Web 搜尋結果視覺化介面設計

A Visual Interface Design for Web Search Results

楊錫謦　　曾閔棋　　楊正仁
元智大學資訊工程研究所
桃園縣中壢市遠東路 135 號
{edgar,angeline,czyang}@syslab.cse.yzu.edu.tw

摘要

隨著網路的蓬勃發展，現今在網際網路上的資訊量已遠遠地超過十年前的情況。雖然目前已經有許多搜尋引擎來幫助使用者在網際網路上找尋所需的資訊，但是一般搜尋引擎是以條列式的文字來表示搜尋結果，如此的表現方式無法展現搜尋結果之間的關係，使用者無法儘速找到相關重要資訊。因此我們在搜尋引擎前端，設計一個視覺介面，稱之為 VISE (Visual Interface for Search Engines)，並實作 VISE 雛形系統。由初步實驗結果發現，由於 VISE 利用圖形清楚地呈現搜尋結果以及它們之間的關係，因此加速了使用者瀏覽搜尋結果的速度，並且讓使用者更快地找到所需的重要資訊。雖然實驗規模並不大，我們相信 VISE 能為未來搜尋結果視覺化方法的發展提供幫助，讓使用者能以更快的速度在網路上找到所需要的資訊。

關鍵詞：Web 搜尋引擎，視覺化，介面設計，熱門參考網站。

一、簡介

隨著網路的蓬勃發展，現今在網際網路上的資訊量已遠遠地超過十年前的情況。由於網路存在這些大量的資訊，使用者在搜尋資訊時，也需要花上很久的時間等待或是判斷，才能找到部分重要的相關資料。也因此，搜尋引擎變成為目前的一個熱門研究方向，有不少研究討論如何增進搜尋出來的精確度，或者加速使用者找到相關重要網頁的過程[5,9,12]。

雖然目前已經有許多搜尋引擎來幫助使用者在網際網路上找尋所需的資訊，如 Yahoo!，Lycos 及 Google，但是由於目前常用的搜尋引擎在結果呈現上，都是以條列的方式（list-based representation）表示，因此提供給使用者的相關資訊，仍然十分有限。例如 Excite、Yahoo 或是 Google 等等，在搜尋結果呈現上在每一項結果的標題下面加上一段對於這項結果內容的簡短描述，或是包含有搜尋字串的一段內文，再以捲軸輔助捲動螢幕，來瀏覽超出螢光幕之外的資訊。因此若是搜尋結果的數量很多，便需要將資訊分開置放在不同的網頁中，並以不同的頁碼當成索引，讓使用者在不同頁面之間瀏覽。我們發現這種條列式的呈現方式，存有下列三項缺點：

1. 在使用者找尋資訊時，條列式呈現的方式會使瀏覽結果的動作相當繁複，並且浪費時間。主要是因為關於條列式呈現中出現的簡短描述，其實只能提供有限的資訊。對尋找重要資訊的目的而言，使用者並不容易判斷哪一些搜尋結果中包含所需要的重要資訊。Amento 等人[2]曾指出，以現存的搜尋工具，使用者對於搜尋結果必須一個接著一個地瀏覽，直到使用者認為滿意為止，而這樣的瀏覽方式卻會花費太多的時間與精神。

2. 條列式呈現方式雖然依照某個序分標準（scoring）排列，但每個項目間，只能呈現出這種序分的線性關係資訊（linear scoring relationship）。但是事實上，從觀察這些使用的過程中，常常會發現某些網頁之間其實存有相互參考的關係。這些關係連結，就不容易以條列方式呈現。使用者也必須一個一個項目瀏覽之後，才能重組這些關係的輪廓。也因此浪費使用者的瀏覽時間。

3. 此外，一些類似或是屬於同一個網站的網頁常會出現在不同的搜尋結果頁面

上，並且有時候它們之間可能相距數個頁面。然而使用者在還沒有翻到「下一頁」之前，無法預測下一頁是否含有一些其他的相關網頁項目。因此這些和之前相關的資訊，往往阻礙使用者的瀏覽速度，無法儘速找到其他相關重要資訊。

過去也有不少研究在討論如何改進搜尋結果的視覺呈現效果。例如 TopicShop [2]以類似微軟視窗作業系統中的檔案管理員介面，顯示網站的相關資訊，其中包括網站所包含網頁數目、超連結數目以及多媒體檔案數目等等。HyperSpace [3]將連續且相關的不同搜尋合併，顯示每一次搜尋之間的關係。WebQuery [6]利用網站間的超連結整理搜尋結果，並使用不同的視覺化技術來呈現搜尋結果。CardVis [11]也是利用超連結將搜尋結果擴大為相互連結的叢聚，並以卡片的方式呈現擴大後的叢聚。

