一個新的文件分群技術運用於網頁搜尋結果

A Novel Document Clustering Algorithm Applied to Web Search Results

陳林志
國立東華大學資訊管理學系助理教授
lcchen@mail.ndhu.edu.tw

摘要

本論文提出一個新的文件分群演算法(稱之為 OTFDC)，其可以應用至多國語言文件，我們也會計算查詢與所有分群之間的關聯係數。OTFDC 具有下列幾項特性：(1)它能夠應用至多國語言環境，(2)它可以改善並增進任何搜尋引擎效能，(3)它能夠自動學習新知識，(4)它可以快速回應使用者的需求。

在本論文裡，我們將進行使用者評比及模擬評比。使用者評比是以熱門關鍵字進行搜尋，並計算查詢與各個分群之間的關聯程度。模擬評比則是針對 AMPRE (Average of Mean of Precision Rate)及 AMREC (Average of Mean of Recall Rate) 指標來進行評比，而模擬評比的對象則是隨機產生 1000 個關鍵字。根據模擬結果，我們的結果明顯優於目前提供分群之搜尋引擎。

關鍵詞：網頁搜尋、文件分群、語意關係、OTFDC

Abstract

In this paper, we proposed a novel document clustering algorithm, called OTFDC, which can be applied to a multilingual document. We will calculate the correlation coefficient between the user's query and the candidate clusters. The OTFDC algorithm owns the following properties: (1) it can be applied to a multilingual environment; (2) it can improve the search engine performance, (3) it can automatically learn new knowledge; (4) it can quickly response to the user's needs.

This paper evaluates the performance of OTFDC by the user's experiment and some simulations. The user's experiment is focused on the popular keywords on the Internet. The simulation criteria used in this paper are AMPRE (Average of Mean of Precision Rate) and AMREC (Average of Mean of Recall Rate). Simulation scenario is randomly generated 1000 keywords. According to the results of the user's experiment and the simulations, our clustering results are significantly better than current search engines with document clustering.

Keywords: Web Search, Document Clustering, Semantic Relation, OTFDC

1. 前言

近年來，網際網路快速發展，經統計至 2008/1/7 被發現的網頁數目已經超過 2,536,000,000[21]。因此，如何在這麼多的網頁找出相關且有用的資訊變成一件重要且困難

的工作。搜尋引擎主要用來設計完成數十億使用者的搜尋需求[27, 1]，當使用者執行搜尋時，搜尋引擎將會回傳相當多的資訊。然而，搜尋引擎並無將這些回傳資訊執行進一步整理，以致使用者必需花費相當多的時間在判斷搜尋結果的過程上。在現今網際網路環境之中，每個網頁皆具有一定主題，因此，我們需要一種技術能夠劃分網頁，使所劃分的結果具有主題分群，並將此分群結果顯示給使用者。

文件分群是一種能將文件自動分群至相關主題的技術[2]，根據主題進行組織文件能幫助使用者快速找到有興趣的資訊。文件分群技術主要可分為階層法及分割法[2, 16, 17, 33, 28, 7, 32]。

階層法主要是透過聚合法(由下而上)或分裂法(由上而下)產生分群樹(亦稱為dendrogram)。聚合法從單一文件分群開始並且重覆的合併兩個以上的適當分群。分裂法從一個包含所有文件的大分群開始並且重覆的分離適當分群。許多演算法屬於階層法，比方：HAC[39]，BIRCH[46]，CURE[10]，ROCK[11]，BHC[13]以及 DIVCLUS-T[5]。

分割法主要是透過分割的過程，將所有文件分割成為預先定義集合的方式進行分群，其中每個集合代表一個分群。分割程序根據某些特定的評估函數建立良好的分群，而此評估函數主要是達成群組內高度相似，群組間低度相似。許多演算法屬於分割法，比方：K-means[25]，PAM[19]，CLARA[19]，Fuzzy c-means[3]，ASI[22]以及 k-Attractors[18]。

在本論文中，我們提出一個新的文件分群演算法，稱之為 OTFDC (On-The-Fly Document Clustering)。OTFDC 是根據群組間存在關聯語意關係建立而成，同時在此關聯語意關係上，我們只關心優良品質的關係(相關定義，請參照第三節)。OTFDC 具有下列特性：它可以應用至多語文件及其它搜尋引擎之上，未監督式學習方式自動學習新知識而不需要使用任何參考文集，同時分群結果可以即時顯示，這特別符合現代搜尋引擎要求即時反應的特性。

本論文的其它部份組織如下：第二節之中，我們將介紹相關文獻。第三節，我們將討論 OTFDC 演算法。第四節，我們將比較 OTFDC 與其它搜尋引擎之間的效能評比，評比的方式包含兩大類：使用者評比及模擬評比。最後，第五節將會討論本論文的未來研究方向。

