南　華　大　學

資訊管理學系

碩士論文

以相似度模式爲基礎的無線廣播資料配置

Data Placement for Wireless Broadcast Under New Affinity Model

研 究 生：蔡閔皓

指導教授：蔡德謙　博士

中華民國 九十四 年 六 月

以相似度模式為基礎的無線廣播資料配置
Data Placement for Wireless Broadcast Under New Affinity Model

研　究　生：蔡閔皓　Student : Ming-Hao Tsai

指導教授：蔡德謙　Advisor :Dr.Derchian Tsaih

南　華　大　學

資訊管理學系

碩士論文

A Thesis

Submitted to Department of Information Management

College of Management

Nan-Hua University

in partial Fulfillment of the Requirements

for the Degree of

Master of Business Administrator

in

Information Management

June 2005

Chaiyi Taiwan, Republic of China.

中華民國 九十四 年 六月

南　華　大　學

資訊管理學系

碩 士 學 位 論 文

以相似度模式爲基礎的無線廣播資料配置

研究生：蔡閔皓

經考試合格特此證明

口試委員：吳光閔

何濂到

指導教授：蔡德謙

系主任(所長)：資訊管理學系 系主任 吳光閔

口試日期：中華民國　94　年　6　月　30　日

誌　　謝

回想當初剛進南華的日子，曾經因壓力太大想一度放棄學業，幸而在父母的支持鼓勵下，讓我無後顧之憂繼續完成學業。也很感謝 蔡德謙老師平日來的指導，在蔡老師的教導下讓我學會了用不同的角度去看事情，真的受益良多。很高興當初的選擇是對的！

也謝謝 吳光閔老師及 何漢彰老師對我的論文指正及建議，讓我的論文能趨於完整，在這邊表達我對兩位老師的謝意。

接下來當然也免不了在 325 研究室裡的好夥伴們，謝謝俊杰、乾訓、明哲、建榮、建磐、如卿、美倫、元安、淑玲、小童、嘉明的幫忙與陪伴，有你們這兩年真的很豐富，這段日子的點點滴滴都將深深地烙印在我心中。也差點漏了一個人，就是學妹曉吟，很感謝她都沒一點抱怨幫忙我修改及校正論文，讓我還能有稍稍喘息的機會，真的很謝謝！

最後，爸、媽、蔡老師、325 研究夥伴們，我想對你們說：「謝謝你們！我永遠愛你們的！」

蔡閔皓　僅識

于南華大學

民國九十四年六月

以相似度模式為基礎的無線廣播資料配置

研究生：蔡閔皓　　指導教授：蔡德謙　博士

南　華　大　學　資訊管理學系碩士班

摘　　要

為了因應大量的客戶端能快速地接收完資料，資料的配置在無線廣播裡扮演一個重要的角色。對多點查詢(Multipoint query)的系統來說，若多個資料項是屬於同一個查詢(Query)內的資料項，則集中廣播此資料將可降低查詢的平均存取時間(Access time)。數種已被提出的技術則是以資料的相近度來叢集(Clustering)廣播的所有資料項。資料的相似度最主要的目的就是要降低查詢距離(Query distance)。然而，為了能有效降低平均存取時間，依據相似度我們提出一個最小空隙法(Minimum Gap)來合併相關的區斷(Segment)。透過實驗結果顯示使用我們的相似度衡量標準來配置廣播資料項確能降低查詢的平均讀取時間。

關鍵字：資料相似度，多點查詢，資料配置

Data Placement for Wireless Broadcast Under New Affinity Model

Student：Ming-Hao Tsai　　Advisors：Dr. Derchian Tsaih

Department of Information Management
The M.B.A. Program
Nan-Hua University

ABSTRACT

In order to adapt to a large number of client can receive data quickly, data placement in wireless broadcast environment act an importance role. For System with multipoint queries, data records which queried by same query are broadcasted contiguously to reduce the average access time. Several techniques have been used in clustering the data by defining the affinity between the data records. The data affinity function defined was mainly aiming at minimizing Query Distance. However, in order to minimize the average access time, we propose a Minimum Gap which merge relevant segments base on affinity function. Through experiments, the result show not only the query's access time can be reduced by using our affinity function.

Keywords: *Data Affinity, Multipoint Query, Data Placement*

目　錄

表　　目　　錄

圖　　目　　錄

第一章　緒論

1.1 研究環境與背景

隨著科技發展及時代進步，無線通訊(wireless communication)的使用相當普遍，已成為人類生活重心的一部份。資料不論在何處都能透過無線裝置來收取，讓使用者無時無刻都能感受其方便。在無線環境裡，由於頻寬(Bandwidth)受限所衍生出大部份的研究議題都著重如何更有效率的去發佈及傳遞資料，找出那些熱門的資訊是較常被使用及如何分配所需的資料，使用有限的頻寬並在最小的時間內服務需要資訊的使用者。

有鑑於無線環境中頻寬的不足[1][2][22][23][30][32][40]，資料通常會以廣播[4][7][8][16][17][26][48]的方式來儘可能滿足客戶端所需要的資料，並使客戶端能在最短時間內將資料完整讀取完成。圖 1-1 為無線廣播環境示意圖。

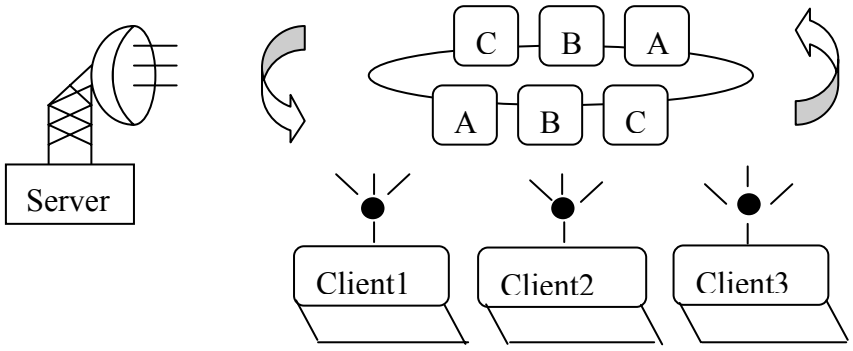


圖 1-1 無線廣播環境示意圖

伺服器會將資料傳送至廣播頻道(Broadcast Channel)中，客戶端可以監聽這個頻道並且取得所需的資料，若其他客戶端亦需讀取相同的資料，也只要進入到相同頻道中監聽即可取得所需資料。

在無線廣播環境下又分為兩種基本模式，一種為「Pull-based」[3][11][14][33][39][45]，如圖 1-2。當客戶端提出了查詢請求給伺服器後，伺服器會將查詢請求內的資料項透過廣播頻道發送出去。在同一時間內，有其他的客戶端提出相同或不相同的查詢請求，那伺服器就可依客戶端查詢要求時間的先後順序，或是客戶端等待時間的長短，來決定排程中的廣播順序以降低客戶端查詢的平均等待時間(Average Waiting Time)。此模式考慮到不同客戶端查詢請求的公平性，使得須讀取冷門資料的客戶端，不會因等待過久而產生挨餓(Starvation)的狀態。

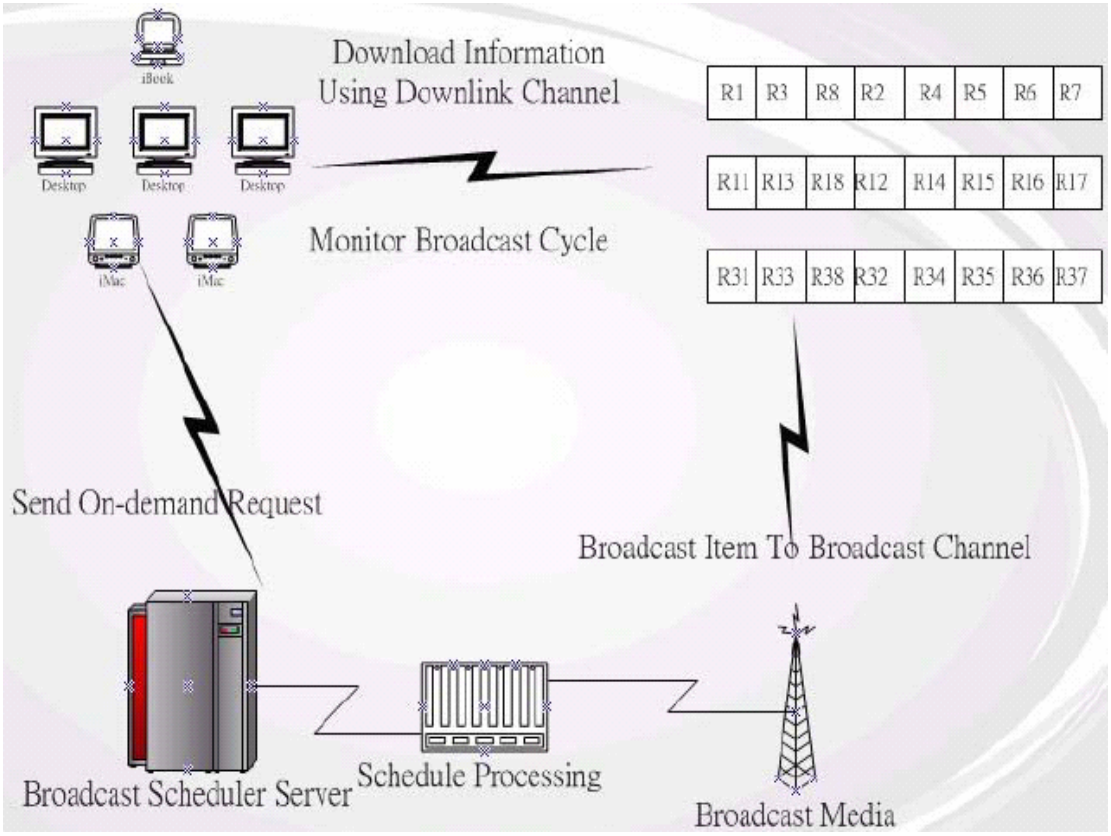


圖 1-2　Pull-based 系統

在 pull-based 系統中，有幾個較常被採用的演算法：

- First Come First Served(FCFS)[3][39][45]，是依照公平的方式，先廣播最早到達的客戶端查詢請求所需資料項，其缺點則是使查詢的平均等待時間變長。
- Most Request First(MRF)[3][39][45]，伺服器端會優先廣播最多客戶端查詢請求最多的資料項，這表示熱門的資料會優先被廣播，而冷門的資料相對地就比較少有機會被廣播，因而產生挨餓(Starvation)現象。

- Longest Wait First(LWF)[3][39][45]，伺服器以客戶端查詢請求的到達時間的順序來廣播所需的資料項，但是此種方法需記錄所有客戶端的資料請求，將會加大伺服器的負擔。
- Request times Wait(RxW)[3][39][45]，1998 年由 D. Aksoy and M.Franklin 兩人提出改良的演算法 RxW[3][45]，此種方法結合了「MRF」與「LWF」兩種演算法判斷的機制，依客戶端查詢請求的等待時間與數量決定所須要廣播的資訊項，其優點是能降低「平均等待時間」。這個方法同時考慮了使用者對資料的要求時間及該資料被要求的熱門程度，從中找到一個最好的平衡點來播出客戶端所需資料。

而在多重的客戶端的環境下，還有另一種廣播的型式，我們稱為「Push-based」[3][14][39]，如圖 1-3。這個方法就是將使用者所要求的資料先全部收集，找出較常被要求的熱門資料項，將所有的資料項組成一個資料廣播序列。在固定的時間內，將此序列在廣播頻道上持續廣播，讓大多數客戶端所請求的資料項皆能直接從廣播頻道中讀取。