但是以上這些研究要不偏重於序分線性關係的視覺化呈現，就是無法清楚地完整呈現搜尋結果之間關係結構。因此，在這些方法中，使用者在尋找重要資訊上，所得到的幫助實在有限。

其實許多搜尋引擎所表現的序分線性關係資訊中，就包含網頁與網頁之間鏈結（link）的關係。這些鏈結關係，可以用有向圖（directed graph）來表示。但是在條列式呈現時，卻無法呈現這些關連。因此我們考慮在搜尋引擎前端，設計一個視覺介面（visual interface），以圖形視覺化的方式，重新呈現搜尋結果的結構。透過這個視覺介面，使用者將可以很快瞭解搜尋結果相互之間的關係，從而快速決定瀏覽的次序，更快找到他們所需要的資訊。使用者甚至可以察覺出來不同搜尋結果的網頁間，複雜的直接或間接的超連結關係，從而發現常常被引用到的重要網頁。

在本論文中，我們將描述說明這個搜尋引擎的視覺化介面設計。這個視覺化介面稱之為 VISE （Visual Interface for Search Engines）。VISE 將搜尋引擎所搜尋得到結果的資訊結構以圖形方式呈現，使得使用者可以瞭解搜尋結果之間的關係，因此縮短瀏覽及尋找的時間，讓使用者以更快的速度找到所需要的資訊。同時，在視覺化呈現上，VISE 在下列兩點上改進，以求得最佳視覺效果：

- 減少叢聚及線條交錯的問題，以求整齊清楚的圖形表現，並且平均地分佈搜尋結果的圖形結構，充分利用顯示空間。
- 避免將資訊結構簡化或分解，而將整個搜尋結果的關係結構完整呈現。

我們並以 Java 實作出一個 VISE 雛形系統，以目前現有的搜尋引擎作為後端，當 VISE 接收到使用者所輸入的查詢字串後，傳送到後端的搜尋引擎。等查詢結果出來後，再將圖形結果輸出到使用者的瀏覽器上。

在初步的評估實驗中，我們將 VISE 的視覺化呈現與原本搜尋引擎的條列式呈現的進行使用者調查比較。調查結果顯示，大多數的使用者均偏好 VISE 的視覺化呈現介面。我們也從調查結果中，初步驗證 VISE 具有下列三項貢獻：

1. 使用者能較快找到重要資訊。因為 VISE 所呈現的搜尋結果關係圖中，將有參考關係的網頁集合起來，因此使用者將會明顯看到具有大量參考關係的網頁群組。這些大群組並且顯示在其他小群組之前，因此使用者更容易發覺一些熱門參考網頁。因此當 VISE 表現出網頁節點關係結構後，使用者的確能夠從中獲益，能夠在較短時間內迅速找到他們所需要的重要資訊。

2. 使用者能更有效率地瀏覽搜尋結果。這是因為具有參考關係的網頁將被聚集在同一個群組中，不會像傳統條列式呈現方式一樣地散佈在不同頁面上。因此使用者在瀏覽時，將會更有效率。使用者可以選擇先瀏覽一些序分不高但是是較為熱門參考的網頁節點，如此便可節省使用者瀏覽搜尋結果的時間。

3. 由於 VISE 是一個視覺化介面，因此不必更動搜尋引擎的設計，也簡化應用上的複雜度。因此如果未來出現更好功能更強的搜尋引擎，只需要適當修改 VISE 中所對應的相關查詢模組，就能夠迅速提供使用者一個視覺化查詢介面。

雖然參與實驗的使用者族群平常使用搜尋引擎的目的是偏向於以研究為主，且有使用搜尋引擎的豐富經驗，但是我們相信其他一般的使用族群在經過簡短時間的適應之後，也能夠利用 VISE 快速找到他們所想要找尋的資訊。因為 VISE 利用圖形代替傳統條列的文字表示，可以讓使用者很直覺地決定瀏覽網頁的順序，使用者不需要浪費過多

的時間在瀏覽搜尋結果的過程中，而能很快找到重要的資訊，可節省下許多時間從事其他的活動。在搜尋引擎方面，由於使用者皆可以透過圖形化的介面找到重要資訊，較之傳統的條列式文字表示方式，加強了對於使用者的幫助，可使得整體的效能提升。

我們將先在第二節中介紹其他的相關研究。在第三節中將會說明 VISE 的系統架構以及視覺化的過程。第四節將介紹 VISE 的雛形系統，以及我們所進行的相關實驗。最後，第五節是我們的結論，並討論 VISE 未來可以進一步進行的研究工作。

二、相關研究

Amento 等人在 1999 年提出 TopicShop 系統[2]，以類似微軟視窗作業系統中的檔案管理員介面，顯示網站的相關資訊，其中包括網站所包含網頁數目、超連結數目以及多媒體檔案數目等等。雖然 TopicShop 利用視覺化的介面將所搜尋到的結果分類，但是由於他的視覺化介面並不考慮將網頁 之間的參考關係呈現出來，因此使用者並不容易發覺熱門參考網頁。

