that is used in the extraction process, and then start to extract.

However, people must have the basic knowledge about the syntax of BODE script, XPath and HTML Tag to build the BODE script. To reduce the threshold of using BODE system, this thesis proposes an algorithm to distinguish the useful information blocks from a single web site, so as to accomplish the goal of automatically generating BODE script.

誌　　謝

首先要感謝我的指導教授，吳毅成博士，由於他的細心指導與協助，本論文才得以順利完成。

另外也要感謝陳隆彬學長、蘇瑞元學長，總在每次的討論之中，給我正確的方向。再來就是實驗室BODE組的各位伙伴，有你們的加油打氣，我才能一路堅持至此，謝謝你們。

最後要感謝的就是我的爸媽，為了這篇論文，犧牲了很多陪伴他們的時間。他們不曾抱怨、始終要我以學業為重，並在我低潮時給我最大的支持與鼓勵，讓我可以安心的做研究、完成學業。謹以此論文，獻給我最愛的爸媽。

目 錄

表 列

圖 列

第一章

緒論

1.1 研究背景及動機

隨著網際網路各項服務及應用的蓬勃發展，網路上的資料量也隨之以爆炸性的速度在成長，依據WorldWideWebSize.com[2]的統計，截至2007年6月14日為止，全世界大約有140億的網頁被搜尋引擎所索引到，也就是全世界公開的網頁至少有140億個以上。

為了有效地利用這個龐大的資料庫，各種相關的研究便因應而生，其中與一般人最息息相關的莫過於搜尋引擎了。完整的搜尋引擎幾乎擁有全世界開放的網頁資料，且網頁的異動在幾天內也會隨之更新，例如Google[3]，只要一個新的網頁放上網際網路，在一個禮拜內大多可以透過關鍵字的搜尋而找到。因此搜尋引擎適合在未知的來源中尋找所需的資料，使用者只要輸入適當的關鍵字，很容易透過它來得到相關的資訊。

另外一個相關的應用是網頁資料萃取系統，使用者可以把要抓取的網頁資料事先設定，然後定時由系統代理使用者去取回所想要的資料，並存在本地端，可以免去枯燥的等待時間。本實驗室所開發的BODE(Browser-Oriented Data Extraction System)[1]即是此類的系統，使用者可以透過圖形化介面的輔助，以滑鼠點按所要擷取的資料區塊，編製出萃取所需的BODE Script，BODE便會依此Script所設定的資料路徑(XPath)至該網頁中逐一將資料轉存成XML檔至本地端電腦中，以便使用者進行離線瀏覽，或是將資料做進一步的應用。

這種個人化的需求，也有更進一步應用的網路服務，例如產品的比價(ex: Google Product Search[4])、學術論文的搜尋(ex: Scholar[5])…等，這些應用也都是基於網頁資料萃取的技術。但是網頁資料的萃取，始終需要人力的介入，如撰寫所需的腳本或是設定

所要擷取的資料區塊等動作，因而無法達到自動化，在實用上便有一定的限制。例如BODE系統進行資料萃取所需的BODE Script需要人工建立，雖然系統已提供相當人性化的圖型介面供使用者以點選的方式進行所要萃取資料的標記作業，但若要進行較為複雜的萃取作業，例如迴圈或條件式判斷，則使用者必須對BODE Script的語法、XPath及Html的語法有一定程度的了解才有辦法編製。因此，若能將網頁資料萃取過程中人力的介入降至最低，一定可以提高實用性，相信更多樣化的Client端應用亦會隨之而來。

所以，本論文的目的即在提出一演算法，可以將單一網站內的網頁切割成資料區塊，並將網站中各頁面相同的區塊(如導覽用的目錄區塊)進行合併，最後計算出各區塊的重要性，以便讓網頁資料萃取系統可以利用此結果，減少人力介入來編寫萃取腳本，進而達到自動化的目的。

1.2 論文內容概述及大綱

本論文的目的在辨識單一網站頁面中屬於重要的內容及導覽區塊，所以明顯地，我們需要一個適當的頁面區塊分割演算法。我們採用馬維英博士團隊所提出的VIPS[6]來做為我們分割區塊的演算法，因為此演算法採用頁面呈現上的一些視覺線索來做為分割依據，可有效切割出具有不同意義的區塊，例如導覽與內文...等。

我們打算由網站作者的角度來試圖分析其所呈現出內容的重要性差異，所以我們採用了網頁重要性分析的鏈結分析演算法的HITS[7]來計算其區塊重要性。由於HITS不適用於單一網站，亦僅能分析頁面重要性，因此我們對HITS做了調整。不同於傳統鏈結分析，我們不計算網站外連結(out-link)，反過來計算網站內連結(in-link)，並且試圖合併各頁面都有的相同區塊(可能是導覽、LOGO、宣告或廣告)。因為這些區塊就作者及使用者的角度而言，不管出現在哪一頁，其意義都相同。合併的動作，有助於導覽區塊Hub值及主要內容區塊的Authority值的合理性。

除此之外，由於Hub及Authority區塊判斷的準確與否，會直接影響到我們演算法的結果，因此我們依區塊的幾何資訊及內容的型態(圖文及鏈結的組成狀況)給定權重帶入HITS的演算法中。

本論文共分成五個章節，第一章說明研究的動機及背景，並簡述論文內容，第二章回顧關於此領域的相關研究，第三章介紹我們所提出的辨識單一網站內各區塊重要性演

算法，第四章則以實驗來驗證所提出演算法的正確性，第五章則提到相關可能的應用、本演算法仍可改善及擴充的地方，及本論文之總結。

第二章

相關研究

上一章提到我們的目的是分析頁面中區塊的重要性，這其牽涉到兩個領域的研究，分別是網頁區塊分割及鏈結分析。在這一章我們將分別介紹這兩個領域的相關研究及其應用。

2.1 網頁區塊分割

早期World Wide Web的相關應用，如Web Information Retrieval及Web Data Extraction，都將網頁視為基礎單元。但實際上網頁通常都是由不同的主題區塊所組成，例如一般網頁大多含有網站共同的頁首、頁尾、導覽列，以及與該頁面主題相關的內容區塊。所以這些應用後續的研究也開始將基礎單元降階至區塊等級，以求更準確的結果，因此網頁區塊的分割技術便隨之發展。

最早期的文件分割技術是運用在純文字檔上，因為文字檔沒有結構可言，因此採用的是Fixed-Window(固定長度)的方式，好處是快速、簡單，壞處是分割出來的資料區塊不見得合適。

因為網頁是屬於半結構化的文件，Fixed-Window的方式並不適用，所以便開始有依DOM-Tree做為分割依據的演算法出現。此類演算法試著要透過DOM-Tree來擷取出結構化資料，但HTML畢竟只是資料呈現的描述語言，而不是像XML是一種結構化的語意描述語言，因此無法準確地將網頁按照語意適當的分割。

目前精確度較高的網頁區塊分割技術是一種基於視覺上線索來執行分割的演算法。很明顯的，網頁上的資料通常不僅僅一個主題，而特定用途的區塊通常位於某些固定的位置，不同的主題區塊間也都會有一些明顯的分隔。VIPS即是利用此視覺上的線索

做為分割依據的網頁區塊分割演算法，可以較為正確切割出符合網頁不同語意的區塊，因此我們選擇了VIPS做為我們網頁區塊分割的演算法。

VIPS演算法分成三個主要步驟，一是依HTML DOM-Tree將頁面分割為適當的區塊；二是找出這些區塊在幾何位置上的水平或垂直分隔線；三是依據找到的分隔線，將頁面分割成若干個區塊並計算區塊的內容相關程度(DoC)，檢查每個區塊的DoC是否符合設定的門檻值？若不符合，則將該區塊當成頁面，重覆步驟一至三的動作，直至符合為止。

圖2.1為VIPS的示意圖，假設頁面的Layout如圖中所示，第一個Round會切出最上方的區塊，而下方三個區塊所組成的大區塊則因為DoC值未達設定值，進入第二個Round。第二個Round則會把最左邊的區塊切出來，相同的，右下方的二個區塊。因為DoC值未達設定的門檻值，進入第三個Round，最後這兩個區塊也被切割出來。

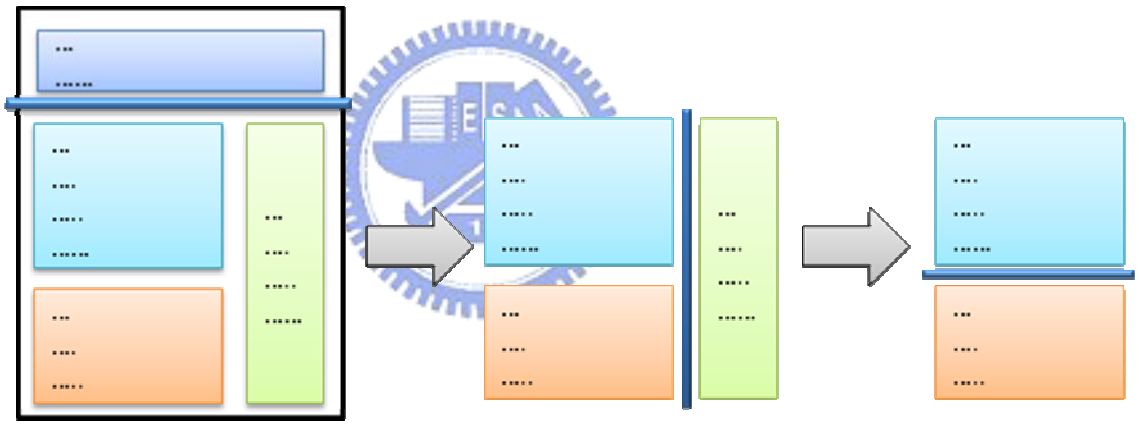


圖2.1 VIPS演算法示意圖

步驟一依HTML DOM-Tree分割區塊是採用啟發式(Heuristic)的方法，依給定的12條規則，由HTML Tag <BODY>開始依著DOM-Tree往下做深先 (depth first) 的巡覽，每遇到一個節點即依序比對這12條規則來決定該節點是否分割，以及是否繼續往下一個節點比對，如此直至整棵樹走完，即可得出該回合所分割出的區塊。

步驟二則依序將步驟一所得到的區塊逐一放入該頁面大小的空間之中，依區塊與頁面空間的重疊情況執行分割、更新及刪除三種動作。如圖2.2所示，先將最上方的區塊1放入，執行分割得到S1及S2；再將區塊2放入，一樣執行分割得到S1、S2及S3；再將區

塊3放入，又再執行一次分割得到S1、S2、S3及S4；最後將區塊4放入，執行刪除S3，並更新S2的邊界，得到S1、S2及S3；最後再將最上方及最下方的分隔線S1、S4刪除，剩下來的分隔線S2即是本回合所得到的分隔線。