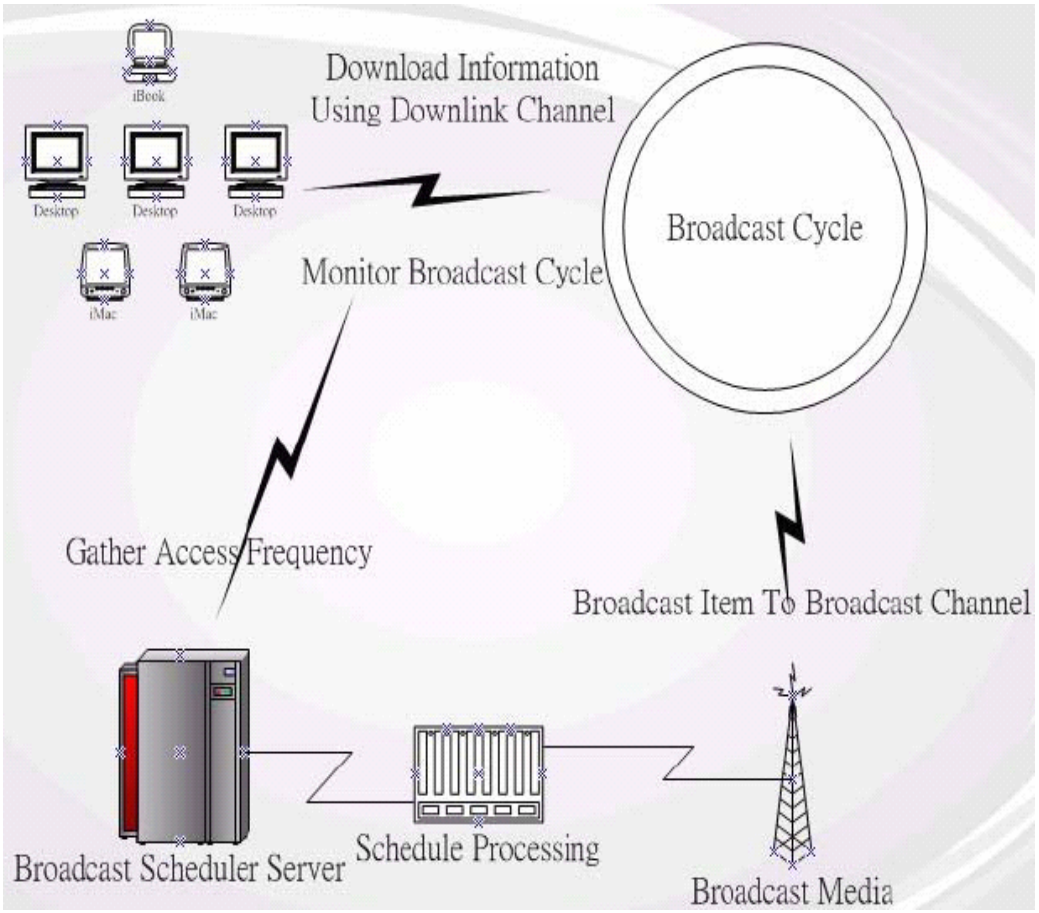


圖 1-3　Push-based 系統

在 Push-based 的環境下，又有許多需被探討的議題，在現實情況中，客戶端所要求的資料項可能大小不相等，若廣播長度較大的資料項所需的時間將比廣播長度較小的資料項為久，且優先廣播長度較小的資料項將能降低資料項的平均等待時間(Average Waiting Time)。所以資料項的大小在現實環境中是應被列入決定廣播順序的考量，但為了讓問題容易化，在本論文皆把資料項的大小假設成相同大小，也就是在廣播頻道中，每一個單位時間都可以剛好播出一個資料項。在此環境下，又可將研究的方向細分為兩種模式「平

坦式」(Flat)與「非平坦式」(Non-flat)，如下圖 1-4。經過排程演算法後，需被客戶端讀取的資料項被伺服器依序廣播，在一廣播週期中所有資料項將只被廣播一次，此種方法被稱為「平坦式」模式[4][18][19]；反之，將資料依照出現頻率多寡或資料與資料間相互關係的影響，在一廣播週期中熱門的資料項複製多次後再排序，稱之為「非平坦式」模式[1][2][15]。

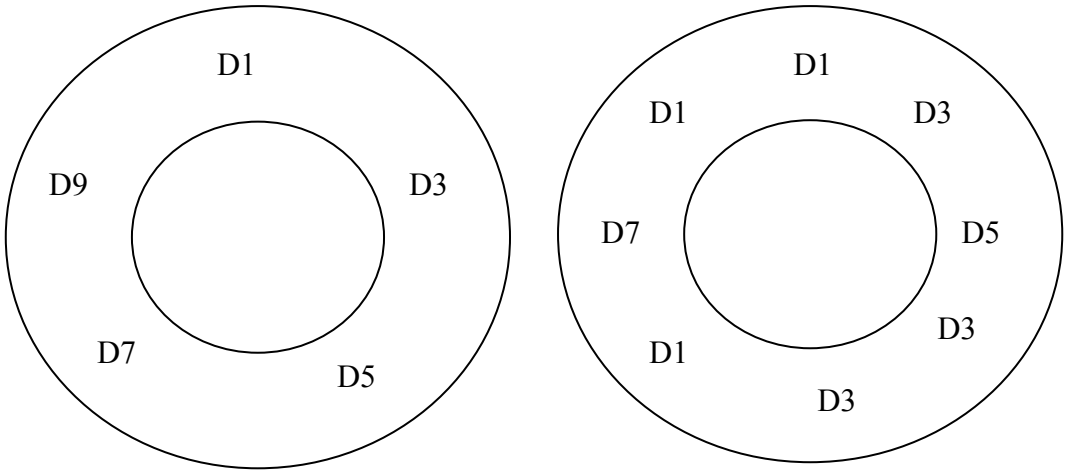


圖 1-4　Flat 與 Non-flat 示意圖

若又依每個客戶端所要求的資料項數目，又可分成每個客戶端只要求一個資料項的環境下，稱為單一資料要求(single data items query)[1][2]，反之若在客戶端可要求兩個以上的資料項的環境下，我們則稱為多重資料要求(multiple data items query)[7][8][27][28]。由上述就可以分為四種情形，分別是「單一資料要求」以「平坦式」

的形式廣播，「單一資料要求」以「非平坦式」的形式廣播，「多資料要求」以「平坦式」的形式廣播，及「多資料要求」以「非平坦式」的形式廣播。

較早期的研究大多探討在「單一資料要求」及「平坦式」模式下的廣播排程，其實驗環境最單純且所需考慮狀況最少，而後有「單一資料要求」、「非平坦式」廣播模式之研究。在現實環境中，行動客戶端提出一筆查詢(Query)要求時，要求多筆資料是較為合理的，所以「多資料要求」、「平坦式」廣播模式及「多資料要求」、「非平坦式」廣播模式是目前無線廣播重要的研究議題。在本篇論文研究即是「多資料要求」、「平坦式」廣播模式，運用排程演算法將廣播資料項做最適當的排列以減少行動客戶端查詢的平均等待時間。

1.2 研究目的

在無線資料廣播環境中，有兩個相關的議題。一個重要的議題為「減少平均的等待時間」[13][24][28][32][49][50][51]，降低平均等待時間才能建立「即時反應」(real-time)[11][29][30]的環境。平均的等待時間降低，直覺上是將熱門的資料作多次廣播，讓大部分的客戶端很快收到所需資料項，但是一些冷門的資料項就可能因為不常被使用，會造成無法滿足部分客戶端的查詢請求，因此也有部分

研究考慮公平性，只要客戶端有要求且伺服器有此資料項，一定能在限定時間內廣播出，常見減少平均等待時間的方法有快取(cache)[6][20][48][52]和非均勻(nonuniform)廣播[1][2]、多通道(multiple channel)[4][12][13][18][36][37][41][44][50]等等方法。

另一項需考慮的議題是行動裝置的能源問題[5][10][15][17][20][21][24][31][46]，因為行動裝置的能源無法保持連續供應以及不容易隨時取得，所以節省能源也是研究無線廣播環境中重要的一環。「索引技術」[5][9][13][15][17][20][22][23][38][47]常被使用於節省能源上；當伺服器端接收到一個客戶端查詢請求時，先檢查此請求的資料項是否已在廣播的清單中，若已在廣播的清單中則傳回一個索引值到客戶端。此時，客戶端收到的索引值可以讓客戶端知道，還需要等待多少的時間，廣播頻道上才會出現所需要的資料項。這段時間，客戶端可以進入「休眠模式」而不用持續監聽頻道上的資料，因為客戶端只在廣播資料項出現時才會持續接收資料所以加入索引技術能有效的節省能源消耗。

在過去，有部份學者在探討如何讓廣播頻道中的資料能被客戶端快速接收完成，其中[34]提出用資料挖掘(data mining)方式，藉由發掘客戶端的移動行為來建立資料的配置。在[7]中提出用由下而上

的叢集及線性相似度函數來合併區斷(segment)。

在廣播順序裡為了要存取多個資料，有學者提出一種衡量資料相近度的查詢距離(Query Distance)以及基於降低此查詢距離下如何建立廣播順序的查詢擴張法(Query Expansion method)[8]。許多後續研究則探討如何縮短這查詢距離其中包含使用格雷碼(Gray Code Method)[25]，及基因演算法在多個頻道上配置廣播資料項[18]。基本上用查詢距離的方法來評估平均存取時間是良好的方法，但它會忽略查詢所讀取資料間的二次式距離影響。

排程問題主要的目標在降低查詢的平均讀取時間，而平均存取時間是和查詢所讀取資料間距離的平方和成反比，藉由這個特性，我們可建立一個新的衡量標準稱為二次式查詢距離(Quadratic Query Distance，簡稱 QQD)來表示在一個查詢裡被共同讀取的資料接近度。使用這個二次式的目標函數能比之前所使用的線性目標函數更能精確地衡量資料項之間的相似度。

1.3 本文架構

本篇第二章為文獻探討將敘述先前已提出的廣播排程演算法。第三章是描述問題的結構和介紹平均存取時間的背景。在第四章，我們介紹資料相似度(data affinity)和區斷相似度(segment affinity)以

及提出一個最小空隙法並再針對最小空隙法的缺點再提出改良後的最小空隙法。第五章為實驗結果與探討，說明我們這個方法的實驗預設環境，相關可以調整的變數及實驗結果，並對實驗結果加以說明討論。而第六章則為結論。

第二章 文獻探討

2.1 查詢擴張法(QEM)

查詢擴張法是在[8]中提出，這個方法的排程是以查詢為主要的考慮方向，在這個環境之下，每個使用者所要求的一組資料形成一個集合(query data set)，可能有部份使用者會要求到同樣的資料集合，所以每個查詢資料項集合(query data set)有其本身的被提出機率(probability)，而且不同的使用者要求的集合之間可有重覆的資料項。QEM 針對每個查詢資料項集合，以本身的機率大小為依據來排程。因為機率越高的，表示越多使用者提出對此資料項請求的查詢。若能以較高機率的查詢資料項集合來優先考慮，可以讓較多要求此資料集合的使用者，可以用較短的時間來存取，以降低查詢的平均讀取時間。

查詢擴張法(QEM)是以查詢資料項集合的機率高低做為排入廣播頻道的考慮優先順序，每排入一個查詢，就會先檢查這個查詢的資料項是否已在廣播排程中，對已列在廣播排程中的資料項不再重覆放入而只放入未列在廣播排程中的資料項。因為所有資料項均以 Flat 的型式播出，所以每個資料只會出現一次。查詢擴張法有下面三個重點：