HyperSpace [3]將連續累進且相關的不同搜尋合併，顯示每一次搜尋之間的關係。因此使用者雖然可以發覺累進搜尋（incremental search）時不同次搜尋之間重複出現的重要網頁但是由於他也並未顯示網頁之間的參考關係，因此使用者只能確認出當查詢條件範圍縮小時的重要網頁，卻無法發覺出其他網頁與這些重要網頁的關係。

WebQuery [6]利用網站間的超連結整理搜尋結果，並使用不同的視覺化技術來呈現搜尋結果。其中有數種視覺化的方式。其中一種是利用相鄰圖形（neighbor graph）來依照網頁的連接度（connectivity）繪出圖形，而這一種方式與 VISE 所用的方式相當類似。但是在相鄰圖形中，WebQuery 會加入一些額外的人工節點（artificial nodes）以使圖形成為一個弱連接圖（weakly connected graph），因此圖形將會變得複雜，也不容易分辨不同的網頁群組。

CardVis [11]也是利用超連結將搜尋結果擴大為相互連結的叢聚，並以卡片的方式呈現擴大後的叢聚。但是由於每張卡片中只能顯示兩層的階層架構，所以其實所顯示的資訊相當有限，使用者必須在不同卡片中瀏覽才能掌握整個群組的架構。

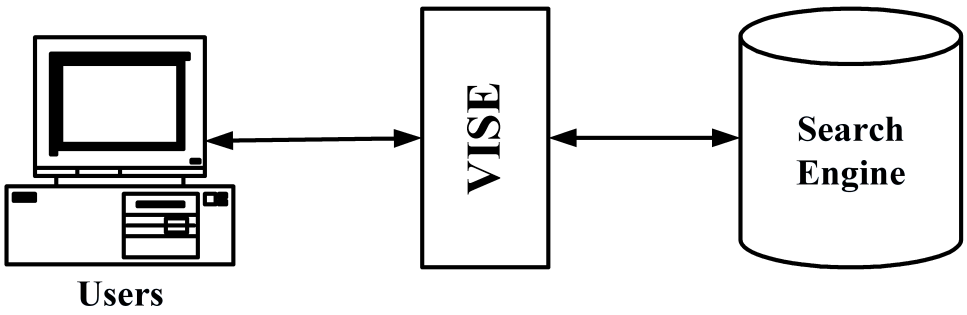
WebCiao 是一個 Web 網站的視覺化及追蹤工具[7]。雖然偏重在網站架構改變上的視覺化呈現，但他也用到一些視覺化方式，會將一個網站中相鄰網頁的關係畫出。只不過他並不著重在網頁參考關係呈現，他的視覺化方式如果用在搜尋引擎上，將無法充分看出各個不同網頁群組中的參考關係。

三、VISE 視覺化介面

在使用者利用搜尋引擎來搜尋網頁時，如果使用者可以得到搜尋結果的關係結構，那將能夠讓使用者快速掌握這些搜尋結果之間的關係。為了達到以上的需求，我們提出 VISE 這個為在搜尋引擎前端的視覺化介面。VISE 將搜尋引擎所傳回的搜尋結果經由視覺化演算法畫出它們之間的關係，並同時提供搜尋結果的相關資訊，如此將讓使用者快速掌握搜尋結果之間的關係，並縮短使用者發覺重要資訊所需要的時間。以下我們將分別介紹 VISE 系統架構以及相關的演算法則。

3.1 系統架構

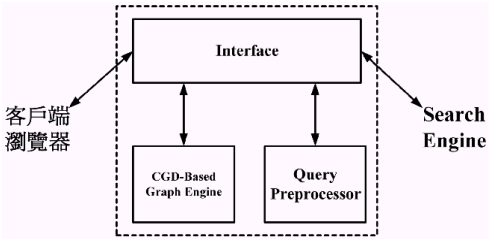
圖一為我們所提出的搜尋結果視覺化架構。VISE 介於使用者與搜尋引擎之間，首先



圖一：搜尋結果視覺化架構

接收使用者的搜尋字串並傳送至搜尋引擎進行搜尋，接著接收搜尋引擎所傳回的搜尋結果，經過分析處理，將所收集到的搜尋結果以節點及線段呈現它們的關係，再將此圖形結構傳回給使用者。