2. 文獻探討

在本節中，我們將會對文件分群進行文獻探討。搜尋類似文件在文件管理是一件相當重要的工作，類似文件被群組起來，使得管理及搜尋文件可以變得更加有效率。因此，當文件個數眾多時，我們需要一個較佳的策略以取得類似文件。文件分群是可以將類似文件進行分群，其中類似是透過一些函數計算而得，通常透過文件分群的過程，可以增加資訊擷取系統的效能及效率。然而，文件分群是一件困難的工作，因為文件具有相當的非結構性[35]。目前研究文件分群嘗試使用不同觀點設計不同分群演算法。

部份研究著重在改善文件分群架構。Nakada 及 Osana[30]使用主題圖(subject graph)架構表達文件分群的主題。在主題圖之中，節點代表文件，邊線代表文件之間所存在的關係。接下來，個別主題圖被整合至更大的主題圖。最後，所產生的最後主題圖即代表文件分群結果。Wang 等[42]提出 CLGR (Clustering with Local and Global Regularization)演算法，其可以對每一個文件建立 local regularized 線性標籤預測器。接下來，CLGR 組合上述所有線性標籤預測器成為 global smoothness 規則。Wan 及 Yang[41]提出 CollabSum (Collaborative single document Summarization)架構，此架構能透過單一分群之中多個文件相互不影響的特性進行文件彙總。

部份研究著重在使用 fuzzy 觀念設計文件分群演算法。Tjhi 及 Chen[37]設計 PFCC (Possibilistic Fuzzy Co-Clustering)演算法，此演算法能對大量文件進行自動分類，PFCC 組合機率式文件分群及fuzzy單字排名公式技術。Tjhi 及 Chen[36] 提出 FCR (Fuzzy Co-clustering with Ruspini's condition)演算法，其可以產生 fuzzy 單字分群，此分群可以擷取單字的自然分佈情形。Saraçoğlu 等[34]設計兩層次的文件處理架構，每份文件透過這兩層次架構計算彼此間的相似度。接下來，使用預先定義的 fuzzy 分群去取出文件相關的特徵向量，並以此特徵向量找出類似文件。

部份研究著重以查詢觀念設計文件分群演算法。Chang 等[4]設計 QCM (Query Concept Method)演算法，此演算法嘗試建立查詢概念以便表達使用者的需求，而非透過修改起始查詢，同時其不需要使用者回饋機制。接下來，QCM 來源即是這些基礎查詢概念，並以此基礎查詢進行文件分群。Na 等[29]提供以查詢概念所建立而成的調整式文件分群演算法，他們宣稱此調整式文件分群可以當成找尋類似文件的過程。Li 等[24, 23]提出 CFWS (Clustering based on Frequent Word Sequences)及 CFWMS (Clustering based on Frequent Word Meaning Sequences)演算法，其考慮在所有文件之中，每個查詢序列出現的次數是否超過一定比例的百分比，他們宣稱常用的查詢序列可以提供文件有用資訊。

部份研究著重在選擇適當的文件分群參數。Maggini 等[26]提出 PCMD (Pseudo Concept Matrix Decomposition)及 PSVD (Pseudo Singular Value Decomposition)演算法，其可以透過回饋方式微調分群參數，此回饋方式是對每一份文件使用適當的參數，他們宣稱當回饋參數選擇合適的話，PCMD 及 PSVD 演算法可以自動組織文件至同類的分群之中。Yang 等[44]嘗試從起始擷取的排名列表中的每一份文件取出部份虛擬標籤參數。接下來，以此標籤參數取出類似文件。Niu 等[31]嘗試選擇適當的分群驗證參數，此參數能夠正確識別文件中的重要特徵集合及分群個數，他們宣稱當分群驗證參數選擇適當的話，文件分群的精確度將會比傳統分群演算法更優良。

3. OTFDC 演算法

在本節裡，我們將提出一個新的文件分群演算法，稱之為 OTFDC (On-The-Fly Document Clustering)。OTFDC 所產生的分群結果，具有使用者查詢及分群結果存在語意關係。

定義 1(語意關係)：我們稱($Cluster_i$, $Cluster_t$)關係具有語意關係，當 $Cluster_i$ 及 $Cluster_t$ 具有下列三項關係中至少一項關係：等價，階層以及聯想關係[45]。當兩個分群所要表達是相同概念時，此兩個分群存在等價關係，例如：("椅子", "傢俱")以及("影印機", "全錄")分別存在等價關係。階層關係是根據依屬關係的程度建立而成，上層關係表示部份或全部分群，下層關係表示分群的成員或元件，例如：("山脈", "阿爾卑斯")以及("神經系統", "大腦")分別存在階層關係。聯想關係代表分群之間存在語意或概念聯想關係，但此關係不屬於等價或階層關係，同時這個關係必須被顯露出來，例如：("麵粉", "小麥")以及("希臘文明", "蘇格拉底")分別存在聯想關係。

OTFDC 使用遞迴方式產生與使用者查詢有關的分群，OTFDC 演算法的精神：透過分群找出更大的分群集合，並且透過遞迴呼叫的方式擴大此分群集合。OTFDC 將重複執行下列兩個步驟，直到找不到其它適合的分群為止。