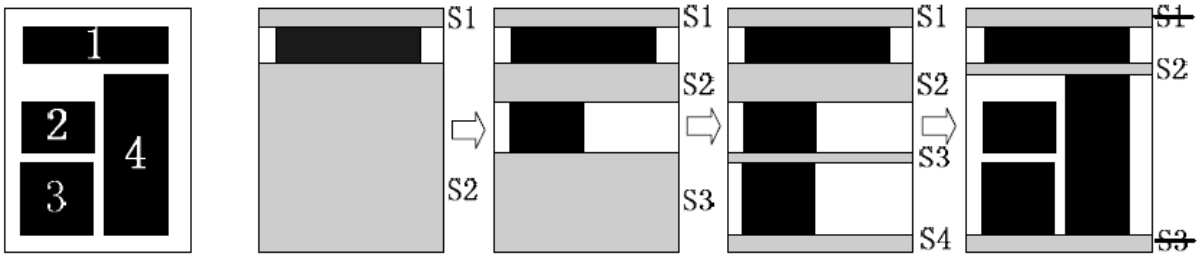


圖2.2 簡單的頁面分隔線偵測流程圖

資料來源：[6] VIPS

步驟三先依步驟二所得出的分隔線建立頁面區塊結構，再設定分隔線的權重。分隔線的權重是依視覺上的差異做為衡量的標準，原作者同樣有列出參考的規則。例如水平分隔線高比較大的，權重比高比較小的來得大；又如分隔線上面字體小，下面字體大的權重也比上下字體一樣大的來得大。最後只保留權重最大的分隔線為此回合切割區塊的依據，再判斷所分割的區塊是否要再進行下一個回合的分割步驟。

2.2 鏈結分析

我們常用的搜尋引擎所使用的演算法即是鏈結分析，像Google所使用的是名為PageRank[8]的鏈結分析演算法，除了PageRank之外， HITS也是類似的演算法。他們主要利用網頁間的鏈結關係來分析頁面的重要性程度，其假設頁面A有鏈結連至頁面B，即代表頁面A對頁面B的重要性的一種認同投票，因此若一頁面被越多的頁面所連結，則其重要性就越高。

PageRank把所有網頁的鏈結關係建立成一*N*x*N*的鄰接矩陣A，N為頁面的數量，若*A(i,j)=1/k*，表示頁面i有超鏈結連至頁面*j*，其中*k*為頁面的超鏈結數量；反之若*A(i,j)=0*，則表示頁面*i*與頁面*j*並無任何鏈結關係。PageRank利用Eigenvector *(AX=λX)*來求解，其中*A*為上面提到的鄰接矩陣，*X*則為所有頁面的PageRank值的陣列。

HITS做法不同於PageRank僅計算頁面主題內容的重要性 (Authority)，更加計了導覽用途的重要性 (Hub)，也不以整個網路所有頁面來做為計算的基礎，而是取搜尋引擎的前50個頁面來做為Root Set，再由這些頁面的超鏈結擴張至適當的頁數(該篇論文的數據為200頁)。最後依擴張後的頁面來按下面的公式計算其Authority及Hub的值。

$$A = M^T H$$
$$H = MA \qquad (1)$$

PageRank的概念就如同圖2.3所呈現的意義，其中箭頭代表的是網頁中鏈結的方向，例如ID=2的頁面有一個鏈結至ID=1的頁面，另有三個指向他的鏈結來自於ID=1,3,4的頁面。因為這是一個封閉的環境，最後一定會達到一個平衡的狀態，每個頁面本身的重要性，會等於所有指向他的鏈結的重要性的加總，而他本身的重要性又會透過頁面內指向其他頁面的鏈結分送給所指向的頁面。

HITS的概念也相似，只是把重要性分成了Authority與Hub兩個值，頁面的Authority值等於所有指向他的鏈結所分到的頁面Hub值的加總；而頁面的Hub值則等於所有他所鏈結到的頁面所分到的Authority值的加總，最後一樣會達到平衡的狀態。

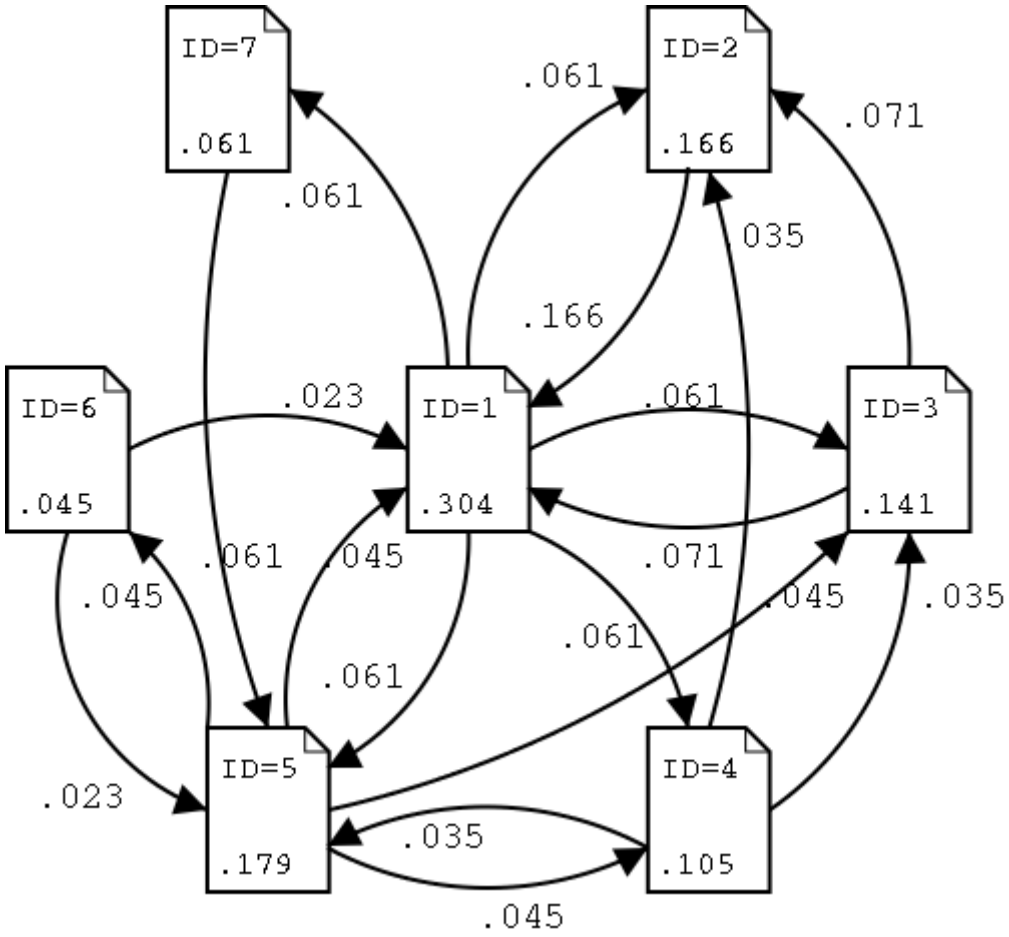


圖2.3 Link Analysis計算方法示意圖

2.3 區塊層級的鏈結分析

由於網路資訊的爆炸性成長，網頁所呈現的資料越來越多樣化，因此除了量會是搜尋引擎的效能主要影響外，準確性也成為另一個問題。例如我們在Google以"資料萃取"為關鍵字查詢，符合的結果筆數約有426,000筆，而其中有部份是屬於不相關的資料或是相關但與使用者的目的不相符的頁面資料。要使用者在這堆茫茫的資料海中逐筆尋找所需要的資料，無疑是相當耗時且另人不悅的過程。

造成這結果的原因除了網頁資料量成長的速度太快之外，另一個主要的原因是目前搜尋引擎無法有效地過濾非內容區塊中的雜訊。因此區塊層級的鏈結分析演算法也開始被提出來試圖解決這樣的問題，如馬維英博士團隊的Block-level Link Analysis[9]。此演算法可以有效的降低PageRank與HITS中會出現的Topic-Drift現象(指結果與搜尋的主題產生偏差)，原因如圖2.4中第一個頁面中兩個鏈結對意義可能不同。例如左邊的區塊

是廣告區塊，若沒有分開計算這兩個鏈結的重要性的話，會導致結果的準確度降低，上面的頁面應該是不重要的，但卻被計算成跟下面的頁面一樣重要。

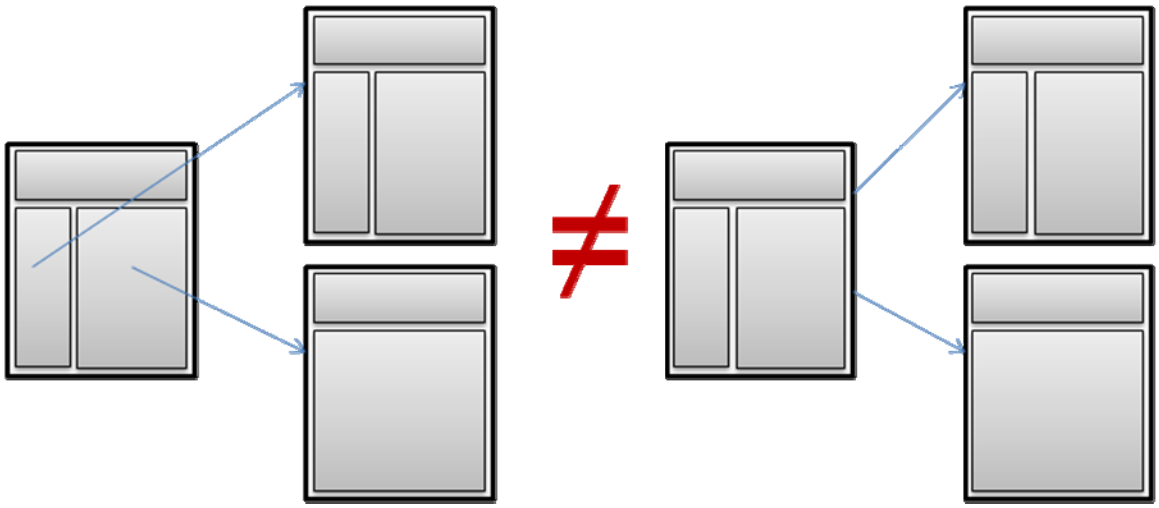


圖2.4 區塊對頁面的鏈結關係與頁面對頁面的鏈結關係

Block-level Link Analysis雖然已經把鏈結關係由頁面對頁面降階至區塊對頁面，但由於主要是運用對象是搜尋引擎，因此其最終產出仍是頁面，相對於Web Information Retrieval及Data Extraction卻不適用。因為這兩種應用需要清楚標示出頁面中要擷取的區塊位置，而不僅僅是重要的頁面而已，所以我們再將鏈結關係由區塊對頁面降階至區塊對區塊。

第三章

演算法介紹

在這個章節中將介紹我們所提出的基於合併的區塊層級鏈結分析演算法，主要是改良HITS演算法。為了符合我們的研究目的—辨識單一網站中重要區塊的位置，我們加上了區塊分割、區塊合併、區塊權重調整及修改了原HITS演算法以符合區塊層級的計算需求，我們的演算法流程如圖3.1所示。