1. 貪婪法，由查詢要求率的高低順序排入到廣播排程中。

2. 新加入排程的查詢，不會更動排程中原有查詢的查詢距離(Query Distance)。

3. 新加入排程的查詢，儘量排列到使其查詢距離越小越好。

在查詢擴張法中還有提到另一個重要的觀念，就是查詢距離(Query Distance)，在廣播頻道中，令廣播頻道的長度(資料個數)為B，查詢距離則為 B 減去此查詢在廣播頻道的最大間隔，舉例來說，若廣播頻道為[d_1,d_2,d_3,d_4,d_5,d_6,d_7]，則 B=7。令此查詢所要讀取的資料項為[d_3,d_4,d_5]，則此查詢的查詢距離為 7-4(此查詢在頻道中的最大間隔為 4：d_6 d_7 d_1 d_2)=3。若將廣播頻道變更為[d_1,d_3,d_5,d_7,d_2,d_4,d_6]，則此查詢的查詢距離就是 7-2(此查詢在頻道中的最大間隔為 2：d_7 d_2 或 d_6 d_1)=5。在上面的例子中，此查詢距離在第一個廣播排程中為 3，而在第二個廣播排程中卻需要 5。

2.2 格雷碼叢集演算法

格雷碼[25]是一種數碼系統，不同於普通電腦內的數位系統以二進位 (binary code)為基礎，Gray Code 對應的二進碼，只會有一個位元的資料不同。

為當數字增加 1 時，Gray Code 只會有一個位元有變化(0 變 1、

或者是 1 變 0)。格雷碼與十進位制之間的轉換必須透過二進位的幫助。如 45，其二進位為 101101。以二進位來表示格雷碼，數個碼將會被編成一個二進位字串，編碼後的連續字串中每個位元的位置不同。

表 2-1 是表示 3 位元的二進制碼轉換為格雷碼。二進位字串的格雷碼值為二進位字串的順序。舉個例子來說，“110”的格雷碼值為 4，也等於“100”的二進制碼。

表 2-1　3 位元的二進制碼轉換為格雷碼

十進制碼	格雷碼	二進制碼
0	000	000
1	001	001
2	011	010
3	010	011
4	110	100
5	111	101
6	101	110
7	100	111

2.2.1 特徵表示

在這個方法裡，我們使用多屬性雜湊函數把資料記錄和查詢表示為資料特徵(data signatures)和查詢特徵(query signatures)。以下例

來解釋資料記錄和查詢如何在多屬性雜湊函數中被表示成特徵。

假設有一個無線資訊系統用來即時更新股票價格的資訊。每個資料記錄有四個屬性：A1(公司)，A2(銷售數量)，A3(購買數量)和A4(價格)。對每一個屬性，我們假設有下列簡單的雜湊函數。

$$h_{公司}(x) = \begin{cases} 00 & \text{假設 x 是財金公司} \\ 01 & \text{假設 x 是製造公司} \\ 10 & \text{假設 x 是電腦通訊公司} \\ 11 & \text{全不是} \end{cases}$$

$$h_{銷售數量}(x) = \begin{cases} 00 & \text{假設 } x < 10000 \\ 01 & \text{假設 } 10000 \le x < 20000 \\ 10 & \text{假設 } 20000 \le x < 30000 \\ 11 & \text{假設 } 30000 \le x \end{cases}$$

$$h_{購買數量}(x) = \begin{cases} 00 & \text{假設 } x < 10000 \\ 01 & \text{假設 } 10000 \le x < 20000 \\ 10 & \text{假設 } 20000 \le x < 30000 \\ 11 & \text{假設 } 30000 \le x \end{cases}$$

$$\text{h 價格(x)} = \begin{cases} 000 & \text{假設 } x < 5 \\ 001 & \text{假設 } 5 \le x < 10 \\ 010 & \text{假設 } 10 \le x < 20 \\ 011 & \text{假設 } 20 \le x < 30 \\ 100 & \text{假設 } 30 \le x < 40 \\ 101 & \text{假設 } 40 \le x < 50 \\ 110 & \text{假設 } 50 \le x < 60 \\ 111 & \text{假設 } 60 \le x \end{cases}$$

一個資料特徵的產生是藉由連結每個屬性 i 的雜湊 l_i 位元向量。因此特徵會變成 l-位元的二進制位元向量，$l = \sum_{i=1}^{k} l_i$(k 是屬性的數目)。我們假設屬性被配置成<A_1,A_2,A_3,A_4>這樣的順序，所以 l_1,l_2,l_3=2 和 l_4=3。

基於以上的雜湊方案，我們可以表示一筆資料記錄{公司='紐約銀行', 銷售數量='13000', 購買數量='2000', 價格='22'}為"010100011"。同樣地，我們也能表示一個局部符合的查詢{銷售數量≥'30000','10' ≤價格<'20'}為 "**11**010" ，其中 "*" 是表示不考慮的情況。

2.2.2 叢集法

這個方法主要是由兩個步驟所組成：第一，基於特定的雜湊函數把資料表示成資料特徵。第二，基於他們的格雷碼值來排序資料特徵。

當有七筆資料記錄 R_1～R_7(如下表 2-2 所示)，而屬性值和雜湊函數也在上節中提到，叢集法的演進如下：

表 2-2　R_1～R_7 資料記錄表

id	公司	銷售數量	購買數量	價格	記錄特徵	格雷碼	二進制碼
R_1	紐約銀行	5000	35000	69	000011111	21	31
R_2	LA 廣播公司	15000	25000	49	110110101	294	437
R_3	San Jose 電腦	25000	15000	39	101001100	392	332
R_4	Dallas 機械	35000	5000	29	011100011	190	227
R_5	Texas 電子	500	45000	19	010011010	236	154
R_6	Miami 工業	8000	31000	4	010011000	239	152
R_7	Hawaii 通訊	24000	19000	32	101001100	392	332

在叢集法的第一步驟裡，我們把資料記錄表示成資料特徵，就是在上表中的「記錄特徵」。在第二步驟裡，我們依照他們的格雷碼做排序動作，所以我們可以得到下列的排程。

$\sigma_{Gray} = <R_1,R_4,R_5,R_6,R_2,R_3,R_7>$,

$\sigma_{binary} = <R_1,R_6,R_5,R_4,R_3,R_7,R_2>$

σ_{Gray} 是用格雷碼方式所得到的排程而 σ_{binary} 是基於二進制碼對資料特徵做排序後所配置的排程。

在排序步驟中，若資料記錄有相同的格雷碼或二進制碼時，可以互相交換其順序。(以上表為例，R_3 和 R_7 有相同的格雷碼)。不管資料是依升冪或降冪排列都不會影響其順序。

2.3 Chung's 演算法[7]

資料廣播有一種特性，就是多個客戶端是獨立的，客戶端不需發出請求到伺服器，但必需接收在廣播上的串流。因此，在無線資訊系統裡，資料廣播已被廣泛使用，因為它是一個眾所皆知的有效方法用來管理大量的行動客戶端。然而，假如用叢集法依照行動客戶端的數量來廣播資料，則會產生嚴重的問題。(在叢集法裡，我們把客戶端數量當做是查詢的數量)。

實際上，查詢的數量在廣播資料串流中是非常龐大的。當資料記錄的數目為 N 且一個多點查詢可讀取 k 個資料，全部可能的多點查詢數量為

$$\sum_{i=1}^{k} C_i^N \tag{2-1}$$

之前的叢集法，像 QEM 和 GCM，當有大量的查詢且他們的頻

率是一樣的就無法顯示出很好的效能。這是因為之前的叢集法會基於一個查詢所存取幾組資料記錄來做出廣播排程，因此，在考慮一些頻率較高的查詢而並非全部的查詢後，廣播排程會有太早決定的現象。

為了解決這樣的問題，我們考慮資料間的關係來建立一個新的叢集法。對於這個新的叢集法，我們定義兩個相似度測量標準，分別是資料相似度和區斷相似度。在以下的定義中，我們將會使用到下列的符號。

表 2-3　符號定義表

d_i	在廣播串流上的一筆資料記錄
$\mid d_i \mid$	d_i 的大小
D	在廣播串流上資料記錄的集合
N	在 D 裡資料記錄的數量
q_i	在廣播串流上行動客戶端所發出的查詢
QDS(q_i)	q_i 存取資料記錄的集合
$N_{data}(q_i)$	查詢 q_i 存取幾筆資料記錄
freq(qi)	查詢 q_i 的頻率

Q	全部查詢集合
M	在 Q 裡查詢的數量
σ	一個廣播排程

2.3.1 資料相似度

兩筆資料記錄的資料相似度表示在同一個查詢裡他們被一起讀取的程度，以下列定義來表示之：

定義一：任何介於兩筆資料 d_i、d_j 的資料相似度為：

$$\text{aff}(d_i,d_j)=\begin{cases}\sum_{\forall qk\in Q}\dfrac{\text{has}(q_k,d_i,d_j)\times\text{freq}(q_k)}{C_2^{N_{data}(q_k)}} & \text{if} \quad i\neq j\\ 0 & \text{otherwise}\end{cases} \tag{2-2}$$

$$\text{has}(q_k,d_i,d_j)=\begin{cases}1 & if \quad d_i\in q_k\wedge d_j\in q_k\\ 0 & \text{otherwise}\end{cases} \tag{2-3}$$

資料相似度有下列的特性：

1. 假如沒有任何查詢同時讀取資料項 d_i、d_j，則 $\text{aff}(d_i,d_j)=0$.

2. 任兩筆記錄的相似度是 *mutual*，例 $\text{aff}(d_i,d_j)=\text{aff}(d_j,d_i)$.

3. 所有成對資料的相似度總和會等於讀取查詢的頻率總和。

$$\sum_{i=1}^{N-1}\sum_{j=i+1}^{N}\text{aff}(d_i,d_j)=\sum_{q_k\in Q\wedge N_{data}(q_k)>1}\text{freq}(q_k) \tag{2-4}$$

一個成對資料項若被愈多查詢共同讀取則其相似度會愈大。在定義中，我們用總和除於所有可能成對的資料項的數目，可以得到

一個查詢所讀取的資料項。這是因為有多筆的資料項被同一個查詢所讀取，每個成對的資料項對於叢集後的效能影響不大。在我們方法中，我們會優先選取較高的相似度來叢集成對的資料。

定義二：讓 N 表示資料記錄的數量。相似度矩陣(affinity matrix)是一個 N x N 的矩陣，定義如下：

$$\mathrm{AM}[i][j] = \mathrm{aff}(d_i,d_j)$$

2.3.2 區斷相似度

區斷是一經排序的資料項序列且被表示為 $S_i = (d_1,d_2,\ldots,d_k)$。叢集演算法則是藉由不斷地合併區斷來產生一組廣播順序。起初，每個區斷只含一筆資料記錄。而在每一次合併區斷將均會使用下列的規則：

1. 區斷 S_i 會合併到與自己的資料項有較大相似度的區斷。
2. 當合併兩個區斷時，我們會把有較高相似度的區斷儘可能的配置在一起。

定義三：S_i 及 S_j 兩區斷的區斷相似度為：

$$\mathrm{SegAff}(S_i,S_j) = \begin{cases} \sum_{dk\in Si}\sum_{dl\in Sj}\mathrm{aff}(d_k,d_l) \\ \quad \times \dfrac{\mathrm{MaxDist}(d_k,d_l) - \mathrm{dist}(d_k,d_l)}{\mathrm{MaxDist}(d_k,d_l)} & \text{if } i \neq j \\ 0 & \text{otherwise} \end{cases} \qquad (2\text{-}5)$$