在視覺化方面，為了避免視覺混亂（visual clutter）以及節點過於集中於少數叢聚（cluster）中，節點最好能平均分佈在螢幕空間中[4,8]。所以我們決定使用 McCreary 等人所提出的 CGD 演算法[10]為基礎來發展我們的視覺化引擎核心。這是因為 CGD 畫圖演算法在空間的分配上，不像在不同的步驟中分別決定節點 X 軸與 Y 軸座標位置的其他畫圖演算法，而是將圖形分解為平行或是線性族群（clan）的集合，以平行族群以及線性族群的特性來"同時"決定節點所佔的 2D 空間大小，進而決定每一個族群所佔的空間，使得節點可以平均分佈在顯示的畫面中。並且在線條交錯方面，雖然 CGD 演算法並未在此求得最少交錯結果，但是已經具有相當良好的結果[10]。事實上，最少線條交錯的問題是一個 NP-hard 的問題[4]，因此在繪圖速度以及視覺效果兩者考量下，我們決定以 CGD 演算法為基礎將 Web 搜尋結果視覺化。



圖二：VISE 之系統架構

圖二為 VISE 內部架構圖，主要由 query preprocessor 以及 CGD-based graph engine 兩個部分所構成。當客戶端送入查詢字串後，VISE 將查詢字串依照後端搜尋引擎的格式重組，送往搜尋引擎。當搜尋引擎傳回搜尋結果時，這時得到的搜尋結果將會送到 query preprocessor 中處理。Query preprocessor 會再一一察看搜尋結果每一項目的網頁內容，因而建構出搜尋結果關係圖的架構。這個架構將再透過 CGD-based graph engine 畫出搜尋結果的結構關係圖，最後把完成的圖形送回使用者端的瀏覽器。

3.2 Query Preprocessor

由於 CGD 畫圖演算法所處理的圖形必須是 Hasse graph，我們必須先分析透過搜尋引擎所得到的搜尋結果，建構圖形結構，並將其簡化為 Hasse graph。上述搜尋結果的分析以及關係圖形的建構，皆是由 query preprocessor 來完成。

Query preprocessor 是由分佈在 VISE 所在主機以及使用者瀏覽器的兩個部分共同組成。將 query preprocessor 分佈在遠端的兩部機器上，主要的原因有下列三項：

- 減少網路頻寬的使用。由主機端的 query preprocessor 先將搜尋結果經過分析，並傳送給使用者端較為簡短的資料結構，佔用的網路資源較直接將搜尋引擎所傳回的網頁內容傳給使用者為少。
- 減少使用者端處理資料的時間。將主要的搜尋結果分析工作，交由可與搜尋引擎結合的 VISE 處理，可由搜尋引擎快速取得資料並分析，使用者端的工作只有接收搜尋結果的資料結構，進而建構圖形。
- 由於使用者端由主機端所得到的是原始的搜尋結果資料結構，在使用者對於圖形作任何控制的時候，可直接於使用者端取得原始資料結構，並建構新圖形，而不須再向主機端要求任何資料，減少作業所花費的時間。

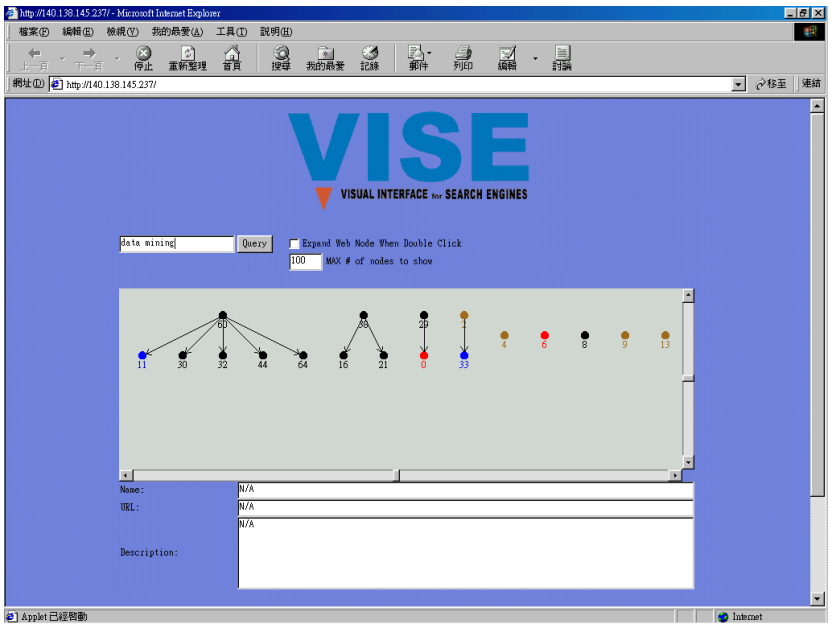
VISE 所在的主機端部分，負責與搜尋引擎溝通並取得搜尋結果，接著分析搜尋結果，將所得的資料結構整理完畢並傳送至使用者端，即使用者的瀏覽器。