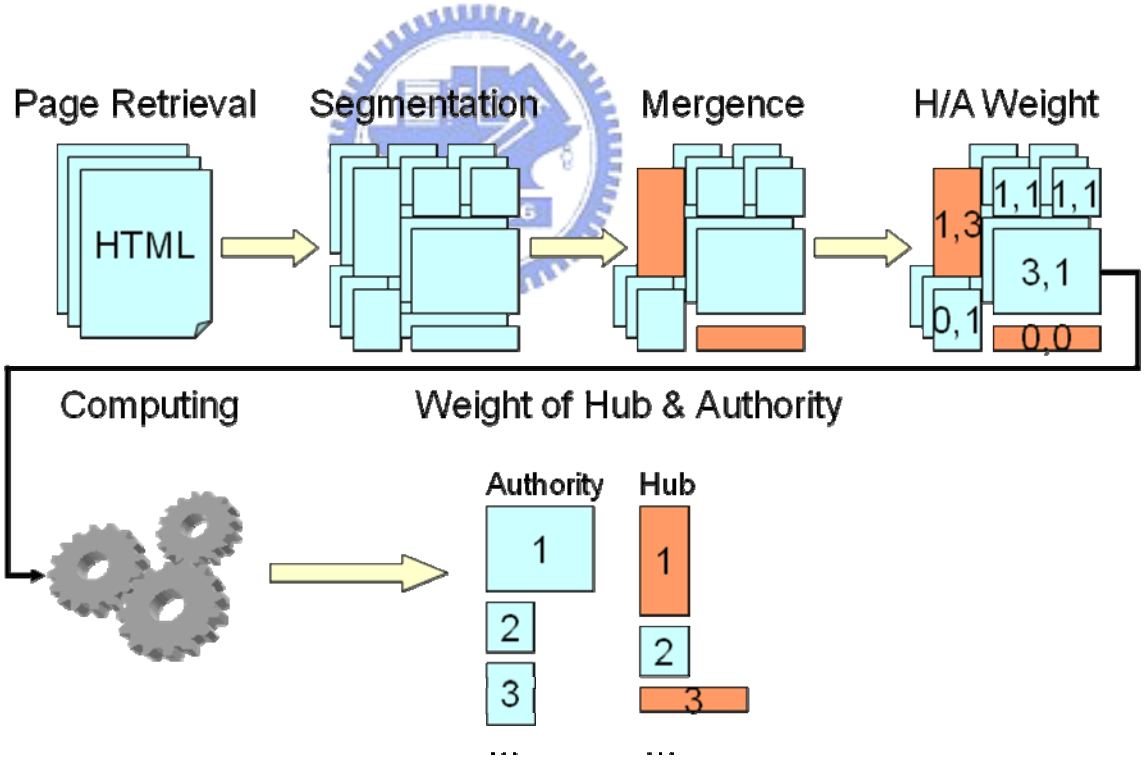


圖3.1 本論文演算法流程

以下各節分別就這幾個項目來做介紹。

3.1 區塊合併及鄰接矩陣之建立

3.1.1 區塊合併

首先我們先解釋為什麼要進行區塊合併？主要原因有二，一是目前網站大多依固定的範本建立而來，因此每頁都會有些特定的部份是相同的，例如頁首及頁尾等網站Logo、登入資訊及版權、隱私權宣告等。再者用來導覽用的區塊，在同一層目錄中也應是相同的，甚至在不同層也有可能相同。這些重複出現的區塊，不管是對網站的作者或是使用者而言，在每一頁所代表的意義應是相同的，因此我們應該將之視為同一區塊來做計算。

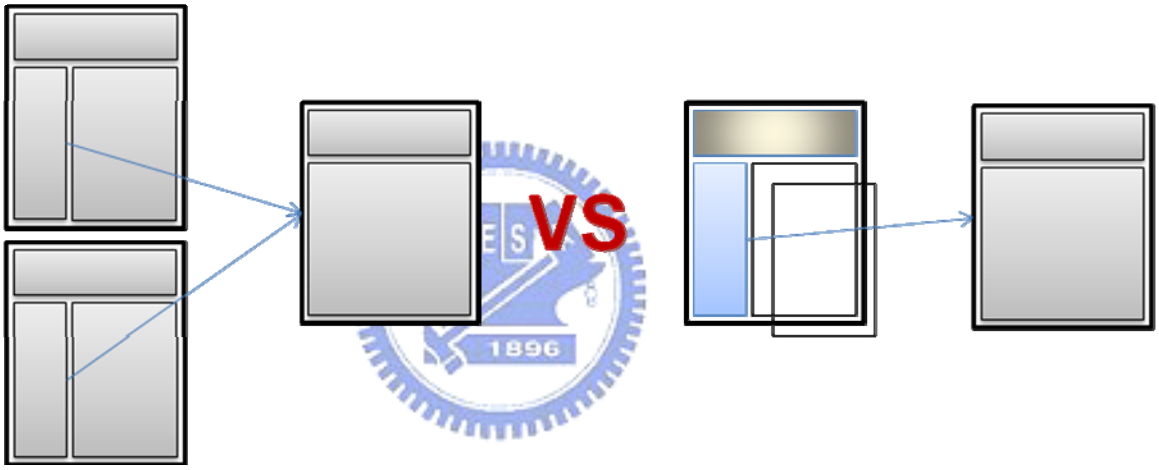


圖3.2 區塊合併與否的意義差別

原因二則是因為我們所採用的鏈結分析演算法，並不適用於單一網站內的鏈結分析，因為網站內部的鏈結僅是導覽用途，不如跨網站的鏈結來得具有重要性認同的意義。因此在不做合併的情況下直接套用該演算法，會因為幾乎每個頁面都有的導覽或回首頁等的區塊，造成被這些鏈結所指向的區塊重要性被過份放大而影響整個演算法的準確性。

在瞭解區塊合併的原因及意義後，我們接下來看如何進行區塊合併。

合併的原則是，屬於Hub型態的區塊能合併的儘量合併，屬於Authority的區塊則避免合併。最簡單的做法即是將所有的區塊依序比對，若innerHtml相同，則將之合併。但

此法相當耗計算成本，因為就擁有1,000頁的網站來說，分割後的總區塊數可能高達10,000個以上，此方法的時間複雜度為$O(n^2)$，因此我們必須要想辦法降低計算量。

前面有提到，原則上只有Hub區塊才有必要進行合併，因為主題內容的區塊在正常情況下是不會重複出現。而Hub型態的區塊通常出現於四個邊界附近，因此我們可以設定一個長度d，只有在離四個邊界長度d範圍內的區塊，才取出做為合併計算的區塊，如此可節省不少的計算成本。由我們實驗結果顯示，在d為200pixels下，大約只有四分之一的區塊需取出做合併計算，如此只需原先十六分之一的合併計算成本。

前面有提到Hub區塊能合併的儘量合併，有些不同層目錄的導覽區塊，有可能會改變其底色、字型等呈現方式來凸顯或強調目前所在位置，但就本質上而言，這此導覽區塊應被合併。因此我們不能單純比對區塊的innerHtml，而應將區塊的有關呈現的tag或其屬性去除，例如<font>、<b>、style屬性…等，再進行比對，以達而Hub區塊合併最大化的目的。

3.1.2 鄰接矩陣之建立

區塊合併完成後，我們要為其建立鄰接矩陣以便後續進行計算之用。鄰接矩陣建立的方式與PageRank及HITS相同，只是我們以區塊為最小單元。同樣的，$A(i,j)=1/k$，表示區塊i有超鏈結連至區塊j，k為區塊i中的超鏈結數量。

3.2 權重設定

權重設定可說是本演算法最重要的部份，設定適當與否，會直接影響到計算的準確性。每一個區塊我們會給定兩個權重，一是Hub權重，標示為WH；一是Authority權重，標示為WA。每個權重又分別可依區塊的幾何屬性、組成內容及合併次數等三個不同的因素來做計算，以下分別說明之。

3.2.1 Hub權重

幾何屬性

這部份的權重可分為面積大小及位置來做計算。

無疑地，區塊面積越大其權重就應越高，但不應取線性正比。經由實驗結果我們發現取自然對數可以得到較為合理的結果。因此我們按照的公式2進行計算。

$$W_{H\text{-}Size}=ln(區塊面積)\,/\,ln(區塊所屬頁面面積) \quad (2)$$

位置，是作者區分該網頁內容重要性的一種方式，出現的位置，越容易被使用者注意到，表示該區塊的內容就越重要。因此，我們將網頁依前一節用以做為判斷是否要計算合併的距離邊界長度*d*以及一般視窗可視高度h將頁面分割成八個部份，如圖3.3。區塊依其位於這八個份部的位置，分別給定不同的位置權重。我們依據常見的網站區分出三類的網頁範本類型，如圖3.4所示，我們詳列了判斷的規則在表3.1~表3.3中分別對應網頁範本類型1~3，依這些表，我們即可依設定的網頁類型查對應的表得出位置的權重。最常見網頁範本類型為第一種，Hub區集中在左上角。依據WebSiteOptimimation.com的統計，網頁中最常被點按的區域為靠近左上角像F型的區塊[14]，此區塊可以被視為Hub區所集中的位置。

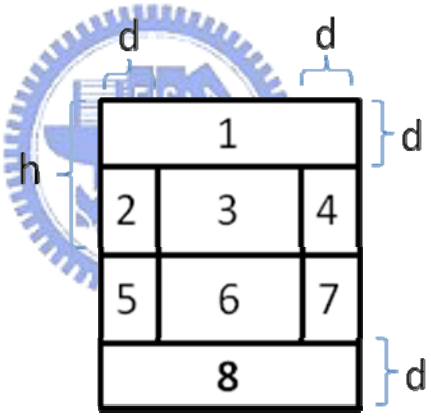
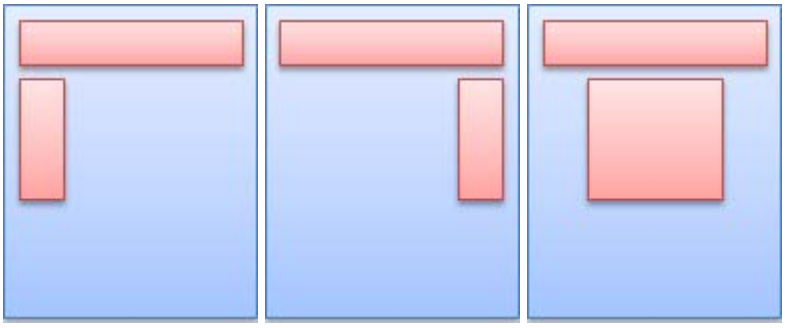


圖3.3 頁面權重區域



(1) Top-Left (2) Top-Right (3) Center

圖3.4 網頁範本類型

表3.1 區塊於頁面區域的Hub與Authority權重 - 網頁範本類型1

區塊	包含		向下超出		向右超出	
	H	A	H	A	H	A
1	0.75	0.1	1	0.75	X	X
2	1	0.25	0.75	0.25	0.75	1
3	0.75	1	0.5	1	0.5	1
4	0.25	0.25	0.25	0.25	X	X
5	0.5	0.1	0.25	0.1	0.25	0.75
6	0.25	0.75	0.25	0.5	0.25	0.75
7	0.25	0.1	0.1	0.1	X	X
8	0.1	0.1	X	X	X	X