對於處理區斷合併，我們定義上式的區斷相似度。為兩個區斷裡資料項的相似度總和，其中每個資料項相似度都被乘於一個距離因子。dist(d_i,d_j)是在現在的廣播順序下介於 d_i 和 d_j 間的最小距離，而 MaxDist(d_i,d_j)是 d_i 和 d_j 間的最大可能距離。

定義四：讓 δ 表示為介於 d_k 和 d_l 間的間隔數。所以，dist(d_k,d_l)及 MaxDist(d_k,d_l)分別定義為：

$$\text{dist}(d_k, d_l) = \begin{cases} |d_k| + \delta + |d_l| & \text{if } \delta < \dfrac{\text{BSize} - |d_k| - |d_l|}{2} \\ \text{BSize} - \delta & otherwise \end{cases} \tag{2-6}$$

$$\text{MaxDist}(d_k, d_l) = \frac{\text{BSize} + |d_k| + |d_l|}{2} \tag{2-7}$$

若在定義三(MaxDist(d_k,d_l)－dist(d_k,d_l))／MaxDist(d_k,d_l)的值接近 1 則兩筆資料(d_k,d_l)在排程中是距離最小，若接近 0 則他們的位置在廣播排程中則是距離最大。

與資料相似度相比，區斷相似度是不對稱的。所以，當我們合併區斷來產生一組廣播排程時我們必須考慮到的不只是合併的區斷且需決定連結的位置。

定義五：令 N 表示為廣播資料項的總數則區斷相似度矩陣(Segment Affinity Matrix)是一個 N x N 的矩陣，定義如下：

SAM[i][j] = SegAff(S_i,S_j)

2.3.3 反向區斷

反向排程的產生是因為資料的順序被放置在原來排程的反向順序。舉例來說，在下方的兩個廣播排程是可互相對調順序的排程。一個查詢讀取時間的決定是由廣播排程上資料的間隔數所決定且在正向順序上的間隔數會等於反向順序上的資料間隔數。在正向順序中一個查詢的讀取時間也會等於在反向順序中一個查詢的讀取時間。

$\sigma_1 = <d_1,d_2,\ldots,d_{i-1},d_i,d_{i+1},\ldots,d_{n-1},d_n>$

$\sigma_2 = <d_n,d_{n-1},\ldots,d_{i+1},d_i,d_{i-1},\ldots d_2,d_1>$

在處理叢集法裡若是考慮以上的特性，我們將定義反向區斷，區斷裡的資料項將是反向的順序：

定義六：區斷 S_i 表示為 $S_i = (d_1,d_2,\ldots,d_n)$，則反向區斷 S_i^{-1} 表示為：$S_i^{-1} = (d_n,d_{n-1},\ldots,d_1)$

特性二：

SegAff(S_i,S_j)= SegAff(S_j^{-1},S_i^{-1})

即區斷 S_i 與 S_j 的區斷相似度會等於它們反向順序後的反向區斷

的區斷相似度。若考慮所有兩個區斷(S_i 和 S_j)的配置，有下列八種可能。

SegAff(S_i,S_j), SegAff(S_j,S_i)

SegAff(S_i^{-1},S_j), SegAff(S_j^{-1},S_i)

SegAff(S_i,S_j^{-1}), SegAff(S_j,S_i^{-1})

SegAff(S_i^{-1},S_j^{-1}), SegAff(S_j^{-1},S_i^{-1})

根據特性二，他們的區斷相似度則符合

SegAff(S_i,S_j) = SegAff(S_j^{-1},S_i^{-1}),

SegAff(S_i^{-1},S_j) = SegAff(S_j^{-1},S_i)

SegAff(S_i,S_j^{-1}) = SegAff(S_j,S_i^{-1}),

SegAff(S_i^{-1},S_j^{-1}) = SegAff(S_j,S_i)

因為可以從四種方案得知所有組合的區斷相似度[SegAff(S_i,S_j) (SegAff(S_j^{-1},S_i^{-1})), SegAff(S_i^{-1},S_j) (SegAff(S_j^{-1},S_i)), SegAff(S_i,S_j^{-1}) (SegAff(S_j,S_i^{-1})), SegAff(S_i^{-1},S_j^{-1}) (SegAff(S_j,S_i))]。所以當 $i > j$ 時，其中 SegAff(S_i,S_j)及 SegAff(S_j,S_i)之值可由區斷相似度矩陣中找出而 SegAff(S_i^{-1},S_j)及 SegAff(S_j,S_i^{-1})之值則可由反向區斷相似度矩陣中得知。反向區斷相似度矩陣(Inverse Segment Affinity Matrix)定義如下：

定義七：讓 N 表示區斷的數量。ISAM 矩陣是一個 N x N 的矩陣，其組成要素如下：

$$\mathrm{ISAM}[i][j] = \begin{cases} \mathrm{SegAff}(S_i^{-1}, S_j) & \text{if } i > j \\ \mathrm{SegAff}(S_i, S_j^{-1}) & \text{otherwise} \end{cases}$$

2.3.4 演算法

在剛開始合併之前，每個區斷都只有一筆資料項。我們所建立的叢集法是藉由合併區斷來產生一組廣播排程。在合併過程中，我們均選擇有區斷相似度較高的兩組區斷加以合併。

合併過程中我們永遠將有較小的索引號碼區斷置於新形成區斷的左側且將有較大的索引號碼區斷置於新形成區斷的右側。並將較大索引號碼的區斷併入到較小索引號碼的區斷。令 S_l^{-1} 表示 S_l 的反向區斷，而 $S_i \oplus S_j$ 則代表連接 S_i 右側的區斷及 S_j 左側的區斷所新形成的區斷。令 $i<j$，則新形成的區斷將有下列四種可能的情形：

Case I: $S_i \oplus S_j$

Case II: $S_i^{-1} \oplus S_j$

Case III: $S_i \oplus S_j^{-1}$

Case IV: $S_i^{-1} \oplus S_j^{-1}$

在文章裡提出有四個不同的區斷相似度並使用兩個輔助的相似度距陣，分別為區斷相似度矩陣及反向區斷相似度矩陣，如下列：

$$SAM[l][m]=\begin{cases}\text{SegAff}(S_l, S_m) & \text{for } l < m, \text{case I}\\ \text{SegAff}(S_m^{-1}, S_l^{-1}) & \text{for } m \le l, \text{case IV}\end{cases}$$

$$ISAM[l][m]=\begin{cases}\text{SegAff}(S_l, S_m^{-1}) & \text{for } l < m, \text{case III}\\ \text{SegAff}(S_m^{-1}, S_l) & \text{for } m \le l, \text{case II}\end{cases}$$

若 $i<j$，介於 S_i 的右側和 S_j 的左側區斷相似度會儲存在 *SAM*[*i*][*j*]。介於左側的 S_i 和右側區斷 S_j 的相似度則儲存在 *SAM*[*j*][*i*]。以此類推，介於右側的 S_i 和右側的 S_j 的區斷相似度儲存在 *ISAM*[*j*][*i*]。介於左側的 S_i 和左側的 S_j 則會儲存在 *ISAM*[*i*][*j*]。在每一次的合併中，藉由找出在 *SAM* 和 *ISAM* 的最大的相似度且連結它們對應的區斷邊均可形成一個新的區斷。

Algorithm *Broadcast_Data_Clustering*
INPUT: *a set of data records D; a set of queries Q*
OUTPUT: *a broadcast schedule*
METHOD:
1. Make AM from D and Q;
2. Initialize each segment S_i to have a data record d_i;
3. Make SAM and ISAM;
4. DO
5. Find Segment S_i and S_j and their merging order where the segment affinity is maximized;
6. Merge S_i and S_j based on the order;
7. Re-compute SAM and ISAM;
8. ***UNTIL*** *(All element values of SAM are 0);*
9. Merge the remaining segments;

圖 2-1　叢集法的演算過程

第三章　問題描述

我們首先定義在無線環境裡資料配置的問題。資料的配置乃決定於廣播的先後次序。假設所有的資料項皆是相同大小且每個查詢均能存取多筆資料項。我們使用查詢距離(Query Distance)的方法來衡量每個查詢的讀取時間。目標就是找出最佳的廣播順序並使得每個查詢的讀取時間能降到最低。

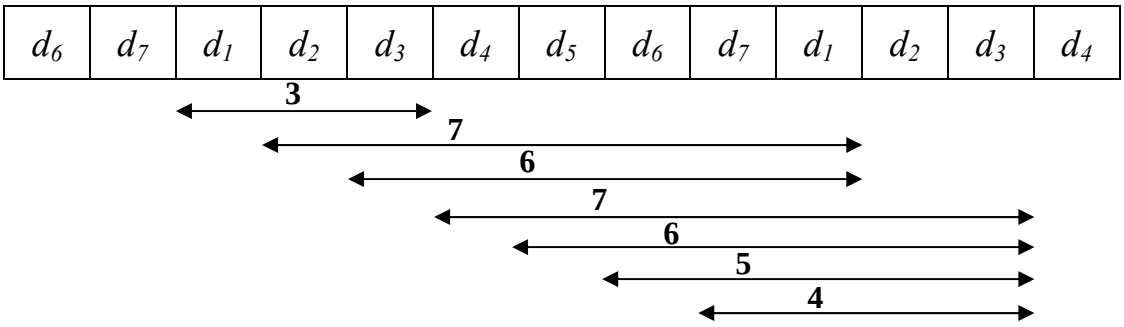
3.1　廣播排程問題

假設有一組待廣播的資料項，包括了 N 個資料項 $D=\{ d_1, d_2, d_3, \cdots, d_N \}$和一組客戶端的查詢包含了 K 個獨立的查詢的集合 $Q=\{ q_1, q_2, q_3, \cdots, q_K\}$且令查詢 q_i 存取的資料項的集合稱做 q_i 的查詢資料集合(Query Data Set) $QDS(q_i)$，其中 $QDS(q_i) \subset D$ 且 $D= \bigcup_{1 \le i \le K} QDS\{q_i\}$令 σ 為無線環境下廣播資料的順序配置，$\sigma=< d_i, d_j , \cdots, d_k>$而 σ_{opt} 是能使查詢的平均讀取時間降到最低的廣播順序。

以圖 3-1 為例，若有一筆查詢 q 欲存取 d_1、d_3 兩個資料而廣播順序為 $\sigma_1 = <d_1, d_2 , d_3 , d_4 , d_5 , d_6 , d_7>$，假設第一筆查詢從 d_1 之前進來至收到 d_3 的時間則為 3 個資料項的時間。若從 d_2 之前進來，則因已錯過這個廣播週期的 d_1 而必須等到下個廣播週期才能接收

到 d_1，因此時間需要 7 個資料項的時間。由圖 3-1(a)得知每個查詢從開始讀取至結束讀取時間分別為<3,7,6,7,6,5,4>，平均的讀取時間則為 38/7。但若將廣播的順序變更為 $\sigma_2 = <d_2, d_1, d_3, d_4, d_5, d_6, d_7>$ 如圖 3-1(b)，每個查詢從開始讀取至結束讀取時間分別為<3,2,7,7,6,5,4>，平均存取時間則降為 34/7。

(a)



(b)