使用者端部份在接收到資料結構後，負責建構搜尋結果關係圖，再將此關係圖結構交由 CGD-based graph engine 將圖畫出。以下將介紹 query preprocessor 兩個主要的工作。

3.2.1 分析搜尋結果

分析搜尋結果的工作一共分為兩個部分。第一是尋找所有搜尋結果之間是否有超連結互相連接，這代表互相連接的搜尋結果在網頁內容上存在有某程度的關連性；第二是在搜尋結果中尋找是否有屬於相同網站的網頁，將屬於相同網站的網頁集合在一起可避免最後所畫出的圖形過於複雜。

為了檢查每一項搜尋結果的超連結是否連結到搜尋結果中的其他項結果，我們會連線至代表每一項搜尋結果的網頁並取得網頁內容，檢查網頁中是否有超連結連接至搜尋結果中其他的網頁，並記錄下這些資訊。



圖三：查詢 “data mining” 字串的結果圖形

除了搜尋結果之間的超連結關係，在搜尋結果中，可能有數項結果實際上是屬於同一個網站下的網頁，我們必須找出這些網頁，並封裝可能屬於同一網站的節點為一個單一節點。此單一節點稱為「代表節點」（delegate node），其中所包含的節點可能是屬於同一網站中的不同網頁。在畫圖時可藉由這些代表節點的資訊，暫時忽略屬於相同網站的節點，並以其中一個節點來代表，這是為了增加圖形能顯示的節點數，並且顧及圖形呈現上的美觀及清晰的目的。

3.2.2 建構圖形演算法

建構圖形時分別需要處理節點與線段兩部分。在建構節點的部分，我們將依照由主機端所傳來的代表節點資訊，先忽略「代表節點」所包含的所有節點，另外再根據最大顯示節點數的設定，決定此次畫圖中可以被顯示的節點。

由於經過分析搜尋結果所得到的有向圖中可能包含有環形路徑（circular path）。這些環形路徑如果不先處理，將使繪圖演算法相當複雜。因此在建構線條的同時，我們先將造成環形路徑的 edge 移除，以消除圖中的環形路徑，等到節點都大致安置到適當位置，再將必須的環形路徑補上，因此環形路徑的數目也將減低。這也符合 Eades 及 Xuimin [8] 所提出的第一項畫圖基本準則的需求，也就是圖形中盡力減少環形路徑的產生。

移除造成環形路徑的 edge 後，圖中可能還有 transitive edge 需要移除。所謂的 transitive edge 是指對於圖中某條 $\overline{AB}$，我們必須檢查圖中是否有另一條路徑由 A 點出發，經過其他的節點而到達 B 點，若是這條第二路徑存在，我們便將 $\overline{AB}$ 由圖中移除。$\overline{AB}$ 也就是 transitive edge。

經過上述兩步驟，所得到的新有向圖即為一個沒有包含任何環形路徑以及 transitive

```
InputGraph()
{   NodeSelected=number of selected nodes;
    NodetoShow=MAX number of nodes to show;
    i=0;
    rCount=number of results;
/*insert node into graph*/
While (NodeSelected<NodetoShow && i<rCount){
    If(result[i] is not contained by other results)
       insert result[i++] into graph G;
}
/*insert edge into graph*/
For each node Node[i] that is inserted
    For each node Node[j] linked from Node[i] and
     Link[j] was inserted
       E=edge(Node[i], Node[j]);
       If(E does not cause a loop path or is not a
            transitive edge)
         insert E into graph G;
    }
}
```

圖四：建構圖形的演算法

edge 的 Hasse graph，接著便可利用 CGD-based graph engine 將此關係圖畫出。圖三即為查詢「data mining」字串所得到的搜尋結果關係圖。圖四則為建構圖形的演算法。

3.3 介面操作

由於搜尋結果的關係結構透過了圖形完整的呈現，使用者可經由圖形瞭解搜尋結果間的關係，並快速地決定瀏覽的順序。當使用者選擇圖形上任何一個節點時，有關於代表這個節點的網頁的資訊將會被顯現在整個圖形的下方，其中包括網頁名稱、關於此網頁的簡短描述以及網頁的 URL。透過雙擊滑鼠動作選項的選擇，使用者可在圖形中的節點上雙擊滑鼠，選擇另開視窗瀏覽代表此節點的網頁內容，或是將被選擇的代表節點展開。

在呈現搜尋結果時，為了降低圖形的複雜度，讓使用者能更容易掌握搜尋結果的結構，因此屬於同一網站的網頁會被封裝為「代表節點」（delegate node），這種節點在圖形中是依照所包含網頁的數量不同而以不同顏色來表現：