(1) 透過舊的分群找出新的分群集合。
(2) 將新的分群集合變為舊的分群。

B-tree 是一種以樹形式表示之資料結構，其可以將資料儲存並確保搜尋、插入、刪除等動作於對數時間完成，其被廣泛的應用於資料庫[20]。B-tree 可以取出樹中較少層數的資料並且將較少的處理資料放進於記憶體處理，因此 B-tree 可以加速搜尋的速度[8]。因此在 OTFDC 演算法之中，我們選擇 B-tree 為我們實作文件分群演算法的資料結構。OTFDC 演算法描述如下：

1 Algorithm OTFDC ($Cluster_i$)
2 {
3 呼叫 OTFDC_Core($Cluster_i$)，以便產生其它分群($Cluster_t$)，並計算 $Cluster_i$ 與($Cluster_t$) 的關聯係數 ($R(Cluster_i, Cluster_t)$);
4 Foreach ($Cluster_t$)
5 {
6 將 ($Cluster_t$) 加入 $\{RK_{i,t}\}$，$R(Cluster_i, Cluster_t)$加入$\{RR_{i,t}\}$;
7 呼叫 OTFDC ($Cluster_t$);
8 } End of Foreach;
9 根據$\{RR_{i,t}\}$排序$\{RK_{i,t}\}$;
10 Return($\{RK_{i,t}\}$,$\{RR_{i,t}\}$);
11 } End of Algorithm;

OTFDC 透過每次呼叫 OTFDC_Core 的方式產生其它分群，其中 OTFDC_Core 的演算法顯示如下：

1 Algorithm OTFDC_Core ($Cluster_i$)
2 {
3 呼叫 SVV ($Cluster_i$)，以便產生符合 $Cluster_i$ 的所有文件($\{WP_{i,m}\}$)[1];
4 計算所有$\{WP_{i,m}\}$其所對應的權重($\{w_{wp_{i,m},i}\}$);
5 根據$\{w_{wp_{i,m},i}\}$排序$\{WP_{i,m}\}$;
6 Foreach ($WP_{i,m}$)
7 {
8 找出目前$WP_{i,m}$所存在的其它分群($Cluster_t$);
9 當 $w_{wp_{t,m},t} \geq T$ 時，將 $Cluster_t$ 加入$\{CK_{i,t}\}$;
10 } End of Foreach;
11 根據$\{w_{wp_{t,m},t}\}$排序$\{CK_{i,t}\}$;
12 將 $Cluster_i$ 的權重$w_{wp_{i,m},i}$正規化，即 $Norm_i = w_{wp_{i,m},i} / \sum_{s=1}^{4} \alpha_{s,i}$;
13 Foreach ($CK_{i,t}$)
14 {
15 將 $Cluster_t$ 的權重$w_{wp_{t,m},t}$正規化，即 $Norm_t = w_{wp_{t,m},t} / \sum_{s=1}^{4} \alpha_{s,t}$;
16 $R(Cluster_i, Cluster_t) = \min(Norm_i, Norm_t)/Norm_i$;
17 If ($R(Cluster_i, Cluster_t) \geq T$)
18 {
19 將$CK_{i,t}$加入$\{ReturnCK_{i,t}\}$;
20 將 $R(Cluster_i, Cluster_t)$ 加入 $\{ReturnR(Cluster_i,Cluster_t)\}$;
21 } End of If;
22 } End of Foreach;
22 Return($\{ReturnCK_{i,t}\}$,$\{ReturnR(Cluster_i, Cluster_t)\}$);
23 } End of Algorithm;

定義 2(高度語意關係)：我們稱($Cluster_i$, $Cluster_t$)關係存在高度語意關係，若且唯若($Cluster_i$, $Cluster_t$) 具有語意關係而且 $T \leq R(Cluster_i, Cluster_t) \leq 1$，其中 T 根據我們模擬的結果為 0.74875[2]。這代表 $Cluster_i$ 及

[1] 根據我們先前的研究[6]，SVV 的效能優於現行搜尋引擎，因此我們以 SVV 的搜尋結果，當成 OTFDC 的來源。當然，OTFDC 也可以使用其它搜尋引擎的搜尋結果為其處理來源。

[2] 由於研討會論文篇幅的限制，我們需捨去相當多的章節，比方 T 值討論時，我們只列出我們模擬的 T 值=0.74875。我們主要隨機選定 1000 個使用者查詢，並模擬個別查詢所求得所有分群結果的平均 MMR 參數[38](稱之為 AMMRR)，最後再對這 1000 個查詢的 AMMRR 平均之後，求得最後 T 值。

$Cluster_t$ 不只存在語意關係而且彼此的關聯係數大於等於下限值(T)。例如：(“王建民”, “mlb”)具有高度語意關係，因為我們可以找到”**王建民在 mlb 工作**”關係，而且此兩個分群的關聯係數 ($R(key_i, key_t)$ =100%) 大於等於下限值 (T=0.74875)。