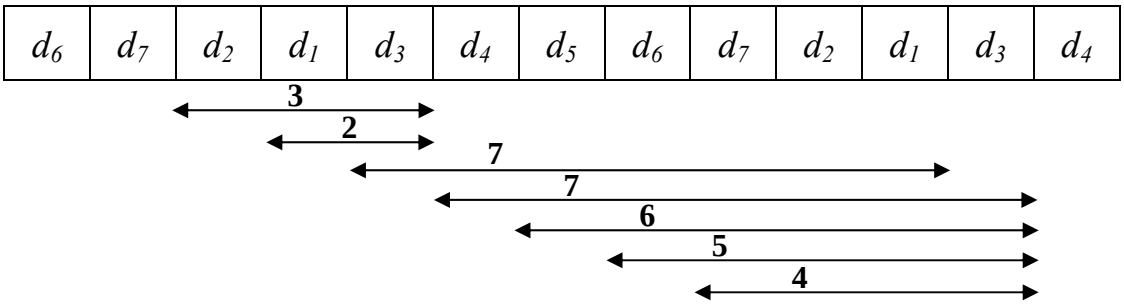


圖 3-1　兩種不同的廣播順序

3.2　平均存取時間

令 N 表示廣播資料項的總數，而 $|q|$ 則為查詢 q 所讀取的資料

項總數，在一個廣播順序 σ 下，查詢 q 的平均存取時間為

$$AT^{avg}(q,\sigma)=N-\sum_{j=1}^{|q|}\frac{\delta_j(\delta_j+1)}{2N} \quad (3\text{-}1)$$

δ_j 是查詢 q 所讀取的第 j 個及第 j+1 個資料項中的資料總數。

在有多個客戶端查詢的系統裡，令查詢 q_k 的平均存取時間表示為 $AT^{avg}(q_k,\sigma)$而它的頻率表示為 $freq(q_k)$ 。在廣播排程 σ 中，查詢的平均存取時間則為：

$$TAT^{avg}(\sigma)=\sum_k freq(q_k)\,AT^{avg}(q_k,\sigma)/\sum_i freq(q_i) \quad (3\text{-}2)$$

以圖 3-1(a)為例，令第一個查詢所需讀取的資料項為$\{d_2,d_4\}$其頻率為 10，第二個查詢所需讀取的資料項為$\{d_2,d_7\}$，其頻率為 9。分別計算兩個廣播順序的查詢距離後，$QD(\sigma_1)$ = (10 * 3 + 9 * 3) / 19 = 57 / 19 而 $QD(\sigma_2)$= (10 * 4 + 9 * 2) / 19 = 58 / 19，依據計算後的結果，σ_1 配置下的平均查詢距離較近。然而，若是考慮平均存取時間，$AT^{avg}(q_1,\sigma_1)= AT^{avg}(q_2,\sigma_1)$= 7 - (1 * 2 + 4 * 5) / 14, $AT^{avg}(q_1,\sigma_2)$ = 7 - (2 * 3 + 3 * 4) / 14 和 $AT^{avg}(q_2,\sigma_2)$ = 7 - (5 * 6) / 14。在廣播順序 σ_1 下的平均存取時間為 $TAT^{avg}(\sigma_1)$=38 / 7 而在 σ_2 配置下的平均存取時間為 $TAT^{avg}(\sigma_2)$ = 10 * (40 / 7) / 19 + 9 * (34 / 7) / 19 = (37.1) / 7，可以看出 σ_2 配置下的平均存取時間較低。

第四章　相似度模式

在這一章中我們探討資料相似度的模式以便在後面章節使用於資料叢集。

4.1　二次式查詢距離

定義一：在廣播資料的配置為 σ 下，查詢 q 的二次式查詢距離表示為：

$$QQD(q,\sigma)=\sum_{j=1}^{|q|}\delta_j^{\,2} \qquad (4\text{-}1)$$

輔助定理一：令 σ_1 和 σ_2 為兩個不同的廣播資料項配置且 q 為任一查詢請求，假如 $QQD(q,\sigma_1)\geqq QQD(q,\sigma_2)$ 則 $AT^{avg}(q,\sigma_1)\leqq AT^{avg}(q,\sigma_2)$

證明：把(3-1)中的 $\sum_{j=1}^{|q|}\delta_j$ 以 N-$|q|$取代後，可清楚看到當二次查詢距離增加時平均的讀取時間也會隨之降低。

定義二：令有一組資料項 D 被一組查詢 $Q=\{q_1, q_2, q_3 \ldots., q_K\}$ 所存取，$\delta_{k,j}$ 是沒有被 q_k 所讀取的第 j 個連續資料項總數，且$|q_k|$是被 q_k 所讀取的資料項總數，則 Q 的二次式查詢距離(quadratic query distance)表示為：

$$TQQD(Q,\sigma)=\sum_{k}\ \mathrm{freq}(q_k)\sum_{j=1}^{|q_k|}\ \delta_{k,j}{}^{2} \qquad (4\text{-}2)$$

輔助定理二：令一個查詢組為 Q 且兩個廣播順序分別為 σ_1 和 σ_2，假如 $TQQD(Q,\sigma_1)\geqq TQQD(Q,\sigma_2)$ 則 $\mathrm{TAT}^{avg}(Q,\sigma_1)\leqq\mathrm{TAT}^{avg}(Q,\sigma_2)$

證明：同輔助定義一。

定義三：有一組資料項 D 和一組查詢 Q，資料排程問題就是找出一個廣播排程 σ_i ，在此 σ_i 下 $TQQD(\sigma_i)$的值為最大，i=1,2,…

提議一：在定義三中，資料排程問題是 (*NP-hard*)。

證明：首先，我們的問題就是將最大化的 $TQQD$ 的問題轉換成使 $\Sigma_k\ \mathrm{freq}(q_k)(N-|q_k|)^2$-$TQQD$ 達到最小化的問題並證明此問題可為一個 *NP-hard* 的二次指派問題(*Quadratic Assignment Problem*)。二次指派問題為發現一個一對一的對應函數 f， $f: V\rightarrow\{1,2,\ldots,|v|\}$，用圖形來表示為 $G=(V,E)$則

$$\min_{f}\sum_{i=1}^{|v|}\sum_{j=1}^{|v|}b_{ij}a_{f(i)f(j)}$$

a_{ij} 是表示位置 i 和位置 j 之間的距離，b_{ij} 是 i 和 j 相對應的邊。讓我們假設每筆查詢只存取兩個資料項。然後將此問題應用在二次指派問題，把資料項當做是節點，b_{ij} 是查詢所存取 d_i 及 d_j 資料項的頻率總和，但 i 不等於 j，而 a_{ij} 是在環型廣播順序中表示介於第 i

筆資料及第 j 筆資料間兩個廣播順序的距離乘積。

當 N 大於等於 20 時，二次指派問題常被認為是難以最佳化問題之一。

4.2 配置成對的資料項

因為 $\Sigma_j \delta_{k,j} = N - |q_k|\ \ \forall k$ 故 $TQQD$ 有兩個理論極端值，分別為其理論上的最大值及理論上的最小值：

$$TQQD_{max}=\Sigma_k \text{freq}(q_k) m_k^2 \tag{4-3}$$

$$TQQD_{min}=\Sigma_k\ \text{freq}(q_k) m_k^2\ /|q_k|. \tag{4-4}$$

其中 $m_k = N - |q_k|$ 表示沒有被 q_k 查詢的資料項數目。假如在一組廣播順序下所有被相同查詢所讀取的資料皆能配置一起則 $TQQD$ 可達到理論上的最大值，亦是對每個查詢 q_k 都只有一個不為 0 的間隔 j，所以當 $l \neq j$ 時，則 $\delta_{k,l} = 0$ 且 $\delta_{k,l} = m_k$。假如在一組廣播順序下所有被相同查詢所讀取的資料皆能平均配置則 $TQQD$ 可達到理論上的最小值，對每個查詢 q_k 來說，$\delta_{k,j} = m_k / |q_k|, \forall j$。假如資料在廣播時沒有經過排程最佳化且每個查詢 q_k 所讀取資料皆是隨機分配，由大量法則可知，沒有被查詢 q_k 所讀取的連續資料項是以參數為 $|q_k|/(m_k+|q_k|)$ 的幾何分配。在這樣隨機廣播排程下 $TQQD$ 會變成：

$$TQQD_{random}=\Sigma_k\, freq(q_k)(2\ m_k^2\ /|q_k|+\ m_k\ ) \tag{4-5}$$

當 $|q_k|$ 增加時，若廣播排程未經最佳化處理則查詢的平均讀取時

間將急遽增加。在只有單筆查詢 q_k 的系統中，若所有 q_k 所讀取的資料項均能平均配置，則 *TQQD* 可達到理論上的最小值。若將 2 個查詢 q_k 所讀取的資料配置一起則 *TQQD* 會從 m_k^2 /$|q_k|$增加到 m_k^2 /($|q_k|$-1)，若再將第 3 個資料項加入則 *TQQD* 會從 m_k^2 /($|q_k|$-1) 增加到 m_k^2 /($|q_k|$-2)。當所有被讀取的資料項均能配置一起時會使 *TQQD* 達到 m_k^2。所以在查詢 q_k 下將第 i 個所讀取資料項配置一起的成本為

$$\text{Cost}(q_k, i) = \frac{m_k^2}{(|q_k| - i)(|q_k| - i + 1)} \qquad 1 \le i \le N - 1$$

平均成本 AvgCost(q_k) 則為

$$\text{AvgCost}(q_k) = \frac{1}{N-1}\sum_i \text{Cost}(q_k, i) = \frac{m_k^2}{|q_k|}$$

4.3 資料相似度與區斷(segment)相似度

定義三：任何介於兩筆資料 d_i、d_j 的資料相似度為：aff(d_i,d_j)=

$$\begin{cases} \sum_k \frac{m_k^{\,2}}{|q_k|}\text{QueryHas}(q_k,d_i)\text{QueryHas}(q_k,d_j)\text{freq}(q_k) & \\ & \text{for } i\neq j \\ 0 & \text{otherwise} \end{cases} \tag{4-6}$$

$$\text{where} \quad \text{QueryHas}(q_k,d_i)=\begin{cases}1 & \text{if } d_i \in q_k \\ 0 & \text{otherwise}\end{cases}$$

定義四：介於 S_i 右側和 S_j 左側間的區斷相似度為：SegAff(S_i,S_j)=

$$\begin{cases} \sum_k \left(\frac{\left(m_k-\left(\text{R}(S_i,k)+\text{L}(S_j,k)\right)\right)^2}{|q_k|} \right) \text{SegHas}(S_i\,,\,q_k) & \\ \text{SegHas}(S_j\,,\,q_k)\text{freq}(q_k) & \text{for } i\neq j \\ 0 & \text{otherwise} \end{cases} \tag{4-7}$$

假如至少有一筆被 q_k 要求的資料記錄在區斷 S_i 裡，SegHas(S_i,q_k) 的值為 1，若沒有則設為 0。R(S_i,k)表示在區斷 S_i 的右側起不為 q_k 讀取的連續資料項總數而 L(S_j,k) 表示在區斷 S_j 的左側起不為 q_k 讀取的連續資料項總數。若沒有任何資料被 q_k 所讀取則 R(S_i,k)和 L(S_j,k) 皆等於區斷 S_i 的長度。在兩個成對區斷邊的區斷相似度是由所有查詢 q_k 的相似度所構成。

4.4 用最小空隙法來合併區斷(Merge segment)

起初每個區斷都只有一筆資料項。每一次藉由連結兩個有最大相似度的區斷邊均可以得到一個新的區斷。我們運用 4.3 節所提出

的資料相似度及區斷相似度，將較大索引號碼的區斷合併到較小索引號碼的區斷。假如任何區斷在合併前需反向則位於區斷左側及右側連續不被查詢讀取的資料項數目且將需互換$\left[\mathrm{R}(S_i,k)，\mathrm{L}(S_i,k)\right]$，且它的廣播順序也須反向。