表3.2 區塊於頁面區域的Hub與Authority權重 - 網頁範本類型2

區塊	包含		向下超出		向右超出	
	H	A	H	A	H	A
1	0.75	0.1	1	0.75	X	X
2	0.25	0.25	0.25	0.25	0.25	1
3	0.75	1	0.5	1	0.5	1
4	1	0.25	0.5	0.25	X	X
5	0.25	0.1	0.1	0.1	0.1	0.75
6	0.25	0.75	0.25	0.5	0.25	0.75
7	0.5	0.1	0.25	0.1	X	X
8	0.1	0.1	X	X	X	X

表3.3 區塊於頁面區域的Hub與Authority權重 - 網頁範本類型3

區塊	包含		向下超出		向右超出	
	H	A	H	A	H	A
1	0.75	0.1	1	0.75	X	X
2	0.25	0.25	0.25	0.25	0.5	1
3	1	1	1	1	1	1
4	0.25	0.25	0.25	0.25	X	X
5	0.25	0.1	0.25	0.1	0.25	0.75
6	0.75	0.75	0.75	0.5	0.75	0.75
7	0.25	0.1	0.1	0.1	X	X
8	0.1	0.1	X	X	X	X

組成內容

Hub權重在此僅做單純的比例計算，該區塊內的超鏈結所佔面積越大者，其權重越高，反之則越低，計算方法如公式3。

$$W_{H\text{-}Type} = 超鏈結所佔面積 / 區塊所佔面積 \quad (3)$$

合併次數

一般而言，出現越多次的區塊，就導覽區塊而言，其重要性越高，但不該成線性正比。我們經由實驗結果得出，若用合併區塊的權重總合除1加上個數的自然對數，結果最為合理。因此，因合併而衍生的權重調整，可以公式4計算。

$$W_H = \Sigma w_h / (1 + ln(k)), \quad k: 合併個數 \quad (4)$$

另外，前10名合併最多的區塊，若其位置是在區域1或8的範圍內，則其權重直接乘上0.1。因為合併次數最多的區塊很有可能是每頁都有的Logo、版權宣告…等，這些區塊對作者及使用者而言並非重要的區塊，所以直接調低其權重以降低其影響程度。

3.2.2 Authority權重

幾何屬性

這部份的權重同Hub的計算方式，同樣依公式3及表3.1進行權重的計算。

組成內容

Authority區塊的重要程度，也可依其組成的內容進行判斷。區塊的內容可分成文字、圖片及互動元件(如文字方框、按鈕…等)三種類別，其權重可參照表3.2進行設定。

表3.4 Authority區塊組成內容權重判斷表

區塊組成內容	權重	說明
文字	0.75	應是內文
圖片	0.5	只有圖片可能是Hub或內文
文字+圖片	1	內文
只有互動元件	0	沒有內容
文字+互動元件	0.1	可能為登入或搜尋區塊
圖+互動元件	0.1	同上
文字+圖+互動元件	0.25	同上，但考慮其多樣性，權重稍高

合併次數

Authority區塊照我們的假設不應被合併，因此就算被合併，其權重也不應改變。故合併次數所影響的權重，我們直接以平均值取代權重，意即公式5。

$W_A = \Sigma w_a / k$, k:合併個數 (5)

3.3 加入合併的區塊層級鏈結分析

原HITS的計算方式如公式(1)，但由於我們的計算單位已由「頁面」調整成「區塊」，所以原公式修正成公式(X)，其中 W_{A_p} 為區塊p的Authority權重、W_{H_q} 為區塊q的Hub權重，公式(6)的第一條式子可視為：區塊p的Authority值等於所有有超鏈結連至區塊p的區塊Hub值乘該區塊的Hub權重的總合，再乘上區塊p的Authority權重，其意義為由頁面Q的區塊q連結至頁面P，然後再看到區塊p的機率；第二條式子可視為：區塊p的Hub值等於所有被區塊p內的超鏈結連結到的區塊的Authority值乘上該區塊的Authority權重的總合。

$$A(p) = W_{A_p} \sum_{q \to p} H(q) W_{H_q}$$
$$H(p) = \sum_{p \to q} A(q) W_{A_q} \quad (6)$$

可再修正為(7)方便計算，其中a是區塊Authority值的陣列、h是區塊Hub值的陣列、A是各區塊的鄰接矩陣、W_H 是區塊Hub權重的Diagonal矩陣，$W_H(i,i) = W_{H_i}$ ，其餘元素為0、W_A 是區塊Authority權重的Diagonal矩陣，$W_A(i,i) = W_{A_i}$ 其餘元素為0。

$$a = (A^T h)^T W_H W_A$$
$$h = (Aa)^T W_A \quad (7)$$

我們將上面兩節所得出的鄰接矩陣A、Authority權重矩陣 W_A 及Hub權重矩陣 W_H 代入公式(7)計算，直至a與h中各元素均收斂至穩定狀態為止，即是我們所要計算的各區塊Authority及Hub值。

第四章

系統實作與實驗

本章節描述所實作之系統的開發環境與各模組功能，並以此系統來實際分析國內幾個政府單位及上市公司的網站，以評估此演算法的準確度。

4.1 系統實作概述

4.1.1 開發環境

本論文的系統以微軟的Visual Studio 2005做為開發工具，對於產出的資料，則存放於微軟的SQL Server 2000上。當初會想以微軟的工具來開發，主要就是為了能相容於實驗室蘇瑞元學長所開發的網頁資料萃取系統BODE，故選擇同一平台進行開發，希望能以Plug-in的方式讓BODE使用本系統的功能為目標。

4.1.2 模組介紹

就功能面來說，系統可分為(1)介面、(2)網頁蒐集、(3)網頁區塊切割、(4)區塊合併偵測、(5)區塊Hub及Authority權重設定及(6)區塊Hub及Authority值計算等六個模組。以下分別就這幾個模組簡單介紹。

介面

負責與使用者互動，提供使用者各項操作、設定的圖形化介面。

網頁蒐集

類似Crawler，會從給定的網址所包含的超鏈結往下擷取網頁，但僅限於同一個Domain Name或是在抓取清單內與排除清單外的頁面。擷取完畢後，會將頁面送給網頁區塊切割模組進行區塊的切割。

網頁區塊切割

在每一個頁面擷取完畢後，會接手進行區塊的切割。本模組所使用的演算法為VIPS[*]，會將過程中所產生的各項資料，包含最終的區塊資料存至SQL Server中。

區塊合併偵測

得出網頁區塊後，會依據設定的條件，取出可能需要執行合併偵測的區塊進行計算，原則上以第三章所提到的計算方式為主，將合併完的區塊資料存至SQL Server中。

區塊Hub及Authority權重設定

在此模組會針對進行合併後的區塊(不管有無合併)，進行權重的設定，會分成Hub與Authority兩種不同的權重，依第三章所提的三個衡量方式進行設定。

區塊Hub及Authority值計算

設定完權重後，則交由此模組進行鄰接矩陣的建立，並依第三章3.3所提出的修正HITS公式進行計算。

4.1.3 系統流程

系統可分為兩大部份，如圖4.1中所示。圖中左半邊負責網頁蒐集及區塊切割每一個頁面為一個循環；右半邊則是進行合併、Hub及Authority權重設定與最後Hub及Authority值的計算，左半邊的網頁蒐集及區塊切割全部完成後，才會執行這部份的計算。

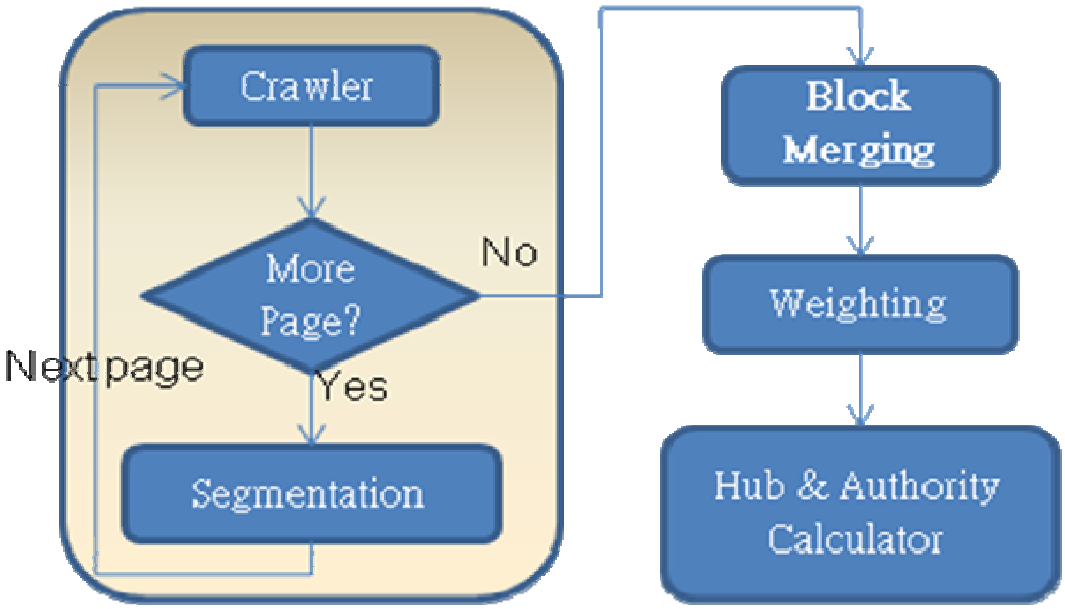


圖4.1 系統流程

4.2 實驗

為了驗證我們所提出演算法的正確性，我們實際以九個國內政府單位及財團法人的網站來進行實驗。

每個網站我們以四種計算方式進行實驗，分別是：一、合併區塊加權重設定；二、不合併區塊但加權重設定；三、合併區塊但不加權重設定；四、不合併區塊也不加權重設定。藉由此四種計算方式來確認合併區塊及權重設定對於單一網站的鏈結分析確實有其效果。

4.2.1 資料來源

網站A：

阿扁總統電子報(http://www.president.gov.tw/1_epaper/iod.html)

總頁數：50

總區塊數：386

合併後區塊數：177

網站B：

國家太空中心(http://www.nspo.org.tw/2005c/sitemap.htm)

總頁數：173

總區塊數：1383

合併後區塊數：287

網站C：

儀器科技研究中心(http://www.itrc.org.tw/)

總頁數：128

總區塊數：2216

合併後區塊數：1128

網站D：

國家地震工程研究中心(http://www.ncree.org/ZH/Home.aspx)

總頁數：123

總區塊數：2893

合併後區塊數：1146

網站E：

行政院(http://www.ey.gov.tw/mp?mp=1)

總頁數：74

總區塊數：1891

合併後區塊數：659

網站F：

立法院-部份區域 (http://www.ly.gov.tw/ly/01_introduce/0101_int/)

總頁數：67

總區塊數：469

合併後區塊數：130

網站G：

總統府(http://www.president.gov.tw/)

總頁數：130

總區塊數：1562

合併後區塊數：325

網站H：

台南縣政府文化局(http://cultrue.tncg.gov.tw/index2.php)