接下來，我們以數學模式來表達 OTFDC 所具有的另一項特性：關聯關係。令 R 代表所有分群的集合，此處 $Cluster_i$ 及 $Cluster_t$ 存在一條關聯路徑長度為 n，若且唯若 $(Cluster_i, Cluster_t) \in R^n$。關聯關係$R^*$包含一對分群$(Cluster_i, Cluster_t)$，使得 $Cluster_i$ 及 $Cluster_t$ 在 R 之中存一條關聯路徑。因為R^n包含一對分群$(Cluster_i, Cluster_t)$，使得其存在一條 $Cluster_i$ 到 $Cluster_t$ 的路徑，長度為 n，這代表R^*將所有 R^n 集合聯集，亦即$R^* = \bigcup_{n=1}^{\infty} R^n$。圖 1 顯示 OTFDC 演算法產生關聯關係$R^*$。

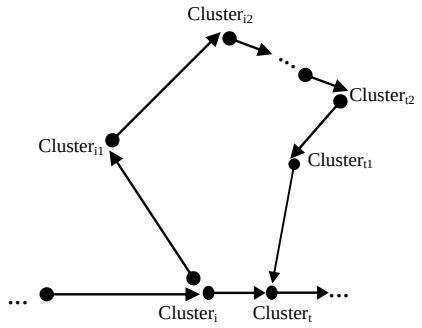

圖 1：$(Cluster_i, Cluster_t)$存在關聯關係R^*

定義 3(關聯高度語意關係)：我們稱$(Cluster_i, Cluster_t)$關係存在關聯高度語意關係，若且唯若$(Cluster_i, Cluster_t)$具有高度語意關係而且$(Cluster_i, Cluster_t)$具有關聯關係。亦即，我們可以找到一個 $Cluster_d$，使得$(Cluster_i, Cluster_d) \in R^*$而且$(Cluster_d, Cluster_t) \in R^*$，其中$(Cluster_i, Cluster_d)$及$(Cluster_d, Cluster_t)$分別具有高度語意關係。例如：(“王建民”, “mlb.com”)具有關聯高度語意關係，因為我們可以找到兩個高度語意關係(“王建民, ”mlb”) $\in R^*$ 以及(“mlb”, “mlb.com”)$\in R^*$。亦即，我們可以找到”**王建民在 mlb 工作**”以及”**mlb 官方網站是 mlb.com**”的關係。

接下來，我們討論 OTFDC 的執行結果。圖 2 顯示當使用者查詢為”王建民”時 OTFDC 所執行的動態分群結果。OTFDC 會依照查詢與分群間的關聯係數(R)進行排序顯示，例如前五個分群與”王建民”的關聯係數 100%。除此之外，我們在分群後面增加星號(✹)，其表示符合下列公式：$R > \bar{R} + n\sigma_R$，其中$\bar{R}$代表所有分群之關聯係數 R 的平均值，σ_R代表代表所有分群之關聯係數 R 的標準差。根據 Chebyshev's 定理 [40]，我們瞭解$R > \bar{R} + n\sigma_R$的機率值小於$1/n^2$。在圖 2 之中，我們設定 n = 3，亦即使用者查詢與個別分群之關聯係數屬於高相關才會顯示星號。

OTFDC 不單單只能對中文文件進行分群，而且還能處理多語文件，目前已經提供可測試系統之中，我們使用中文文件當成 OTFDC 的來源，OTFDC 也可以處理多語文件，當 OTFDC 的文件來源是其它語言。圖 3 顯示當使用者查詢為”p2p”時 OTFDC 的分群結果，此結果 OTFDC 的來源為 Google 的英文搜尋結果。然而，我們不能提供線上測試，因為此英文搜尋結果是有版權的問題。在未來裡，我們將會收集英文網頁，以便提供 OTFDC 英文文件分群系統之線上測試。

圖 2：OTFDC 執行”王建民”分群結果

圖 3：OTFDC 執行"p2p"分群結果

4. 效能評比

在本節裡，我們將評估使用者及模擬實驗的效能。使用者評估著重在網際網路上熱門關鍵字，同時計算關鍵字與各個分群之間的關聯係數。在本論文中，模擬參數使用 Average Mean of Precision Rate (AMPRE)以及 Average Mean of Recall Rate (AMREC)，模擬情形為隨機產生 1000 個測試關鍵字。根據使用者及模擬評估的結果，我們的 OTFDC 的文件分群效能明顯的優於其它搜尋引擎。

4.1 使用者評估

在本評估之中，我們對於使用者如何評估 OTFDC 與其它搜尋引擎的結果感到興趣。我們設計一份網路問卷以供使用者填寫 OTFDC、Google、Yahoo[3]以及 Vivisimo[4]的評比，問卷時間收集時間從 2006/9/1 至 2007/11/30 為止，回收的有效問卷包含 84 份。使用者評估著重在網際網路上常用的關鍵字，此關鍵字是根據 Google Press Center 於 2006 年 8 月所統計的 15 個關鍵字[9]，其分別是：104、王建民、無名小站、白色巨塔、宮野蠻王妃、林志玲、未上市股票、火影忍者、阿嬌、蔡依林、山下智久、foxy、mlb、洋基、博客來。使用者獨立評估每個搜尋引擎的回傳結果，其中包含未提供文件分群(Google 及 Yahoo)以及提供文件分群(OTFDC 及 Vivisimo)的搜尋引擎。我們的問卷是依據 Likert 的 5 點式評估[43]，其中 1 代表高度反對，5 代表高度贊成。