在合併 S_j 的資料記錄到 S_i 後，區斷 S_i 的參數會更新如下：

$$\mathrm{L}(S_i,k)=\mathrm{L}(S_i,k)+(1-\mathrm{SegHas}(S_i,q_k))\mathrm{L}(S_j,k)$$

$$\mathrm{R}(S_i,k)=\mathrm{R}(S_j,k)+(1-\mathrm{SegHas}(S_j,q_k))\mathrm{R}(S_i,k)$$

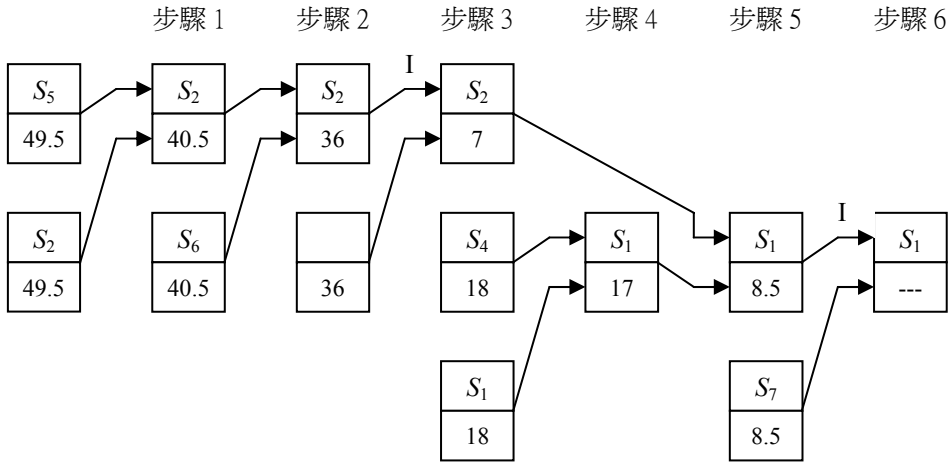
$$\mathrm{SegHas}(S_i,q_k)=\mathrm{SegHas}(S_i,q_k)\mid\mathrm{SegHas}(S_j,q_k)$$

在這些參數被改變之後，從 *SAM* 及 *ISAM* 得出的相似值若跟新形成區斷有關的話也將需加以更新。假如 d_i 屬於 q_k 所讀取的資料則將 SegHas(S_i,q_k)設為 1 而 R(S_i,q_k), L(S_i,q_k)設為 0。若 d_i 不屬於 q_k 所讀取的資料則把 SegHas(S_i,q_k)設為 0，而 R(S_i,q_k), L(S_i,q_k)均設為 1。接下來則依據介於每對區斷邊的相似度來建立相似度矩陣(*SAM* 和 *ISAM*)及連續合併有最大區斷相似度的區斷邊。下個例子可用來描述最小空隙法如何合併區斷：

(例)令有一組查詢的集合 $Q=\{q_1,q_2,q_3,q_4\}$，分別查詢的資料為：$q_1=\{d_2,d_3,d_5,d_6\}$, $q_2=\{d_1,d_2,d_4,d_5\}$, $q_3=\{d_4,d_5,d_6,d_7\}$, $q_4=\{d_1,d_2,d_3,d_7\}$ 而

頻率分別為 $freq(q_1) = 14, freq(q_2) = 8, freq(q_3) = 4, freq(q_4)=2$。

合併的過程如圖 4-1。步驟 1 中，*ISAM*[2][5]有最大的相似度 49.5 且符合 case I 的條件，所以將 S_2 和 S_5 合併成新的 S_2。合併之後，在 S_2 裡的參數，*SAM* 和 *ISAM* 都需被更新。在步驟二中，*SAM*[2][6]有最大的相似度 40.5 且亦符合 case I 的條件，所以將 S_2 和 S_6 合併成新的 S_2。合併之後，須更新 S_2 裡的參數，*SAM* 和 *ISAM*。步驟三中 *ISAM*[3][2]有最大相似度 36 也符合 case II 的條件，所以將 S_2 的反向和 S_3 正向合併成新的 S_2。合併之後，須更新 S_2 裡的參數，*SAM* 和 *ISAM*。在接下來的步驟都是以此類推，因而最後產生的排程為 $<d_3 \oplus d_2 \oplus d_5 \oplus d_6 \oplus d_4 \oplus d_1 \oplus d_7>$。



最後的廣播順序爲：

$$<<d_3 \overset{36}{\oplus} <<d_2 \overset{49.5}{\oplus} d_5> \overset{40.5}{\oplus} d_6>> \overset{17}{\oplus} <d_4 \overset{18}{\oplus} d_1>> \overset{8.5}{\oplus} d_7$$

圖 4-1 執行最小空隙法的合併步驟

4.5 使用改良後最小空隙法來降低時間與空間複雜度

在這個部份，我們建立一個改良型的最小空隙法來降低最小空隙法在時間與空間的複雜度。這個方法可分為兩個部份：第一部份是降低搜尋範圍，第二部份則為建立部份相似度矩陣。然而，第二部份需配合第一部份一起執行。

定義五：讓 S_{dummy} 表示一個虛設區斷，此虛設的資料區斷裏只包括一個被所有查詢請求所需讀取的虛設資料項。區斷 S_i 的虛設相似度是介於 S_i 的右側和 S_{dummy} 的左側並表示如下：

$$\text{Dummy}(S_i)=\text{SegAff}(S_i, S_{dummy}) \tag{4-8}$$

以上式為例，介於 S_i 左側及 S_{dummy} 右側的區斷相似度與 S_i 的反向區斷及 S_i^{-1} 的虛設相似度相同，表示如下：

$$\text{SegAff}(S_{dummy}, S_i) = \text{SegAff}(S_i^{-1}, S_{dummy}) = \text{Dummy}(S_i^{-1}) \tag{4-9}$$

提議一：對於每個成對的資料區斷 S_i 和 S_j 有下列的特性：

$\text{Dummy}(S_i) \geqq \text{SegAff}(S_i, S_j)$ 且

$\text{Dummy}(S_i^{-1}) \geqq \text{SegAff}(S_i^{-1}, S_j)$

證明：在每個查詢的區斷相似度中若減少不被查詢讀取的連續資料項總數則介於任何區斷側邊和 S_{dummy} 間的區斷相似度不會小於本身區斷側邊和其他區斷間的區斷相似度。

提議二：對任何資料區斷 S_i、S_j、S_k、S_l，有下列的定理：

假如 SegAff(S_i , S_j) ≧ Dummy(S_k)

則 SegAff(S_i , S_j) ≧ SegAff(S_k , S_l)

證明：從提議一得知。

在開始最小空隙法之前，我們先建立一個虛擬串列 *DumList*。它的每一虛擬項目均包含一個 *Seg* 指標連接到一個區斷，一個 *SegId* 是用來表示該區斷的號碼，一個 *next* 指標連接到下一個虛擬項目，一個 *previous* 指標接到上一個虛擬項目，一個 *indication* 旗標指出區段的順序是正向順序或反向順序，一個 *twin* 指標連結到相同擁有區斷但在不同的順序下的虛擬項目，而 *dummy* 則是區斷在此順序下的虛設相似度。

在 *DumList* 裡，每一個虛擬項目中的虛設相似度是由此區斷與虛設區斷間的相似值所計算得知的。假如 *Seg* 連接到 S_i 且從 *indication* 中得知此虛擬項目中的區斷是正向順序，則其虛設相似度為介於 S_i 右側與虛設區斷之間的區斷相似度，即為 Dummy(S_i)。若 *Seg* 連接到 S_i 且從 *indication* 中得知此虛擬項目中的區斷是反向順序，則其虛設相似度 $_i$ 左側及虛設區斷之間的區斷相似度，即為 Dummy(S_i^{-1})。

因為 *SAM* 和 *ISAM* 矩陣裡最大的區斷相似度即為 *DumList* 裡任兩個區斷右側之間的最大區斷相似度，故在每一次的區斷合併中，可從 *DumList* 中找出有最大區斷相似度的兩個虛擬項目並將其合併。若 $i<j$，從任兩個虛擬項目中挑選出來的區斷相似度有四種可能組合為(S_i, S_j)，(S_i^{-1}, S_j)，(S_i, S_j^{-1})和(S_i^{-1}, S_j^{-1})。藉由反轉第二個區斷並合併到第一個區斷可以連接兩個區斷的右側，故新形成的區斷可能為 $S_i \oplus S_j^{-1}$，$S_i^{-1} \oplus S_j^{-1}$， $S_i \oplus S_j$ 和 $S_i^{-1} \oplus S_j$。

此串列是依照虛設相似度的大小做降冪排列，如下表 1。建立 *N* 個 *node* 需 O(*NK*)的複雜度，若再依 *node* 由大而小排列則需要 O(*N*log(*N*)) 的複雜度，所以建立整個 *DumList* 的複雜度為 O(max(*NK*,*N*log(*N*)))。

表 4-1　為 *DumList* 裡依據虛設相似度的大小所排列後的結果

S_5	S_5^{-1}	S_2	S_2^{-1}	S_6	S_6^{-1}	S_3	S_3^{-1}	S_4	S_4^{-1}	S_1	S_1^{-1}	S_7	S_7^{-1}
58.5	58.5	54	54	40.5	40.5	36	36	27	27	22.5	22.5	13.5	13.5

4.5.1　減少搜尋空間

在每次合併之後均必須再次搜尋相似度矩陣裡的最大值。根據提議一，對於在 *DumList* 裡的每個資料項，若它的虛設相似度小於或等於最小修正相似度 w_{min} 則介於任何區斷與連接它右側區斷間的

區斷相似度不可能會大於 w_{min}，所以搜尋相似度矩陣中的最大值，只需搜尋在相似度矩陣裡虛設相似度大於 w_{min} 的虛擬項目。

開始的做法是把 *Dumlist* 裡的第二個虛擬項目當作是一個候選虛擬項目，然後搜尋在 *Dumlist* 裡介於此虛擬項目和擁有不同區斷號碼且有較大虛設相似度的虛擬項目之間的區斷相似值。在搜尋過程中，若有一較大值出現，則將其存入最小修正相似度 w_{min}。在此串列中一直重覆挑選下個資料項當做候選虛擬項目並搜尋在相似度矩陣裡的最大值直到候選資料項的虛設相似度小於或等於 w_{min}。

因為較大索引號碼的區斷永遠合併到較小索引號碼的區斷，所以當最佳的區斷對被找到後，有較大區斷號碼的虛擬項目會從 *Dumlist* 中刪除而兩個虛擬項目中有較小的區斷號碼的虛設相似度則須被更新，而 *Dumlist* 則須依新的虛設相似度新排列。

合併之後新形成的區斷都會有不同的 $R(S_i,q_k)$，$L(S_i,q_k)$ 及 $SegHas(S_i,q_k)$，在下次合併開始之前，介於新形成區斷及其他區斷之間四種不同合併順序的區斷相似度均需更新。

輸入： 一組資料項 *D*; 一組查詢 *Q*

輸出： 一組資料廣播排程

1. 起初 *DumList*, $SegHas(S_i,q_k)$, $L(S_i, q_k)$, $R(S_i, q_k) \forall i,k$.

2. *first_item= DumList.top, candidate=first_item->next,*

3.DO

$w_{min} \leftarrow -\varepsilon$

4. for each *temp_item* $\in$ [*DumList.top*,c*andidate*->previous]

{

if SegAff(temp_item.seg,(*candidate.Seg*)$^{-1}$) is not in its affinity matrix(either *SAM* or *ISAM*) then compute it and save back.

if SegAff(*temp_item.Seg*,(*candidate.Seg*)$^{-1}$)> w_{min}

{

w_{min}←SegAff(*temp_item.Seg*,(*candidate.Seg*)$^{-1}$)

*Item*1=Smaller_Id_Item(*temp_item.Seg,candidate.Seg*)

*Item*2=Larger_Id_Item(*temp_item.Seg,candidate.Seg*))

If *candidate.dummy*<= w_{min}. go to 6

}

candidate= candidate ->next

}

5. go to 4 if *candidate.dummy*> w_{min}.