總頁數：119

總區塊數：3080

合併後區塊數：836

網站I：

苗栗縣政府(http://www.miaoli.gov.tw/index.asp)

總頁數：78

總區塊數：2525

合併後區塊數：591

共同設定：

網站擷取深度：3層

網頁權重型態：第一種 - 左上

平均視窗高度(h)：800 pixels

平均頁面邊界大小(d)：200 pixels

頁面切割的細緻度(PDoC)：0.7

4.2.2 實驗結果

我們以系統所計算出來的前100名的Hub與Authority區塊來做為評估的基準，分別由人工判斷各網站的前100名Hub與Authority區塊是否合適，分成三種等級，如表4.1所示。

表4.1 Authority與Hub評估等級表

等級	Authority	Hub
Good	與網站或網頁主題相關的內容區塊	直接導向與網站或網頁主題相關內容的導覽區塊
Normal	不與網站或網頁主題直接相關，但仍為有用的內容區塊。	最上層的導覽列，不直接連向資料區塊
Bad	無用或導覽性質的區塊	無用的導覽列(如回首頁、版權宣告…等)廣告及內容區塊

我們依表4.1所定義的內容來統計各實驗網站的四種計算方式的前100名Authority與Hub區塊，所得的圖表如下：

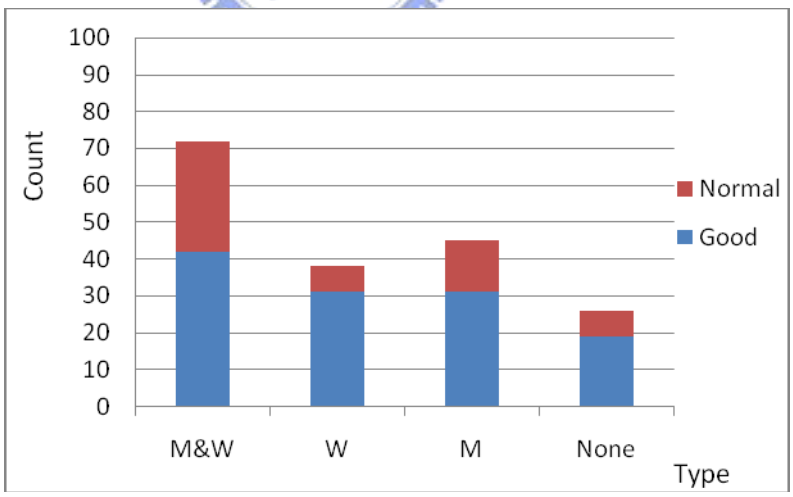


圖4.2 阿扁總統電子報 Authority區塊結果

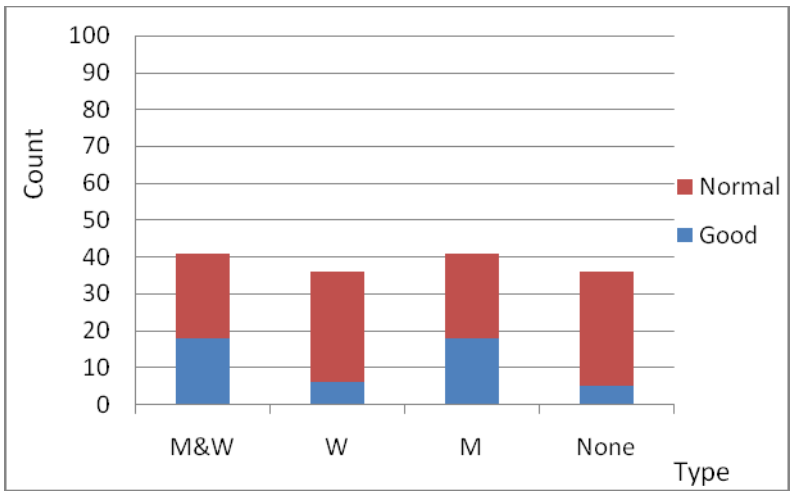


圖4.3 阿扁總統電子報 Hub區塊結果

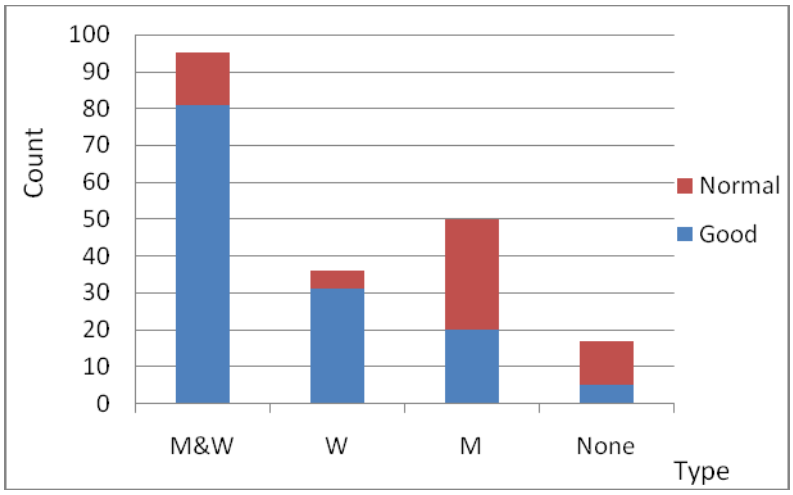


圖4.4 國家太空中心 Authority區塊結果

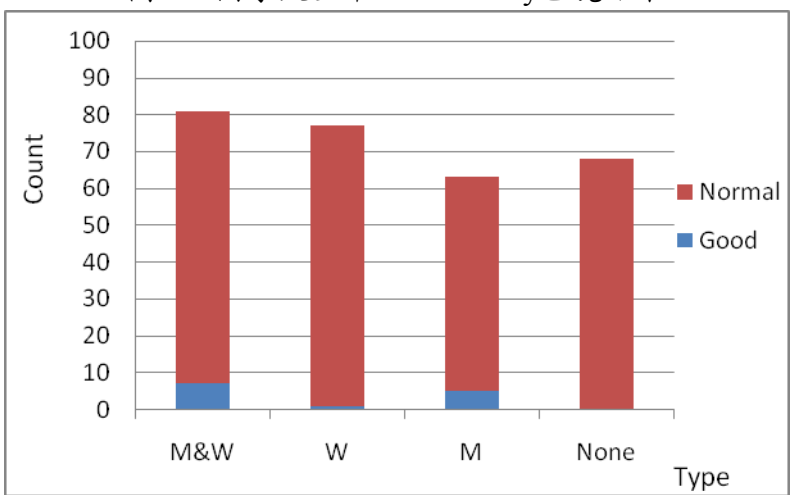


圖4.5 國家太空中心 Hub區塊結果

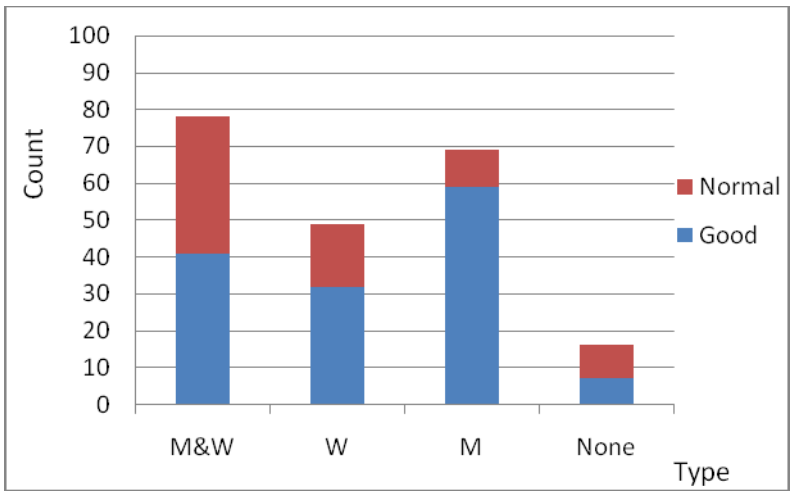


圖4.6 儀器科技研究中心 Authority區塊結果

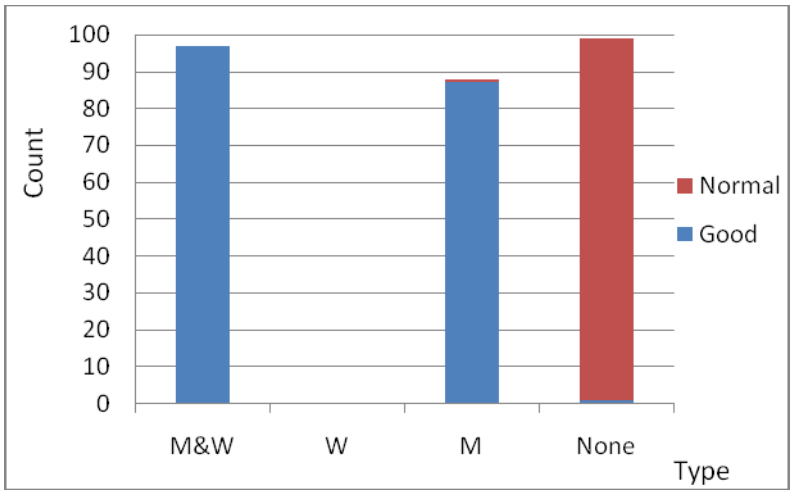


圖4.7 儀器科技研究中心 Hub區塊結果

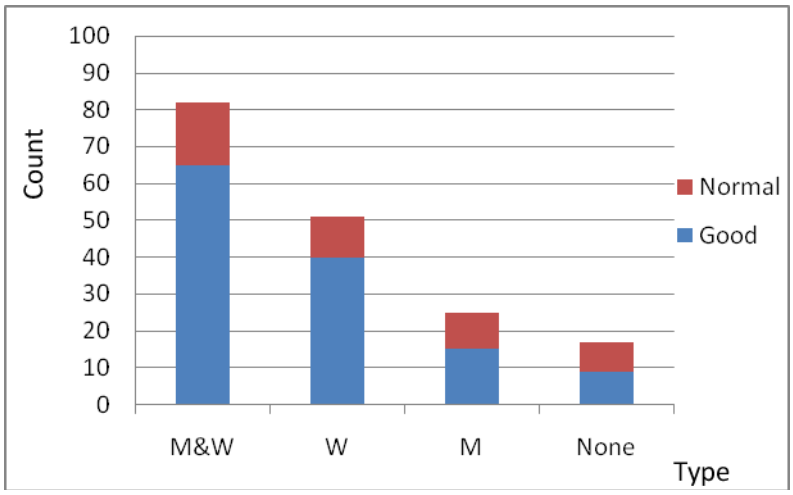


圖4.8 國家地震工程研究中心 Authority區塊結果

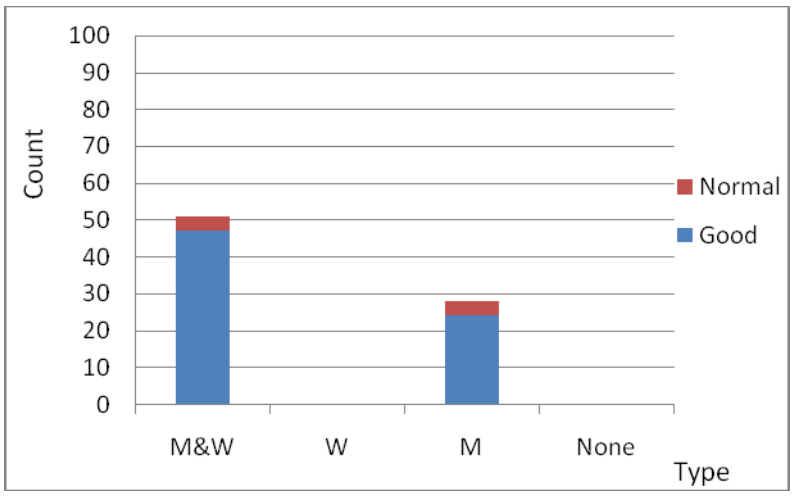