- 只有包含二個網頁的節點以藍色表示。
- 包含三或四個網頁的節點以褐色表示。
- 包含五個或五個以上的網頁時，則以紅色節點表示。

為了讓使用者可以快速分辨出代表節點之間的不同，並且避免過多的圖形資訊與其代表意義的對應造成使用者的負擔，我們只以藍色、褐色以及紅色三種顏色分別代表包含網頁數量少、中等以及多三種代表節點。而又由於大部分的搜尋結果中，屬於同一個網站的網頁數量並不會太多，因此便以上述三個區間來作為我們分類的標準。而為了避免在圖形重繪之後，使用者無法分辨哪些節點為展開後所看見的新節點，在展開的同時，原本被包含的節點與原先代表此網站的單一節點之間，會以灰色的線條連接，讓使用者容易識別。

四、實驗評估與結果

為了評估 VISE 對使用者的的實用效能，我們設計並進行了二個實驗，來比較 VISE 與傳統條列式呈現方式上的優劣。我們是以實地操作以及問卷的方式，讓使用者使用 VISE 及現有的搜尋引擎，從而比較 VISE 與搜尋引擎傳統條列式呈現之間的優缺點。在實驗中，我們選用 Google 搜尋引擎作為比較對象，並且也以它為 VISE 後端的搜尋引擎，以減少因從不同搜尋引擎所獲得結果而讓使用者存有不同使用習慣上的偏差。這也由於 Google 不論在搜尋速度上，或是搜尋精準度上，都具有相當良好的表現，也是普遍被使用者所廣泛使用並具有知名度的搜尋引擎，因此在我們實驗中，即以 Google 搜尋引擎為選擇。雖然我們並未以其他搜尋引擎再進一步驗證，但只要是與類似 Google 一樣具有條列式呈現的搜尋引擎相比，應都有極為類似的結果。以下將介紹評估實驗的方法與結果。

4.1 實驗方法

我們所提出的視覺化介面 VISE，主要目的在於加速使用者瀏覽搜尋結果的速度，縮短使用者找到所需資訊的時間。所以我們所選擇的評估方式，主要針對一般的搜尋引擎傳統條列式呈現與 VISE 視覺化呈現兩者在搜尋速度上的比較。因此我們決定採取實地操作的方式，徵求志願者分別使用 VISE 以及 Google 搜尋引擎來查詢。實驗結束後我們讓參與實驗者填寫問卷調查表，對於 VISE 與 Google 兩者在四個方面進行評分，作為評估 VISE 在實用上，是否可以幫助使用者更快速地搜尋所需的資訊：

- 首先是「找到所需要重要資訊的速度」。這是指找到所需資訊的過程所經過的時間長短。我們所評估的過程是從使用者看到搜尋結果之後，到使用者認為找到所需資訊為止所用的時間。這是由於我們所實作的 VISE 雛形系統是建構在一部單獨的主機上，而不是如理想中與搜尋引擎做直接的結合，因此我們所採用的實驗方式，並不考慮等待 VISE 取得搜尋結果，並經過分析處理等動作時所必須等待網路傳輸所花費的時間。
- 其次是「是否能找到使用者所需要的資訊」。這是指使用者所看到的搜尋結果，是否真為使用者所查字串的相關資訊。雖然我們所實作的 VISE 雛形系統用來畫圖的搜尋結果是由 Google 獲得，與使用 Google 查詢同樣字串所得的結果一樣。然而 VISE 會將搜尋結果中具有互相連結關係的群組優先顯示在所有搜尋結果之前，所以 VISE 與 Google 所呈現的結果將不盡相同，因此我們將其列入評分的項目之一。

表一：實驗一的總體結果。

評分項目	VISE				Google			
	平均	標準差	最高分	最低分	平均	標準差	最高分	最低分
找尋所需資訊的速度	8.05	0.72	9	7.5	7.2	0.87	9	6
是否找到重要資訊	8.3	0.64	9	7	7.6	1.2	9	3
清楚顯示結果	7.9	0.94	9	7	7.4	1.28	9	5
搜尋結果之間關係	8.5	0.67	10	8	6	1.55	9	4

表二：實驗二的總體結果。

評分項目	VISE				Google			
	平均	標準差	最高分	最低分	平均	標準差	最高分	最低分
找尋所需資訊的速度	8.1	0.83	9	6	7.2	0.87	9	6
是否找到重要資訊	8.4	0.66	9	7	7.5	0.81	9	6
清楚顯示結果	8.2	0.75	9	7	7.4	0.92	9	6
搜尋結果之間關係	8.4	0.49	9	8	6.4	1.43	9	4

- 第三項是「搜尋結果的呈現是否清楚」。在搜尋結果的表現方式方面，由於 VISE 採用圖形的表現方式，不同於 Google 傳統式的條列式表現，因此搜尋結果的顯示是否清楚、使用者是否可以容易地瀏覽搜尋結果，必須列入評分的項目。