使用者評估的結果顯示於表 1。其中每一格(不包含最後一列)代表每個關鍵字於每個搜尋引擎所回傳結果在此 84 份有效樣本的平均分數，例如：對於"104"，OTFDC 的平均分數為 4.37。最後一列的數字代表每個搜尋引擎在此 15 個熱門關鍵字的平均分數，OTFDC、Google、Yahoo 以及 Vivisimo 的平均分數分別為：4.21、3.26、3.23、3.75。這樣的結果跟我們原來的期望相同，即使用者對於分群結果具語意關係(OTFDC，Vivisimo)優於無語意關係(Google，Yahoo)。另一方面，由於 OTFDC 提供關聯高度語意關係優於Vivisimo 只提供語意關係[5]，即代表我們不單單考慮分群之間所具有的語意關係，而且還考慮彼此需具有高度及關聯關係。因此，我們的 OTFDC 能夠比 Vivisimo 更加優良[6]。

[3] Google 及 Yahoo 目前並無提供語意式文件分群功能，目前只提供關鍵字文件分群功能，比方：當輸入"建民"，他們只會找出包含"建民"之類的相關搜尋結果，這樣代表可能找出："朱建民"、"趙建民"、"建民農場"之類的文件

[4] Vivisimo 連續兩年獲得 InfoWorld 關於資料管理的最佳搜尋引擎[15, 12]，Vivisimo 本身即是強調其語意式文件分群的能力

[5] Vivisimo 並無考慮關聯及高度關係，比方經我們測試：當我們輸入"王建民"，其分群結果包含："Google, Discussions"，"淘寶網"，"中國網"，"中時生活,陳盈珊"，然而實際情形，這些分群結果可能跟"王建民"有語意關係，然而我們實在很難聯想到這些分群與"王建民"存在什麼關係

[6] 此處我們省略統計檢定的詳細說明，在我們實際的統計檢定裡，其中包括 ANOVA 及 LSD 檢定，根據 ANOVA 的結果，$F=4.22899$ 大於 $F^{3}_{10496}(0.05) = 2.60575$(F 分配值)，因此我們拒絕 H_0 假設，即代表 4 個搜尋引擎的效能具有顯著差異，接下來，我們使用 LSD 檢定，判定 OTFDC 明顯的優於 Google、Yahoo、Vivisimo(其中的 LSD 值=5.47236)

表 1：15 個常用關鍵字的滿意度分配 (關鍵字日期：2006 年 8 月)

	OTFDC	Google	Yahoo	Vivisimo
104	4.37	3.42	3.39	3.73
王建民	4.14	3.32	3.22	3.77
無名小站	4.13	3.13	3.16	3.86
白色巨塔	4.28	3.38	3.22	3.59
宮野蠻王妃	4.31	3.13	3.04	3.87
林志玲	4.36	3.18	3.06	3.86
未上市股票	4.29	3.11	3.17	3.68
火影忍者	4.17	3.2	3.26	3.79
阿嬌	4.16	3.27	3.25	3.59
蔡依林	4.12	3.1	3.39	3.88
山下智久	4.14	3.25	3.26	3.67
foxy	4.22	3.26	3.27	3.83
mlb	4.36	3.38	3.19	3.76
洋基	4.03	3.33	3.25	3.71
博客來	4.14	3.43	3.25	3.73
平均	4.21	3.26	3.23	3.75

4.2 模擬評估

在模擬評估之中，我們隨機產生 1000 個測試關鍵字，評估 OTFDC、Google、Yahoo 及 Vivisimo 之 Average Mean of Precision Rate (AMPRE) 及 Average Mean of Recall Rate (AMREC)之評估指標。我們的模擬環境是：Intel Core2Duo E6400，4GB DDR2-667 RAM，Windows 2003 (64 bit)。

首先，為了我們模擬程式可以正確識別正確答案，因此，我們需要定義正確的答案(文件)列表。正確文件列表代表某個查詢 TK_i 在所有搜尋引擎所回傳的第一個文件，比方：當 TK_i ＝”小遊戲”時，OTFDC 的第一個文件=”howkid.com”、Google 的第一個文件=”www.slime.com.tw/mail/game_1.htm”、Yahoo 的第一個文件=”howkid.com”、Vivisimo 的第一個文件=”howkid.com”，因此正確文件列表={“www.slime.com.tw/mail/game_1.htm”, “howkid.com”}。接下來我們使用下列式子來定義 Precision Rate (PRE)及 Recall Rate (REC)：