6. merge the segments which linked by *Item*1 and *Item*2,

new segment will be *Item*1.*Seg* $\oplus$ (*Item*2.*Seg*)$^{-1}$

link the normal sequence of new segment to *Item*1.*Seg* and inverse sequence to *Item*1.*twin*.*Seg*

delete *Item*2 , *Item*2.*twin*

update SegHas(*Item*1.*Seg* ,q_k), L(*Item*1.*Seg*,*k*), R(*Item*1.*Seg*,*k*)

update *Item*1.*dummy*, *Item*1.*twin*.*dummy* and reposition *Item*1 and *Item*1.*twin* in *DumList*

update affinity value in *SAM* and *ISAM* only between the new formed segment and items in *DumList* which have dummy affinity larger than w_{min}

7. UNTIL *DumList* is empty

圖 4-2　改良後最小空隙法的完整演算法

限制在相似度矩陣中的可搜尋區斷數量可以大量地降低最大相似值的搜尋數，尤其當 N 的值變大時。然而，建立相似度矩陣仍需要 $O(N^2K)$的複雜度而在每次合併後更新相似度矩陣的複雜度仍然需要 $O(NK)$。

4.5.2　建立部份相似度矩陣

為了降低最小空隙法的記憶體空間及時間複雜度，我們提出了三個改善的方向。第一，先減少在建立相似度矩陣時需被計算的相似值數量。第二，降低相似度矩陣所需記憶體空間。第三，減少每次合併時所需更新的相似值數量。

當使用最小修正相似度 w_{min} 以降低搜尋數目後，我們亦能應用 w_{min} 值來部份建立相似度矩陣。既然大部份相似度矩陣中的值從來未曾被使用過也未曾被更新過，我們便能在有需要時才真正計算這

些相似值。

DumList 建立後，在處理合併開始前不需計算在相似度矩陣裡的任何相似值。介於任兩區斷側的相似值只在需要時才會被計算。在搜尋期間，介於在 *DumList* 的候選虛擬項目及其他虛擬項目的區斷相似度必需跟 w_{min} 做比較，以決定符合的相似值。從較大索引號碼的區斷合併到較小索引號碼的區斷之後，較大索引號碼的區斷則被直接刪除，而這些與較大索引號碼的區斷相關的相似值根本不須事先求出。

假如在處理合併之前不需建立相似度矩陣，它們能被建立成稀疏矩陣並能使記憶體空間需求大大的減少。

在經過每次合併後，在相似度矩陣裡的更新只限制在新形成的區斷和其他虛擬值大於 w_{min} 的區斷。圖 4-2 是整個改良後的最小空隙法完整演算法，並使用與上相同的例子來解釋此方法：

在每次合併步驟使用此種改善方法，相同的成對區斷側會基於最小空隙法而合併產生新的區斷，表 4-2 將有詳細的說明。在第一個步驟裡，需要有 4 次搜尋且只會搜尋到此區斷裡連接 S_6 的第五個虛擬項目需要計算 4 次相似度並且不需更新任何相似度及虛擬項目的虛設相似度。在第二步驟裡需要有 2 次搜尋且只需搜尋到此區斷

裡連接 S_6^{-1} 的第四個虛擬項目，*DumList* 無任何虛設相似度需更新，只有二個 SegAff(S_6,S_2)和 SegAff(S_6,S_2^{-1})的相似度要重新計算。在第三步驟裡需要有 2 次搜尋且只會搜尋到此區斷裡連接 S_6^{-1} 的第四個虛擬項目，*DumList* 無任何虛設相似度需更新且只有四個相似度要重新計算。在步驟三至六中搜尋次數、相似度計算次數等等都在表 4-2 有詳細說明。

假如使用一般的二維矩陣，使用最小空隙法需要 98 記憶體空間，而若使用稀疏矩陣則記憶體空間只需 12 個。使用這個改善的方法，總搜尋的次數為 28，也比使用最小空隙法時所耗費 224 次的搜尋較少。在每次步驟裡，有 14 個需被計算的虛設相似度及兩個需更新的虛設相似度，所以總共需計算的相似度數目為 48。而在使用最小空隙法中，如果建立完整的相似度矩陣需計算 84 個區斷相似度及 60 個更新虛設相似度，所以總共需要計算的相似度數目為 144。故使用此方法能降低所需計算相似度的數目。

表 4-2　使用改良後最小空隙法的合併步驟

	Merge	New segment	w_{min}	compute/ update	search	*DumList* Searched						
1	$S_2 \oplus S_5^{-1}$	S_2:<d_2,d_5>	49.5	4/0	4	S_5	S_5^{-1}	S_2	S_2^{-1}	S_6		
						58.5	58.5	54	54	40.5		
2	$S_2 \oplus S_6^{-1}$	S_2:<d_2,d_5,d_6>	40.5	2/0	2	S_2	S_2^{-1}	S_6	S_6^{-1}			
						60.5	58	40.5	40.5			
3	$S_2^{-1} \oplus S_3^{-1}$	S_2:<d_6,d_5,d_2,d_3>	36	2/0	2	S_2^{-1}	S_2	S_3	S_3^{-1}			
						58	49	36	36			
4	$S_1 \oplus S_4^{-1}$	S_1:<d_1,d_4>	18	8/4	12	S_2^{-1}	S_2	S_4	S_4^{-1}	S_1	S_1^{-1}	S_7
						49	45	27	27	22.5	22.5	13.5
5	$S_1 \oplus S_2^{-1}$	S_1:<d_1,d_4,d_6,d_5, d_2,d_3>	17	0/0	4	S_2^{-1}	S_2	S_1	S_1^{-1}	S_7		
						49	45	29	26.5	13.5		
6	$S_1^{-1} \oplus S_7^{-1}$	S_1:<d_3,d_2,d_5,d_6, d_4,d_1,d_7>	8.5	4/0	4	S_1^{-1}	S_1	S_7	S_7^{-1}			
						45	30	13.5	13.5			

第五章　效能評估

為了評估最小空隙法及[7]所提出的方法的效能，我們執行一連串的實驗分析來證明。5.1 節為整個模擬環境，5.2 節為實驗檔的產生與介紹，5.3 節為模擬分析。

5.1　模擬環境

我們的模擬平台，所使用的硬體為 Intel Celeron(R)、CPU 1.7GHz、RAM 256MB，使用的作業系統為 Microsoft Windows XP，模擬程式開發使用 JAVA。而在我們的廣播環境中，客戶端可透過上傳的頻道提出想要的資料到伺服器端，而送出的需求資料為兩個以上的資料項，則為「多重資料項要求」的廣播模式，假設這些需求資料項是「固定大小」、「長度相等」，廣播環境的伺服器端接收到每個客戶端要求的資料項，會放到伺服器上進行查詢存取頻率的計算，伺服器依據廣播資料查詢存取頻率及以往收集到的歷史資料進行統計分析，運用最小空隙法的廣播排程演算法，產生出一組較佳的廣播順序讓客戶端的查詢讀取時間能降到最低。

5.2　實驗檔的產生與介紹

透過造檔程式產生出實驗所需的資料庫檔案，而造出的資料檔主要用來模擬行動客戶端對於廣播頻道上的資料之需求變化及分配情形。就查詢的存取頻率分佈狀況而言，主要可分為以下四種型態之資料庫檔案：均勻分佈(Uniform Distribution)資料檔、常態分配(Normal Distribution)資料檔、指數分配(Exponential Distribution)資料檔、及偏態分配(Zipf Distribution)資料檔，這四種型態的資料檔均用來模擬無線廣播環境裡行動客戶端讀取資料的情形。

表 5-1　檔名符號定義表

檔名符號	表示的意義
zd	偏態分配資料檔
q	表示行動客戶端的查詢個數
s	表示選擇率(%)
t	表示存取偏斜程度

例如：zd500q100s002t25 即表示廣播資料項有 500 筆，為偏態分配，客戶端的查詢數有 100 個，選擇率為 2%表示每一個查詢所讀取的資料有 10 筆(即廣播資料項 500 筆乘於 2%)，存取偏斜程度為 2.5。

將產生的資料檔分為三種，第一種為改變存取偏斜程度由 1.5~4.0，

第二種為每一個查詢所讀取的資料項由 20~60 筆，第三種為改變廣播資料項由 100~1000，存取偏態為 1.0，第四種為改變廣播資料項由 100~1000，存取偏態為 2.0，如表 5-2 為本論文實驗用的資料檔型態。

表 5-2　本論文實驗用的資料檔型態

第一種	第二種
zd1000q100s002t15	zd1000q100s002t20
zd1000q100s002t20	zd1000q100s003t20
zd1000q100s002t25	zd1000q100s004t20
zd1000q100s002t30	zd1000q100s005t20
zd1000q100s002t35	zd1000q100s006t20
zd1000q100s002t40	
第三種	第四種
zd100q100s002t10	zd100q100s002t20
zd200q100s002t10	zd200q100s002t20
zd300q100s002t10	zd300q100s002t20
zd400q100s002t10	zd400q100s002t20
zd500q100s002t10	zd500q100s002t20
zd600q100s002t10	zd600q100s002t20
zd700q100s002t10	zd700q100s002t20
zd800q100s002t10	zd800q100s002t20
zd900q100s002t10	zd900q100s002t20

zd1000q100s002t10	zd1000q100s002t20

資料檔產生後我們將其代入到最小空隙演算法程式中，並與隨機方式所產生的廣播順序及 Chung's 方法下產生的廣播順序做平均存取時間的比較。

5.3 模擬分析

在這個部份，我們透過各種實驗來比較評估我們所提出的方法。評估的標準有：第一，廣播順序下每個查詢的平均存取時間。第二為系統在合併處理過程中所耗費的時間。

在模擬實驗中分別以不同的資料項總數和查詢來執行。每個查詢的存取頻率則用不同的 skewness 參數在 zipf 分配下進行實驗。

首先以第一種資料型態來模擬實驗，以資料項 1000 筆，客戶端查詢 100 筆，每筆查詢所包含資料個數 20 個，存取偏斜程度分別為 1.5、2.0、2.5、3.0、3.5、4.0 做比較。表 5-3 為實驗後數據。

表 5-3　在不同偏斜度中，三種方式的平均存取時間

skewness	**1.5**	**2**	**2.5**	**3**	**3.5**	**4**
Chung's	606.648	565.858	539.198	532.464	524.983	522.16
MG	595.707	555.858	533.214	527.555	522.742	520.907
Random	948.247	954.087	953.637	957.183	947.857	945.817