- 最後是「是否顯示出搜尋結果之間的關係」。這是指是否能表現出搜尋結果之間的關係。使用者是否能夠透過搜尋結果的表現，進而得到搜尋結果關係結構的相關資訊，也必須列入評分的項目中。

為了評估 VISE 視覺化介面在實用上的效能，我們採取使用者行為分析的模式，並徵求了十位志願者參與實地操作的評估。所有志願者都是來自於元智大學資訊工程研究所的研究生，其中包括三位女性與七位男性，年齡皆介於 24 到 26 歲之間，十位志願者平常使用搜尋引擎的目的主要在於尋找研究相關的資料。

這十位志願者平時經常使用搜尋引擎作為查詢資料的工具，並且對於搜尋引擎的操作皆有一定程度的認識，在經過詳細的解說與閱讀過使用說明，並試用 VISE 查詢資料數次之後，對於 VISE 的操作與目的也都有相當程度的瞭解。因此可以對於 VISE 與 Google 的比較上有較為客觀的評估。

評估實驗分為兩個部分，而實驗的結果是以問卷調查的方式來完成。在參與實驗者開始實驗之前，我們會先發下兩份調查表，每一位參與實驗者在兩部分的實驗過程中需各完成一份評分調查表。評分的方式是以十分為滿分、一分為最低分。

實驗一中列有五個我們事先使用 VISE 查詢過的字串，並且可以在五個字串的搜尋結果關係圖中得到有用的資訊。這五個字串分別是「thesis writing」、「active network」、「proxy」、「NBA」、以及「wireless」。參與實驗者必須使用 VISE 於五個字串中挑選至少三個進行查詢，所挑選的三個字串也必須使用 Google 查詢過一次，最後就 VISE 與 Google 的表現給予評分。

實驗二是讓參與實驗的十位志願者自行決定查詢的字串，以 VISE 跟 Google 各查詢過一次。參與實驗者必須將查詢過的字串記錄在調查表中，並在調查表的最後給予評分。

4.2 實驗結果及討論

實驗結果如表一與表二所示，表中的分數是十位參與實驗者所給予的評分的平均值。我們可以由表中的分數發現，VISE 的得分都較 Google 介面為佳，尤其在「搜尋結果的關係是否表現」這個項目上。

由於實驗一所提供的五個字串，是已經經過我們使用 VISE 查詢過，並且確定可由搜尋結果關係圖中快速找到與字串相關的資訊。因此使用者在「找尋所需資訊的速度」這個項目中，給予 VISE 較高的評價，而且這也符合我們的預期。

在「是否找到重要資訊」這個項目，同樣是因為在我們所提供的五個字串的搜尋結

果關係圖中，優先顯示在圖形前方的節點都包含了與字串相關的重要資訊，因此參與實驗者在這個項目上也給予 VISE 較高的評價。由此也可看出當呈現出節點關係結構後，VISE 的確能較為幫助使用者找到重要資訊。使用者也的確能夠從中獲益，能夠在較短時間內迅速找到他們所需要的重要資訊。

但在「搜尋結果的顯示是否清楚」方面的評分結果，我們發現這項結果需要進一步討論，因為在這個項目中，VISE 與 Google 的分數差距並不大。我們發現這是由於以下兩個原因所造成。第一，VISE 無法同時顯示好幾個節點的網頁相關資訊，但是在條列式呈現方式的輸出結果中，每一項目都加入預先瀏覽部分內容，因此透過這些條列出「預覽」的方式，使用者反而不覺得視覺化呈現的方式能提供較清楚的結果顯示。第二，在圖形顯示時，若在一個連通元件中的每一個節點皆具有大量的超連結，在圖形中會出現大量線條交錯而導致混亂的情況，造成顯示上的不清楚。由於以上兩個原因，VISE 領先的幅度很小。

在最後一項評估項目中，VISE 的平均分數大幅領先二分以上，這是由於使用者幾乎無法由 Google 的條列式結果中，瞭解有關搜尋結果之間的關係結構，只能藉由相關的網頁資訊來判斷是否此網頁中包含使用者需要的資訊。而透過 VISE 的搜尋結果關係圖，我們將互相連結的節點優先顯示，使用者可以利用圖形的顯示，直覺地決定瀏覽搜尋結果的順序，加快使用者搜尋的速度，因此也導致在其他三項的評估項目中，VISE 所得到的分數皆較 Google 為高。

實驗二的實驗環境接近於一般使用者日常使用的情況。唯一的差別是實驗時，操作的時間較短。而結果也幾乎與實驗一類似。但是在「搜尋結果的顯示是否清楚」方面上，此時 VISE 又更稍微領先 Google。經過我們的追蹤探討，發覺當使用者習慣視覺化呈現後，偏好這類方式的使用者更給予 VISE 相當高的評價。我們推測這是因為視覺化的效果對於使用者的影響較深刻。不過這部分仍然需要在未來進一步探討。