NC_i = 查詢為 TK_i 時取出及相關文件個數

NE_i = 查詢為 TK_i 時取出之文件個數

NT_i = 查詢為 TK_i 時相關之文件個數

$$PRE_{TK_i} = \frac{NC_i}{NE_i} \qquad REC_{TK_i} = \frac{NC_i}{NT_i}$$

接下來，我們隨機產生 1000 個測試關鍵字進行評比，由於PRE_{TK_i}及REC_{TK_i}都是針對某一特定 TK_i，因此當我們產生的測試關鍵字超過 1 個時，我們需要定義 Mean of PRE (MPRE)以及 Mean of REC (MREC)，圖 4 及 5 表示隨機產生 50 個到 1000 個測試關鍵字的 MPRE 及 MREC 曲線：

$$MPRE = \frac{1}{||\{TK\}||}\sum_{TK_i=1}^{||\{TK\}||} PRE_{TK_i}$$

$$MREC = \frac{1}{||\{TK\}||}\sum_{TK_i=1}^{||\{TK\}||} REC_{TK_i}$$

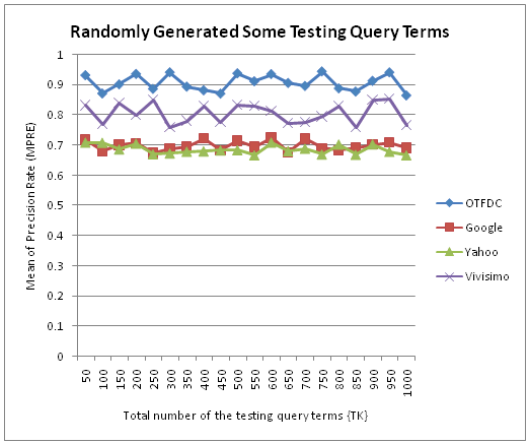

圖 4：隨機產生 1000 個測試關鍵字之 MPRE 曲線

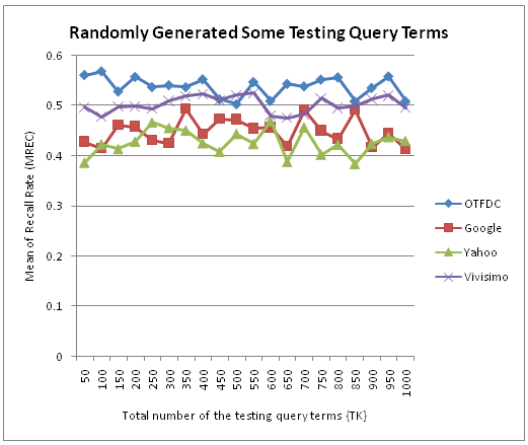

圖 5：隨機產生 1000 個測試關鍵字之 MREC 曲線

在表 2 之中，我們平均所有參數：Average MPRE (AMPRE) / Average MREC (AMREC)，求得 4 個搜尋引擎的 AMPRE 分別為：OTFDC (0.90662)、Google (0.69821)、Yahoo (0.68449)、Vivisimo (0.80583)，4 個搜尋引擎的 AMREC 分別為：OTFDC (0.53765)、Google (0.44833)、Yahoo (0.42708)、Vivisimo (0.50244)。根據上述結果，我們可以驗證 OTFDC 效能不單單優於 Google 及 Yahoo，而且還優於 Vivisimo。

表 2：搜尋引擎間的 AMPRE 及 AMREC 值

	OTFDC	Google	Yahoo	Vivisimo
AMPRE	0.90662	0.69821	0.68449	0.80583
AMREC	0.53765	0.44833	0.42708	0.50244

接下來，為了驗證 OTFDC 可以應用且改善其它搜尋引擎效能，所以我們將 OTFDC 原始的輸入 SVV 分別改成 Google、Yahoo、Vivisimo，此組合引擎分別為：Google-OTFDC、Yahoo-OTFDC、Vivisimo-OTFDC。表 3 分別代表 Google、Google-OTFDC、Yahoo、Yahoo-OTFDC、Vivisimo、Vivisimo-OTFDC 的 AMPRE 及 AMREC 結果。明顯地，我們發現 Google-OTFDC、Yahoo-OTFDC、Vivisimo-OTFDC 皆優於其來源搜尋引擎，此結果是起因於我們採用 OTFDC。

表 3：當分群演算法採用 OTFDC 之後，搜尋引擎間的 AMPRE 及 AMREC 值

	Google	Google-OTFDC	Improved Rate	Yahoo	Yahoo-OTFDC	Improved Rate	Vivisimo	Vivisimo-OTFDC	Improved Rate
AMPRE	0.69821	0.89235	27.81%	0.68449	0.88257	28.94%	0.80583	0.85145	5.66%
AMREC	0.44833	0.55164	23.04%	0.42708	0.54362	27.29%	0.50244	0.52143	3.78%

為了確認 OTFDC 可以應用至多語文件，我們將使用 Google 英文頁面(Google_Eng)而非中文頁面，當成 OTFDC 的來源。在這個模擬情況裡，我們從 MetaSpy[14]隨機產生 1000 個英文測試關鍵字。表 4 分別代表 Google_Eng、Google_Eng-OTFDC、Vivisimo 的 AMPRE 及 AMREC 結果。我們發現，Google_Eng-OTFDC 所提供的關聯高度語意關係仍然優於 Vivisimo 所提供的語意關係，亦即代表 OTFDC 在處理多語文件時仍優於現行搜尋引擎。