圖4.9 國家地震工程研究中心 Hub區塊結果

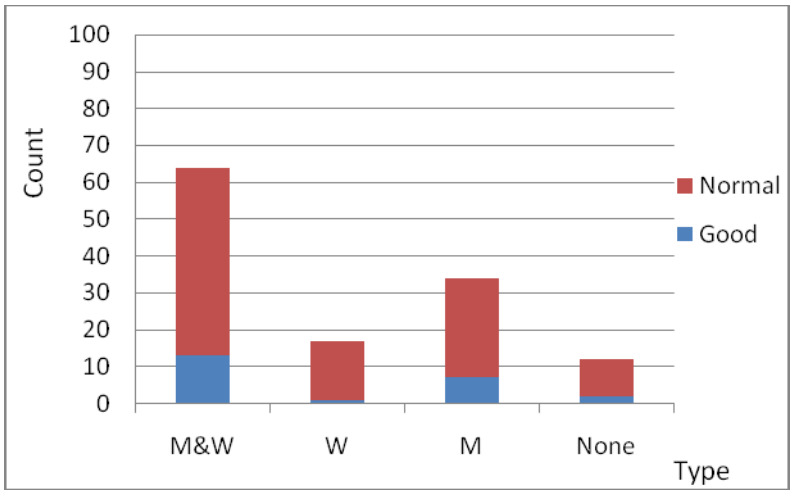


圖4.10 行政院 Authority區塊結果

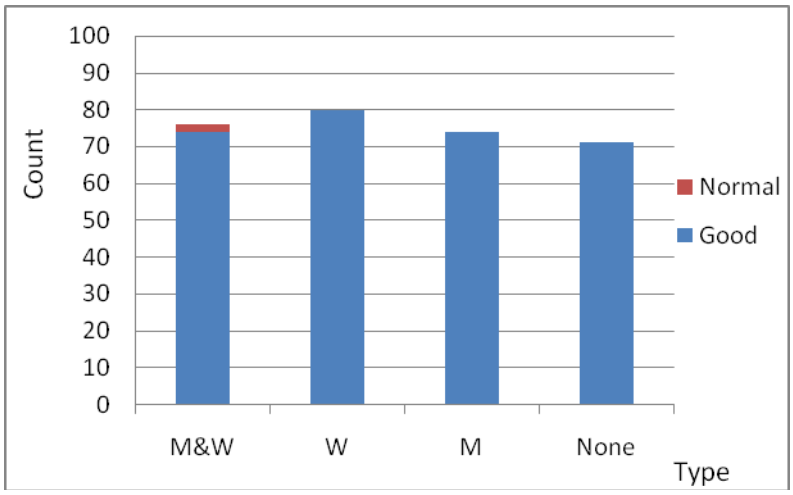


圖4.11 行政院 Hub區塊結果

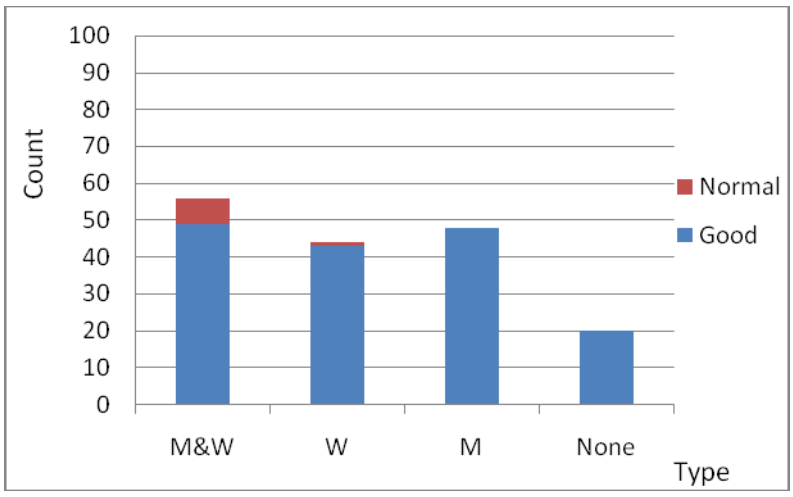


圖4.12 立法院-部份區域 Authority區塊結果

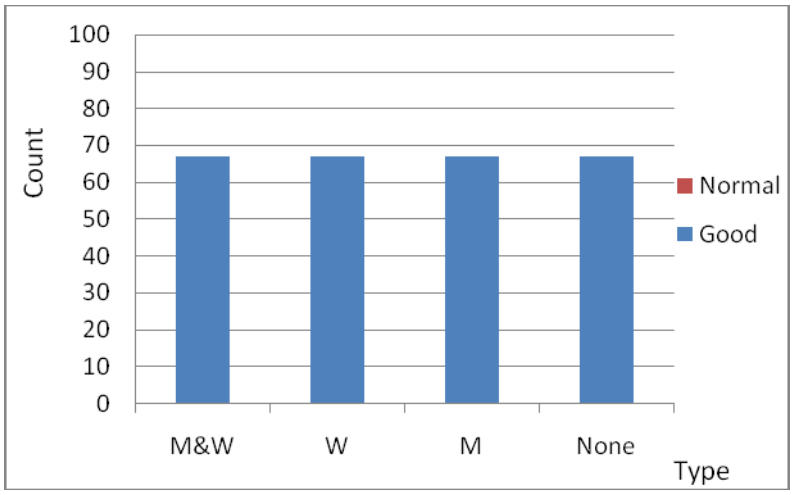


圖4.13 立法院-部份區域 Hub區塊結果

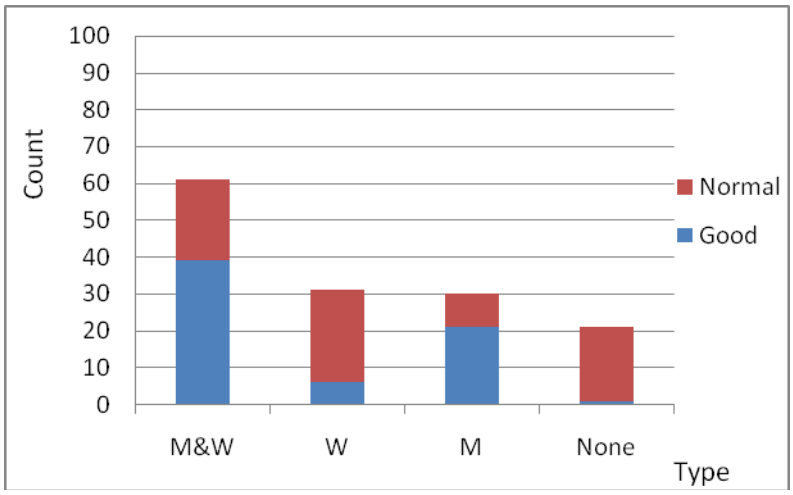


圖4.14 總統府Authority區塊結果

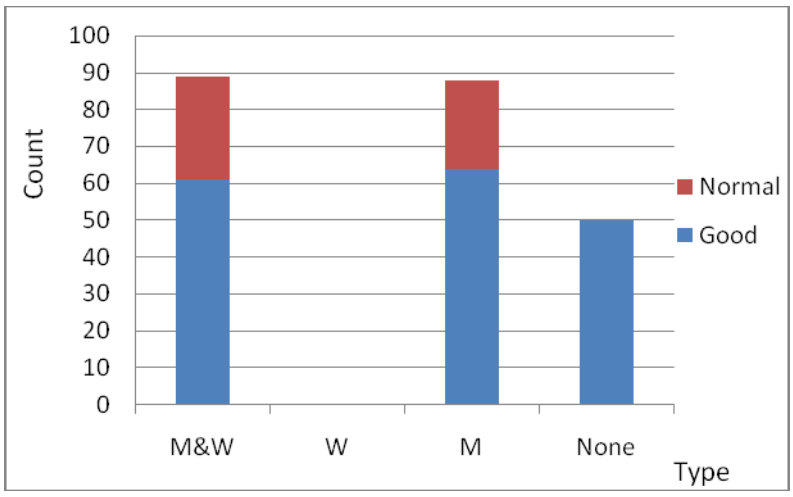


圖4.15 總統府Hub區塊結果

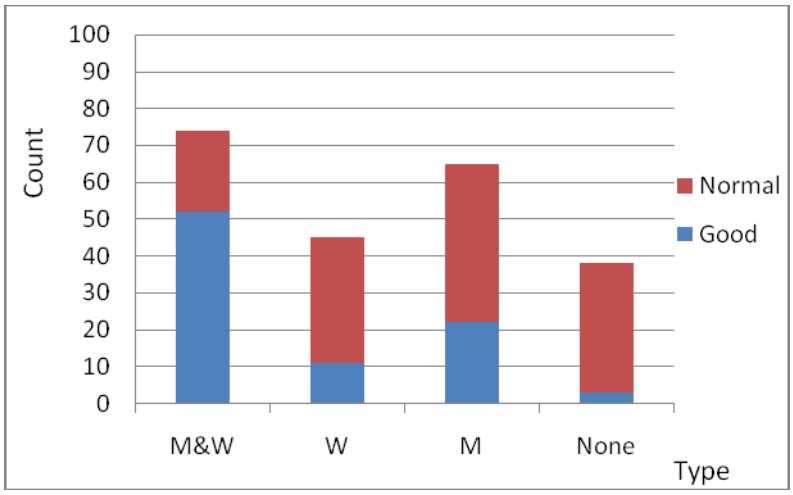


圖4.16 台南縣政府文化局Authority區塊結果

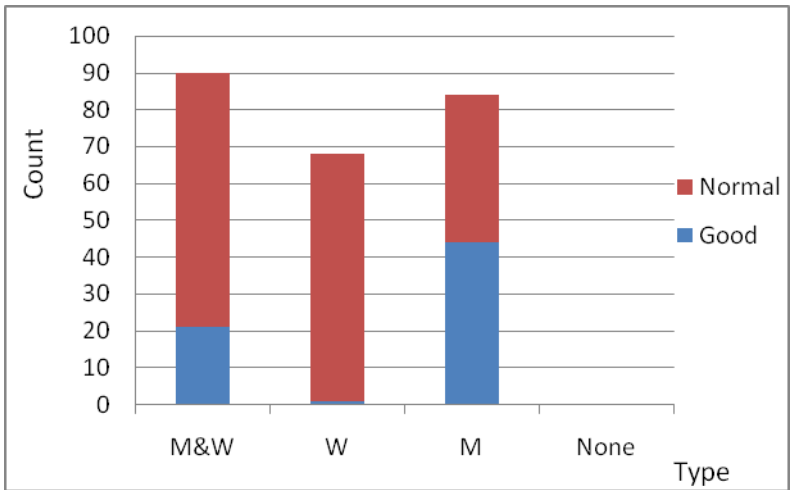


圖 4.17 台南縣政府文化局Hub區塊結果

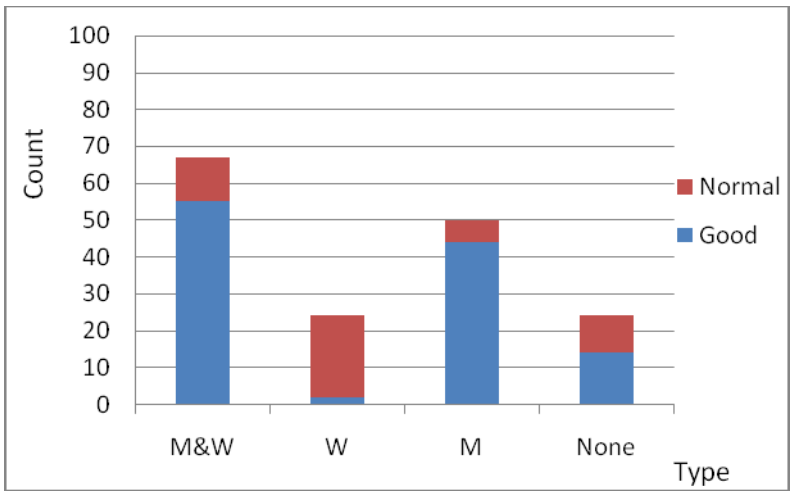


圖4.18 苗栗縣政府Authority區塊結果

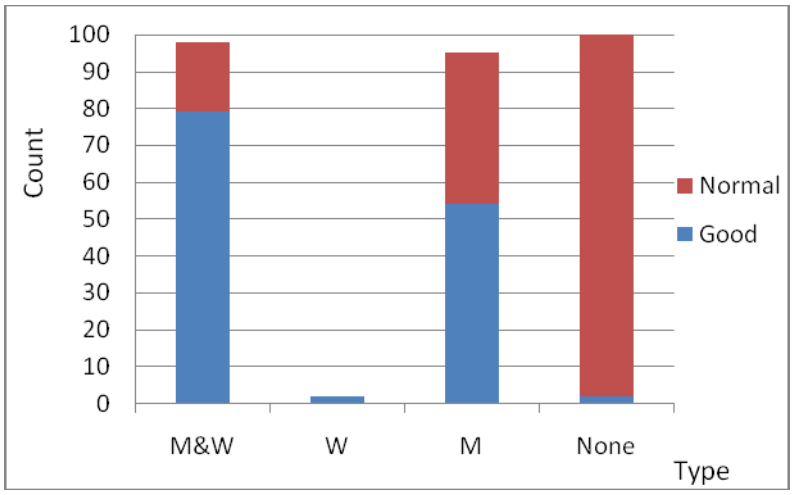


圖 4.19 苗栗縣政府Hub區塊結果

依據九個網站的Authority及Hub，我們整理出兩張比較的圖表，圖4-18與圖4-19。其中Y軸為各網站系統計算的前100個區塊經人工判斷為確實為Authority或Hub；X軸為各網站代號，其中A: 阿扁總統電子報、B: 國家太空中心、C: 儀器科技研究中心、D:國家地震工程研究中心、E: 行政院、F: 立法院、G: 總統府、H:台南縣政府文化局、I:苗栗縣政府。

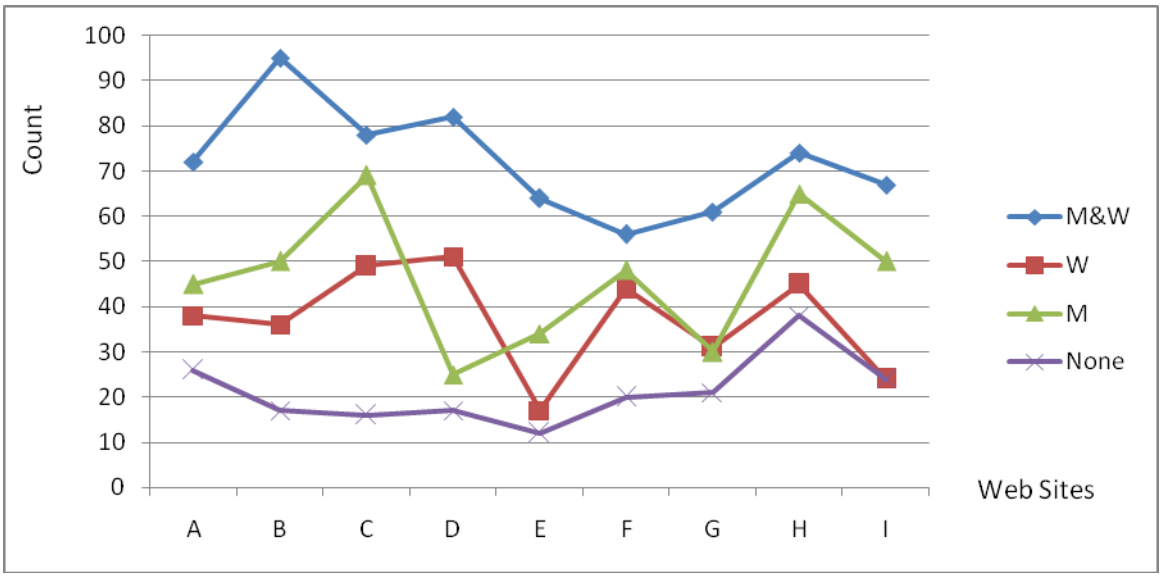


圖4-18 九個實驗網站的四種計算方式的Authority值比較表

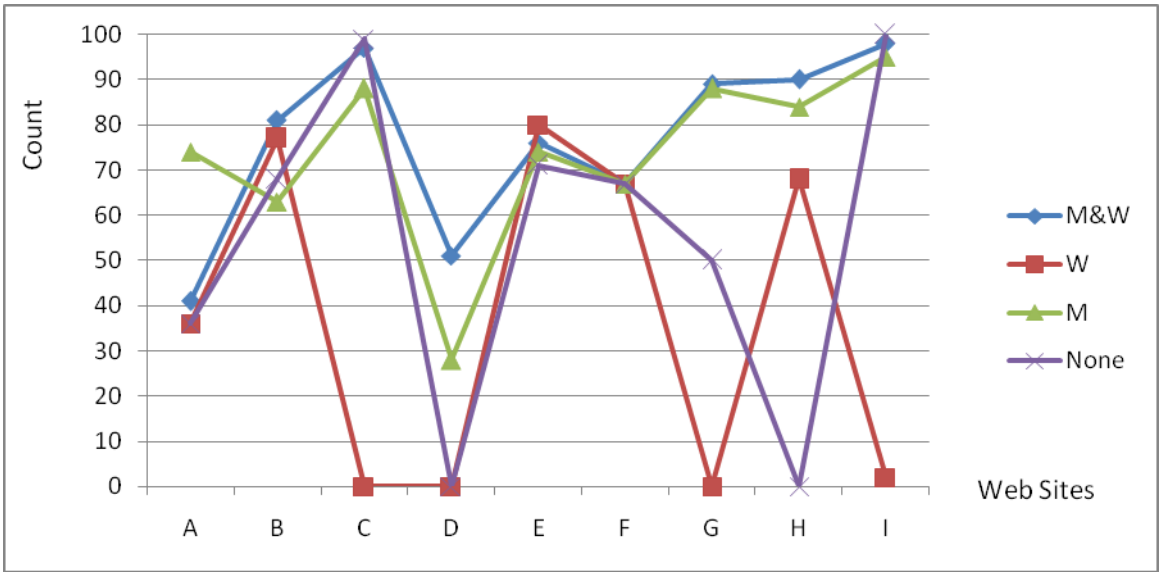


圖4-19 九個實驗網站的四種計算方式的Hub值比較表