透過實驗中的使用者評估調查，我們確實發現在四個評分項目中，VISE 所得到的評價都較高，這也證明 VISE 在利用搜尋引擎找尋資料上，可以幫助使用者更為快速並準確地找到所需的資訊。其他詳細的分項結果及相關討論，囿於篇幅，請見 [1]。

五、結論

我們所提出的視覺化介面 VISE 介於使用者與搜尋引擎之間，分析搜尋引擎所傳回的搜尋結果，並將其結構關係圖畫出，以圖形代替傳統的條列式文字表示方式。使用者可由搜尋結果關係圖中看見彼此互相連結的節點集合，由於這些節點中可能包含有使用者所需要的重要資訊，使用者可以快速決定瀏覽的順序，因而減少瀏覽的時間，更快地找到使用者所需要的資訊。另外在瀏覽搜尋結果關係圖的過程中，由封裝屬於相同網站的節點而得到的「代表節點」（delegate node），是以其所包含的節點數量不同而以不同的顏色表示，使用者可以選擇並展開代表節點，以得到更為詳細的關係結構。

為了評估 VISE 的實用效能，我們徵求十位志願者以實地操作以及問卷的方式，來比較 VISE 與 Google 之間的優缺點。實驗的結果顯示，VISE 的表現皆較條列式呈現方式為佳，尤其在「搜尋結果之間關係」的顯示方面更有大幅領現的表現。因此隨著這個項目的領先，也間接導致 VISE 在其他三個項目的結果有良好的表現。雖然參與評估實驗的人數較少，但是由實驗結果可以發現，VISE 在使用者瀏覽搜尋結果的時候的確能提供幫助，讓使用者能夠提高瀏覽速度並減少找到所需資訊所耗費的時間。

由於網際網路的資訊量暴增，如何能有效率地管理與存取搜尋引擎所傳回的大量搜尋結果，是亟待研究的問題。本論文中所提出的視覺化介面-VISE，以圖形代替傳統的條列式文字敘述，減少使用者瀏覽搜尋結果並找到所需資訊的時間，相信能為未來搜尋結果視覺化方法的發展提供幫助，讓使用者能以更快的速度在網路上找到所需要的資訊。

六、誌謝

本論文研究為國科會補助經費之研究成果，計畫編號為 NSC90-2213-E-155-018。

七、參考文獻

[1] 楊錫謦， “Web 搜尋結果視覺化”，元智大學資訊工程研究所，碩士論文，2001。

[2] B. Amento, W. Hill, L. Terveen, D. Hix, and P. Ju. “An Empirical Evaluation of User Interfaces for Topic Management of Web

Sites". In *Proceedings of the CHI 99 Conference on Human Factors in Computing Systems*, pp. 552–559, May 1999.

[3] R. Beale, R. J. McNab, and I. H. Witten. "Visualizing Sequences of Queries: A New Tool for Information Retrieval". In *Proceedings of 1997 IEEE Conference on Information Visualization*, pp. 57–62, August 1997.

[4] G. D. Battista, P. Eades, R. Tamassia, and I. G. Tollis, "Graph Drawing - Algorithms for the Visualization of Graphs," Pentice Hall Inc., 1999.

[5] S. Brin and L. Page. "The Anatomy of a Large-Scale Hypertextual Web Search Engine." In *Proceedings of the 7th International World Wise Web Conference*, 1998.

[6] J. Carriere and R. Kazman. "WebQuery: Searching and Visualizing the Web through Connectivity". In *Proceedings of the 6th International World Wide Web Conference*, pp. 701–711, April 1997.

[7] Y.-F. Chen and E. Koutsofios, "WebCiao: A Website Visualization and Tracking System", In *WebNet97*, 1997.

[8] P. Eades and L. Xuemin. "How to Draw a Directed Graph". In *Proceedings of the IEEE Workshop on Visual Languages*, pp. 13–17, October 1989.

[9] S. Lawrence and C. L. Giles, "Context and Page Analysis for Improved Web Search," *IEEE Internet Computing*, Vol. 2, No. 4, pp. 38-46, July/August 1998.

[10] C. L. McCreary, R. O. Chapman, and F.-S. Shieh. "Using Graph Parsing for Automatic Graph Drawing". *IEEE Transactions on Systems, Man and Cybernetics, Part* A, Vol. 28, No.5, pp. 545–561, September 1998.

[11] S. Mukherjea and Y. Hara. "Visualizing World-Wide Web Search Engine Results". In *Proceedings of the 1999 IEEE International Conference on Information Visualization*, pp. 400–405, July 1999.

[12] J. Williams and R. Starzl, "Tuning Up the Search Engine," *IEEE IT Professional*, Vol. 3 No.1, pp. 60-62, January/February 2001.