表 4：OTFDC 的來源為 Google 英文頁面時的比較

	Google_Eng	Google_Eng-OTFDC	Vivisimo
AMPRE	0.71245	0.91426	0.81324
AMREC	0.46213	0.55234	0.50477

5. 結論及未來方向

在本論文之中，我們提出一個 OTFDC 文件分群演算法，此演算法不只能處理中文文件，而且可以應用到多國語言文件。OTFDC 分群的結果具有關聯高度語意關係，這個特性對於多國語言環境特別有用，比方：當使用者查詢為"王建民"，分群結果可能出現"mlb"，以實際的觀點來看，"王建民"是台灣裡最有名的棒球球星，而且其目前在 MLB(美國職棒大聯盟)工作，根據我們評估的結果，OTFDC 能提供較好的效能。其次，為了驗證 OTFDC 也可以改善現在搜尋引擎，我們也模擬了當 OTFDC 應用至其它搜尋引擎之後的效能，模擬結果顯示 OTFDC 確能超越其原始搜尋引擎的效能。最

後，由於 OTFDC 是動態的產生分群，而分群結果會根據原始文件的分佈情形來產生，亦即當原始文件是最新文件資訊時，OTFDC 將會自動產生新的知識，而不用透過輸入任何文件集的動作。另一方面，雖然 OTFDC 是一個遞迴演算法，然而我們透過兩個步驟來達成即時回應效果：(1) 資料結構採用 B-tree；(2) 利用 T 值來控制高度相關分群，以便處理更進一步的關聯語意關係，由於 T 值是採取高度相關判別，因此低度相關的關聯語意關係將不會進入後續層次的分群，所以實際符合關聯高度語意關係的分群是少數。

在未來裡，我們將完成下列幾項工作：(1) 收集英文文件，以便提供英文線上測試平台，我們預計收集至少 10,000,000 英文文件；(2) OTFDC 可以運用至相當多的領域。比方醫學，當我們輸入疾病名稱，系統將會產生諸如併發症、對抗藥物之類的回答。比方生物學，當我們輸入生物名稱，系統將會產生諸如生物天敵、生物特性之類的回答。

註：本論文實作之文件分群演算法網址：
http://cayley.sytes.net/chinese

誌謝：本論文經費由國科會專題計畫所補助，計畫編號 NSC 96-2218-E-259-001。

參考文獻

[1] J. N. Alertbox, *One Billion Internet Users*, 2005.

[2] P. Berkhin, *Survey of clustering data mining techniques*, 2002.

[3] J. C. Bezdek, *Pattern Recognition with Fuzzy Objective Function Algorithms*, Kluwer Academic, 1981.

[4] Y. Chang, M. Kim and V. V. Raghavan, *Construction of query concepts based on feature clustering of documents*, Information Retrieval, 9 (2006), pp. 231-248.

[5] M. Chavent, Y. Lechevallier and O. Briant, *DIVCLUS-T: A monothetic divisive hierarchical clustering method*, Computational Statistics and Data Analysis, 52 (2007), pp. 687-701.

[6] L. C. Chen and C. J. Luh, *Web page prediction from metasearch results*, Internet Research: Electronic Networking Applications and Policy, 15 (2005), pp. 421-446.

[7] K. J. Cios, W. Pedrycz, R. W. Swiniarski and L. A. Kurgan, *Data Mining: Knowledge Discovery Methods*, Springer, 2007.

[8] T. H. Cormen, *Introduction to Algorithms*, MIT Press, 2001.

[9] Google, *Google Press Center: Zeitgeist Archive*, 2006.

[10] S. Guha, R. Rastogi and K. Shim, *CURE: an efficient clustering algorithm for large databases*, *Proceedings of the 1998 ACM SIGMOD international conference on Management of data*, 1998, pp. 73-84.

[11] S. Guha, R. Rastogi and K. Shim, *Rock: A Robust Clustering Algorithm for Categorical Attributes*, Information Systems, 25 (2000), pp. 345-366.

[12] M. Heck, *Wanted: Methods for moving mountains of content*, 2006.

[13] K. A. Heller and Z. Ghahramani, *Bayesian hierarchical clustering*, *Proceedings of the 22nd international conference on Machine learning*, 2005, pp. 297-304.

[14] InfoSpace, *Unfiltered Metaspy - MetaCrawler*, 2007.

[15] InfoWorld, *2007 Technology of the Year Awards: Data Management*, 2008.

[16] A. K. Jain and R. C. Dubes, *Algorithms for Clustering Data*, Prentice Hall, 1998.

[17] A. K. Jain, M. N. Murty and P. J. Flynn, *Data clustering: a review*, ACM Computing Surveys, 31 (1999), pp. 264-323.