在Authority部份，由圖4-18可以看出，區塊合併與權重設定對於Authority區塊的判斷正確與否，佔了很重要的影響，明顯優於只做合併或只做權重設定，或是都不做的計算結果。因此我們可以由這樣的結果得證明，在一個網站內部做區塊層級的鏈結分析，區塊合併與區塊權重的設定，對於內容區塊的判斷有著決定性的影響，比不做區塊合併與權重設定的準確度高出有三倍之多，詳細的數據可參考表4.2。

表4.2 9個實驗網站的前100名Authority區塊正確性比較表

Web Site	A	B	C	D	E	F	G	H	I	Avg.
M&W	72	95	78	82	64	56	61	74	67	72.1
W	38	36	49	51	17	44	31	45	24	37.2
M	45	50	69	25	34	48	30	65	50	46.2
None	26	17	16	17	12	20	21	38	24	21.2

表4.3 9個實驗網站的前100名Hub區塊正確性比較表

Web Site	A	B	C	D	E	F	G	H	I	Avg.
M&W	41	81	97	51	76	67	89	90	98	76.7
W	36	77	0	0	80	67	0	68	2	36.7
M	74	63	88	28	74	67	88	84	95	73.4
None	36	68	99	0	71	67	50	0	100	54.6

另外在Hub部份，由圖4-19可以看出，區塊合併與權重設定對於Hub區塊的判斷正確與否，似乎不甚相關。對於此現象，與理論不相符，經由研究實驗數據後發現，Hub部份在沒有執行合併的情況下，導覽區塊的數量相較於有執行合併的情況大了數倍之多。因此對於有執行合併的計算方式而言，一個Hub區塊會等於沒有執行合併區塊的計算方式的10個以上的Hub區塊。因此沒有執行合併區塊的計算方式在Hub部份會有大好或大壞的情況產生，排在前面100名的Hub區塊有可能一大串連續好的Hub區塊，也有可能是一大串連續不好的Hub區塊，其中C、D、G、H、I網站即是此種情況，但在有執行合併的計算方式來說，這些連續好或不好的Hub區塊，只會算一個。

所以相較於Authority區塊，Hub區塊比較不適合用此種方式進行統計，必須要擴大計算至所有的被系統標記為Hub的區塊才能得出與理論較為接近的數值。

由此九個網站所得出的實驗數據，整體的平均有用區塊辨識比率約為72.1%，數字雖稱不上漂亮，但已足以證明我們的演算法是可以有效地找出單一網站內的內容區塊。至於數字會有高低起伏的原因，經探討有發現有兩點，一是網頁區塊切割未達完美，二是權重的設定與網頁作者的角度不符。