[18] Y. Kanellopoulos, P. Antonellis, C. Tjortjis and C. Makris, *k-Attractors: A Clustering Algorithm for Software Measurement Data Analysis*, *Proceedings of the 19th IEEE International Conference on Tools with Artificial Intelligence*, 2007, pp. 358-365.

[19] L. Kaufman and P. J. Rousseeuw, *Finding Groups in Data: An Introduction to Cluster Analysis*, John Wiley and Sons, 2005.

[20] D. E. Knuth, *The Art of Computer Programming, Volume 3: Sorting and Searching*, Addison Wesley, 1997.

[21] M. d. Kunder, *The size of the World Wide Web*, 2008.

[22] T. Li, S. Ma and M. Ogihara, *Document clustering via adaptive subspace iteration*, *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, 2004, pp. 218-225.

[23] Y. Li and S. M. Chung, *Text document clustering based on frequent word sequences*, *Proceedings of the 14th ACM international conference on Information and knowledge management*, 2005, pp. 293-294.

[24] Y. Li, S. M. Chung and J. D. Holt, *Text document clustering based on frequent word meaning sequences*, Data and Knowledge Engineering, 64 (2008), pp. 381-404.

[25] J. B. MacQueen, *Some Methods for classification and Analysis of Multivariate Observations*, *Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability*, 1967, pp. 281-297.

[26] M. Maggini, L. Rigutini and M. Turchi, *Pseudo-Supervised Clustering for Text Documents*, *Proceedings of the 2004 IEEE/WIC/ACM International Conference on Web Intelligence*, 2004, pp. 363-369.

[27] MiniwattsMarketingGroup, *Top 20 countries with the highest number of Internet users*, 2007.

[28] S. Mitra and T. Acharya, *Data Mining: Multimedia, Soft Computing, and Bioinformatics*, John Wiley and Sons, 2003.

[29] S.-H. Na, I.-S. Kang and J.-H. Lee, *Adaptive document clustering based on query-based similarity*, Information Processing and Management: an International Journal, 43 (2007), pp. 887-901.

[30] M. Nakada and Y. Osana, *Document clustering based on similarity of subjects using integrated subject graph*, *Proceedings of the 24th*

IASTED international conference on Artificial intelligence and applications, 2006, pp. 410-415.

[31] Z.-Y. Niu, D.-H. Ji and C. L. Tan, *Using cluster validation criterion to identify optimal feature subset and cluster number for document clustering*, Information Processing and Management: an International Journal, 43 (2007), pp. 730-739.

[32] P. Pantel and D. Lin, *Document clustering with committees*, *Proceeding of the 25th ACM International Conference on Research and Development in Information Retrieval*, 2002, pp. 199-206.

[33] A. K. Pujari, *Data mining techniques*, Hyderabad, 2001.

[34] R. Saraçoğlu, K. Tütüncü and N. Allahverdi, *A fuzzy clustering approach for finding similar documents using a novel similarity measure*, Expert Systems with Applications: An International Journal, 33 (2007), pp. 600-605.

[35] P. S. Szczepaniak, J. Segovia, J. Kacprzyk and L. A. Zadeh, *Intelligent exploration of the web*, Physica-Verlag GmbH 2003.

[36] W.-C. Tjhi and L. Chen, *A partitioning based algorithm to fuzzy co-cluster documents and words*, Pattern Recognition Letters, 27 (2006), pp. 151-159.

[37] W.-C. Tjhi and L. Chen, *Possibilistic fuzzy co-clustering of large document collections*, Pattern Recognition, 40 (2007), pp. 3452-3466.

[38] E. Voorhees and D. Tice, *The TREC-8 Question Answering Track Evaluation*, *Proceeding of the 8th Text Retrieval Conference*, 1999, pp. 77-82.

[39] E. M. Voorhees, *Implementing Agglomerative Hierarchical Clustering Algorithms for Use in Document Retrieval*, Information Processing and Management, 22 (1986), pp. 465-476.

[40] R. E. Walpole, R. H. Myers, S. L. Myers and K. Ye, *Probability & Statistics for Engineers & Scientists*, Prentice Hall, 2006.

[41] X. Wan and J. Yang, *CollabSum: exploiting multiple document clustering for collaborative single document summarizations*, *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, 2007, pp. 143-150.

[42] F. Wang, C. Zhang and T. Li, *Regularized clustering for documents*, *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, 2007, pp. 95-102.

[43] Wikipedia, *Likert scale - Wikipedia, the free encyclopedia*, 2008.

[44] L. Yang, D. Ji, G. Zhou, Y. Nie and G. Xi, *Document re-ranking using cluster validation and label propagation*, *Proceedings of the 15th ACM international conference on Information and knowledge*

management, 2006, pp. 690-697.

[45] M. L. Zeng, *Construction of Controlled Vocabularies: A Primer* 2005.

[46] T. Zhang, R. Ramakrishnan and M. Livny, *BIRCH: An Efficient Data Clustering Method for Very Large Databases*, *Proceedings of the 1996 ACM SIGMOD International Conference Management of Data*, 1996, pp. 103-114.