原因一是因為我們所引用的網頁區塊切割演算法VIPS並無公開其程式或原始碼，我們是跟據馬維英博士團隊所發表的論文內容實作而來的，並無法確定是否有完整實作，因此自然在頁面切割的準度上，勢必是要打折扣的。再者當初此演算法發表的時候，網

頁用來做Layout的主要標籤是<TABLE>、<TR>與<TD>，但現在主流的Layout標籤是<DIV>及<SPAN>配上CSS。在這部份，因為並沒有修正的論文出現，我們是以自訂的規則處理，正確性當然還可以再進一步調整改善。

原因二則是因為網站中每一頁的Layout樣式不見得符合實驗所設定的Top-Left網頁型態，通常網頁應該都是按照某幾個範本製作的，我們所選擇的Top-Left型態與大多數網頁的範本的Layout相仿，但畢竟會有部份、甚至多數的頁面是與我們的假設不符的，此時權重的設定就會有偏差，自然影響計算的結果。

第五章

結論

5.1 總結

我們在本篇論文中提出了適用於單一網站區塊內容重要性分析的鏈結分析演算法，並藉由實作的系統來驗證理論是否正確。由實驗結果來看，我們確實可以辨識出單一網站中重要的內容及導覽區塊，但是精確度仍可再做進一步提昇。

當初會做這方面研究的主要目的即是著眼於目前實驗室的網頁資料萃取系統BODE，在萃取資料的過程中仍需大量的人力介入，方能撰寫系統萃取資料所需的BODE Script，因此希望能降低、甚至是不需要人力介入即可完成，因此我們需要一個可以自動辨識網站重要內容區塊的演算法來輔助，故產生了本論文所提之演算法。

當然，此演算法的應用不單單僅限於網頁資料萃取的自動化而已，接下來，將說明此演算法可應用的領域。

5.2 相關應用

網頁資料萃取自動化

前面有提到，本論文的主要目的即是在於輔助本實驗室的網頁資料萃取系統(BODE)的自動化。因為本論文的目的即在找出單一網站內的重要內容區塊(Authority)與導覽區塊(Hub)的排名，我們可以藉由找到的這些Authority與Hub區塊，建構出從首頁算起的資料路徑，即區塊的XPath。BODE進行萃取所需要的BODE Script中指定資料位置的即是XPath。

當然，我們所切割出來的區塊的細緻程度可能無法滿足BODE所需，因為BODE所萃取的大多為DOM-Tree中的葉節點(leaf node)，而本論文所得到的是一個代表同一語意的區塊。例如在一個3C的賣場網站，我們所得到的重要內容區塊可能會是一個描述各個不同等級的電腦規格的<TABLE>區塊，也就是說是電腦規格的集合。但BODE要萃取的資料則是單一筆的電腦規格，可能是<TR>或是<TD>區塊，然後依序地將該<TABLE>區塊中的每一筆電腦資料萃取出來，所以這中間仍有一些研究必須要進行，也就是如何把區塊的資料轉成更詳細的物件或語意資料以符合BODE萃取所需。

網頁過濾及更新追蹤

目前一般的網站內容都相當豐富，而且幾乎天天都會有更新，若網站沒有提供RSS的訂閱服務，使用者就必須要先花時間找尋網站中有興趣或重要的網頁，再來更是要自己定期追蹤是否有相關的更新發生，無疑又是一件費時的差事。

因此我們可以運用此演算法，配合上特定的關鍵字協助使用者過濾出相關的重要內容，並做為基準，爾後定時再依同樣的條件對網站進行分析，藉由Diff演算法判斷出是否有更新，自動蒐集並呈現給使用者相關的資料，增加資訊利用的效率。

如此，就算網站沒有提供RSS的服務，也可以達到類似的效果，並事先將資料下載至本機端，節省網路傳輸的等待時間。

搜尋引擎精確度提昇

除了BODE之外，Search Engine也可以利用本論文的演算法來增進其精確度。怎麼說呢？因為我們的演算法主要就是在過濾單一網站中的重要區塊，因此若Search Engine可以先以我們的演算法計算其所抓取的各個網站，而得出其中重要的區塊後再進行各頁面分數的計算，相信可以在精確度上做一定程度的提昇。

就拿現在最熱門的Search Engine Google來說好了，其搜尋結果中仍免不了夾雜著一些導覽或是廣告連結等不適用的資訊，若能先行將每個網站中這些不重要的部份事先去除，應該可儘量降低這種情況。但此種應用仍須考量計算成本，因為多了這層計算工作，勢必對效能有所影響，而且要如何融入原先Search Engine的演算法中也是值得考慮的因素。

5.3 未來工作

網頁類型自動判斷

由實驗的結果我們可以發現，區塊權重的設定可以說是本演算法成功與否的關鍵，目前我們依三種不同的網頁類型事先定義好設定權重的規則，由人為去判斷每次要進行分析的網站是屬於何種類型。因為同一個網站的網頁，可能會有不同的呈現方式，導致我們所設定的權重規則無法完全符合每一個網頁。因此若要增加精確度並減少人工介入，網頁類型的自動判斷是未來要進行的一個方向。可能可以藉由Pattern的比對，事先建立幾種典型的DOM Tree Pattern或是VIPS所建構出來的Semantic Content Structure Pattern，自動判斷該網頁應該套用何種權重的設定方式。

修正網頁切割演算法

在第四章實驗的結果檢討中有提到現在的網頁Layout標籤開始大量使用<DIV>，這是VIPS所沒有考慮到的部份，我們是以<TABLE>標簽的處理規則再稍加改善，自然準確度會較低。

因此為求整體演算法準確度的提昇，勢必是要針對目前Layout用的標籤對於VIPS演算法的衝擊做相對應的修正，因此應該對<DIV>標籤在頁面上使用的情況詳加分析，特別為他定義額外的規則，而不要以<TABLE>標籤為基礎。

關鍵字

目前本論文所得出的重要內容及導覽區塊是比較一般性的，也就是說完全依據網站的呈現方式及各區塊間的鏈結關係進行分析所得來的。若要進行特定領域的分析，例如使用者可能想利用BODE到某3C賣場的網站自動萃取筆記型電腦的資料，便需要導入關鍵字的影響。當然，關鍵字不能只是純粹將含有該關鍵字的區塊的權重加重，還必須要考慮區塊間連結的正向及反向關係，例如含有關鍵字的Hub區塊對其所連結到的Authority區塊的影響，或含有關鍵字的Authority區塊對連結到他的Hub區塊的影響等，都是必須考慮的重點。

資料路徑辨識

上面相關應用有提到，本論文的主要目的是在輔助本實驗室的網頁資料萃取系統BODE的自動化，也就是要如何降低人力的介入、或甚至不需要人力介入來產生所需的BODE Script，即資料XPath。所以資料路徑的辨識仍是一個重要的問題，否則空有資料區塊，但沒有相對於首頁的路徑及區塊內容的資料項目化，對於網頁資料萃取系統的自動化，仍無太大的助益。

參考文獻

[1] I-Chen Wu, Jui-Yuan Su, Loon-Been Chen, "A Web Data Extraction Description Language and Its Implementation," compsac, pp. 293-298, 29th Annual International Computer Software and Applications Conference (COMPSAC'05) Volume 1, 2005

[2] http://www.worldwidewebsize.com

[3] http://www.google.com

[4] http://www.google.com/products

[5] http://www.scholar.com

[6] Cai, D., Yu, S., Wen, JR, Ma, WY, 2003. "VIPS: A Vision-base Page Segmentation Algorithm.", Technical Report, MSR-TR-2003-79, Microsoft Research Asia.

[7] J. Kleinberg, "Authoritative sources in a hyperlinked environment", *Journal of the ACM*, Vol. 46, No. 5, pp. 604-622,1999.

[8] L. Page, S. Brin, R. Motwani, and T. Winograd, "The PageRank citation ranking: Bringing order to the web", *Technical report*, Stanford University, Stanford, CA, 1998.

[9] Deng Cai, Xiaofei He, Ji-Rong Wen and Wei-Ying Ma., "Block-level Link Analysis", Microsoft Technical Report MSR-TR-2004-50, 2004.

[10] Deng Cai, Shipeng Yu, Ji-Rong Wen, Wei-Ying Ma., "Block-based Web Search.", In Proc. of the SIGIR'04 Conf., pages 456-463, 2004.

[11] Z. Nie, Y. Zhang, JR Wen, and WY Ma., "Object- level ranking: Bringing order to web objects.", In Proceedings of WWW Conference, 2005.

[12] Ruihua Song, Haifeng Liu, Ji-Rong Wen and Wei-Ying Ma, "Learning Block Importance Models for Web Pages[A]." In proceeding of the Thirteenth World Wide Web conference[C], New York, NY: ACM Press, 2004, 203-211.

[13] Shian-Hua Lin,Jan-Ming Ho. "Discovering Informative Content Blocks from Web Documents", KDD-02, 2002

[14] http://www.websiteoptimization.com/speed/tweak/clickstream/

[15] Chang, C.-H., and Shao-Chen, L. IEPAD: Information extraction based on pattern discovery. In Proceedings of the tenth international conference on World Wide Web(2001)

[16] Arasu, A., Garcia-Molina, H.: Extracting Structured Data from Web Pages. In: Proceedings of ACM SIGMOD International Conference on Management of Data (SIGMOD 2003), San Diego, California, USA, ACM Press (2003)

[17] Hung-Yu Kao, Shian-Hua Lin, Jan-Ming Ho, Ming-Syan Chen, Mining Web Information Structures and Contents based on Entropy Analysis, IEEE Transactions on Knowledge and Data Engineering , volume 16, issue 1, pages 41-55, Jan 2004.

[18] Hung-Yu Kao, Jan-Ming Ho, Ming-Syan Chen, WISDOM : Web Intra-page Informative Structure Mining based on Document Object Model, IEEE Transactions on Knowledge and Data Engineering, volume 17, issue 5, pages 614- 627, May 2005.

[19] Mendez-Torreblanca, A., Montes-y-Gomez, M., and Lopez-Lopez, A.: A Trend Discovery. System for Dynamic Web Content Mining. Proceedings of the 11. th. International Confer-. ence on Computing, Mexico City, Mexico (2002)

[20] S. Debnath, P. Mitra, N. Pal, and C. L. Giles, “Automatic Identification of Informative Sections of Web Pages,” IEEE Transactions on Knowledge and Data Engineering 17, 9, Sep. 2005.

[21] Deng Cai, Xiaofei He, Zhiwei Li, Wei-Ying Ma and Ji-Rong Wen, “Hierarchical Clustering of WWW Image Search Results Using Visual, Textual and Link Analysis”,12th ACM International Conference on Multimedia, Oct. 2004 .

[22] Zaiqing Nie, Yunxiao Ma, Shuming Shi, Ji-Rong Wen and Wei-Ying Ma, Web Object Retrieval, The 16th international World Wide Web conference (WWW 2007)

[23] CHEN, Z, LI, T, WANG, J, LIU, W Y and MA, W Y, "A Unified Framework for Web Link Analysis", Proceedings of the 3rd International Conference on Web Information Systems Engineering (WISE 2002), Singapore, December 2002, pp 63-72.