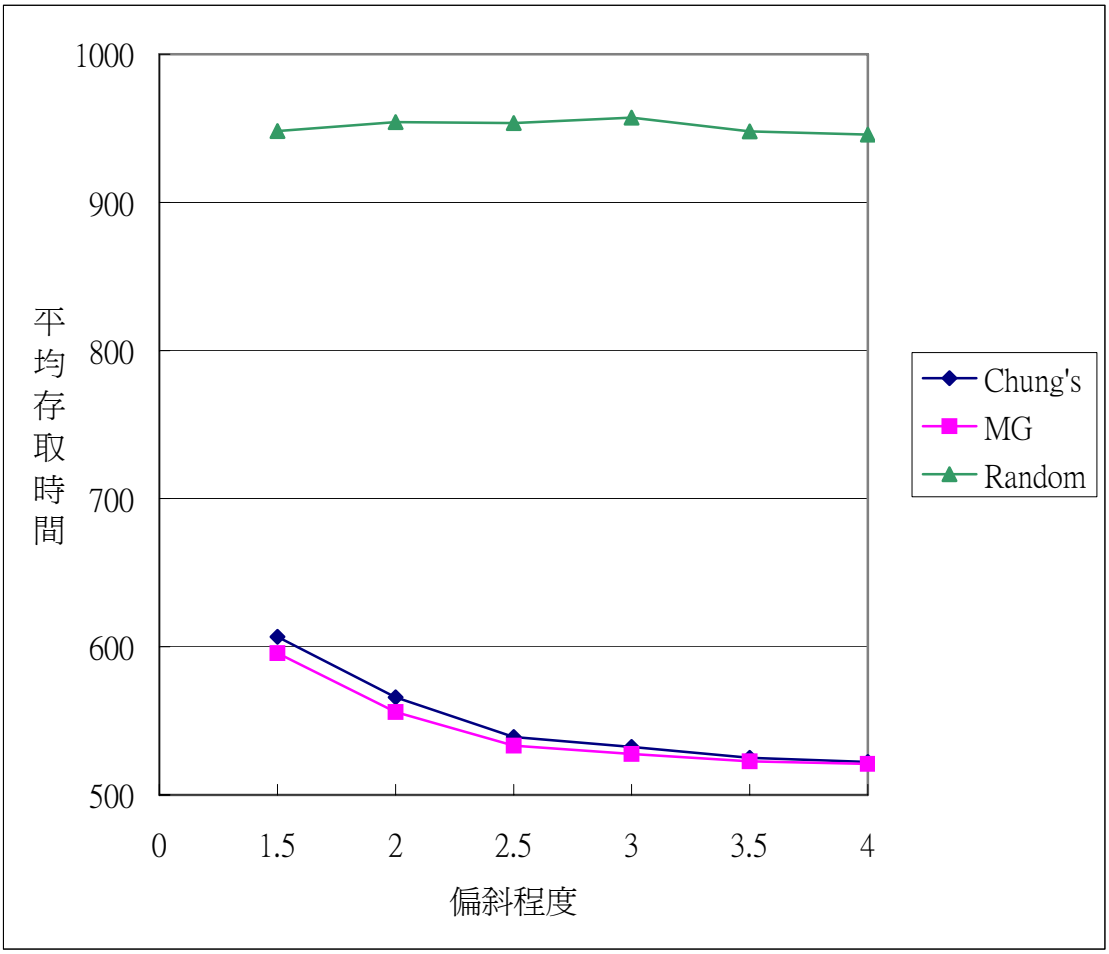


圖 5-1　依據各種不同偏斜程度下的平均存取時間

在圖 5-1 顯示平均存取時間的結果，用查詢的存取頻率偏斜度來比較各種方法。模擬數據為 100 個查詢隨機讀取 1000 個資料記錄，結果顯示出在不同的偏斜度上使用最小空隙法配置出的廣播順序能有效降低查詢的讀取時間。

再者以第二種資料型態實驗，使用資料項 1000 筆，客戶端查詢 100 筆，每筆查詢所包含資料個數分別為 20、30、40、50、60

筆做為比較的標準。

表 5-4　在查詢不同的資料數，三種方式的平均存取時間

Query	**20**	**30**	**40**	**50**	**60**
Chung's	556.11	596.588	619.691	640.879	658.145
MG	551.262	583.546	609.966	631.592	651.546
Random	958.253	967.561	974.253	981.631	984.644

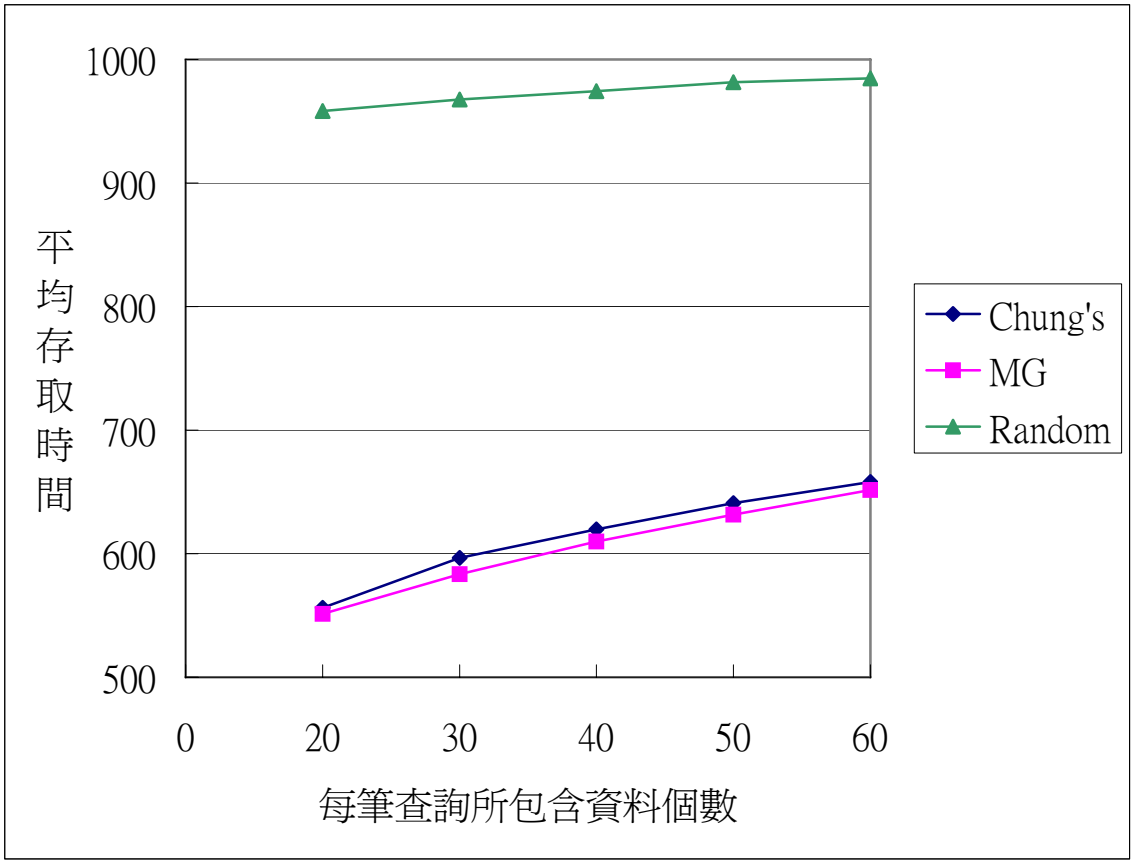


圖 5-2　依據每個查詢所存取不同資料個數下的平均存取時間

在圖 5-2 顯示平均存取時間的結果，在 1000 筆資料和 100 個查詢下存取 20 個資料記錄在不同的查詢存取頻率偏斜度。最小空隙法在考慮所有偏斜度是有較好的效能。在 100 個查詢百分之二的資

料項，它的偏斜程度皆等於 2。

以第三種資料型態實驗，將變動其資料個數分別為 100、200、300、400、500、600、700、800、900、1000 筆，而客戶端查詢則為 100 筆，每筆查詢所包含資料個數 20 個，存取偏斜程度為 1.0 做為實驗依據。表 5-5 為實驗後數據。

表 5-5　在資料個數不同的情況下，三種方式的平均存取時間

資料個數	100	200	300	400	500	600	700	800	900	1000
chung's	54.169	116.751	183.452	257.196	331.637	400.706	477.098	553.296	622.378	703.028
MG	53.332	114.659	180.219	252.775	326.063	396.635	470.751	545.407	618.774	689.697
Random	55.41	123.786	205.891	304.745	411.726	521.66	620.914	738.878	845.408	954.589

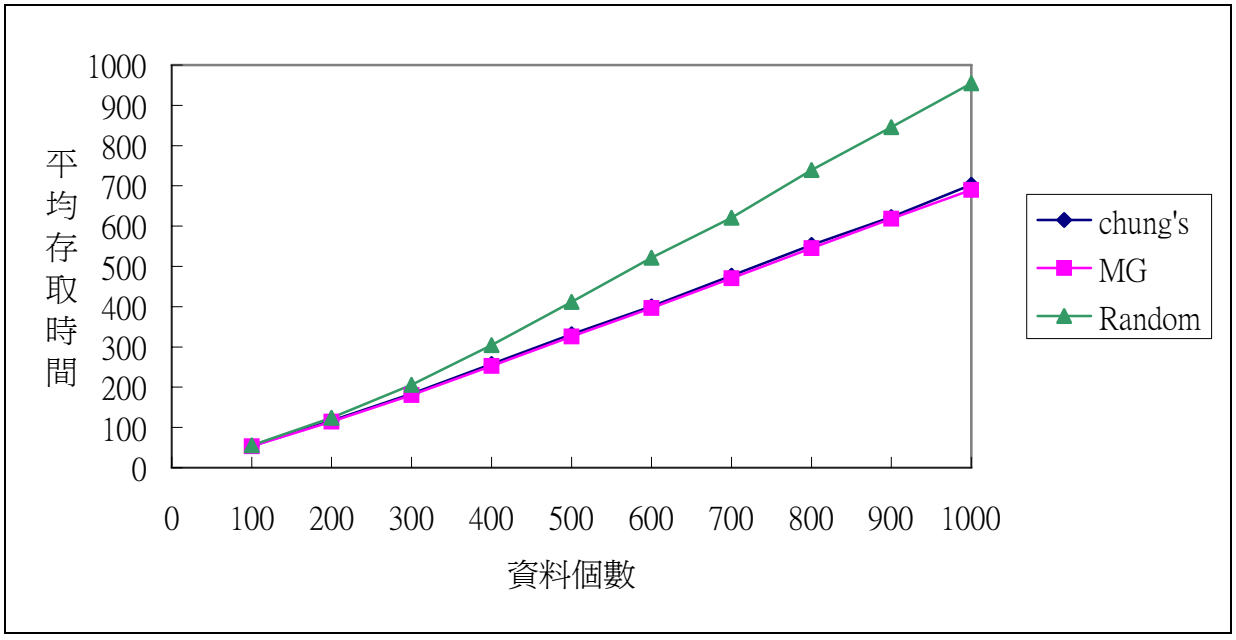


圖 5-3　依據不同資料個數下的平均存取時間

在圖 5-3 顯示平均存取時間的結果，在資料個數不同分別為 100、200、300、400、500、600、700、800、900、1000 筆，而查詢則為 100 筆，每筆查詢所包含資料個數 20 個，存取偏斜程度為 1.0 的實驗環境下，最小空隙法在縮短平均存取時間下仍然有不錯的效能。

最後，再以第四種資料型態做為實驗，依舊變動其資料個數分別為 100、200、300、400、500、600、700、800、900、1000 筆和客戶端查詢 100 筆，每筆查詢所包含資料個數 20 個，存取偏斜程度則改為 2.0 做為實驗依據。表 5-6 為實驗後數據。

表 5-6　在資料個數不同的情況下，三種方式的平均存取時間

	100	**200**	**300**	**400**	**500**	**600**	**700**	**800**	**900**	**1000**
chung's	52.129	106.805	163.743	217.521	276.855	337.19	389.578	448.649	501.723	557.799
MG	51.862	106.071	161.99	216.352	274.779	330.834	385.662	443.176	493.538	552.65
Random	55.315	120.308	222.614	303.58	394.309	514.44	624.542	734.767	854.233	953.911

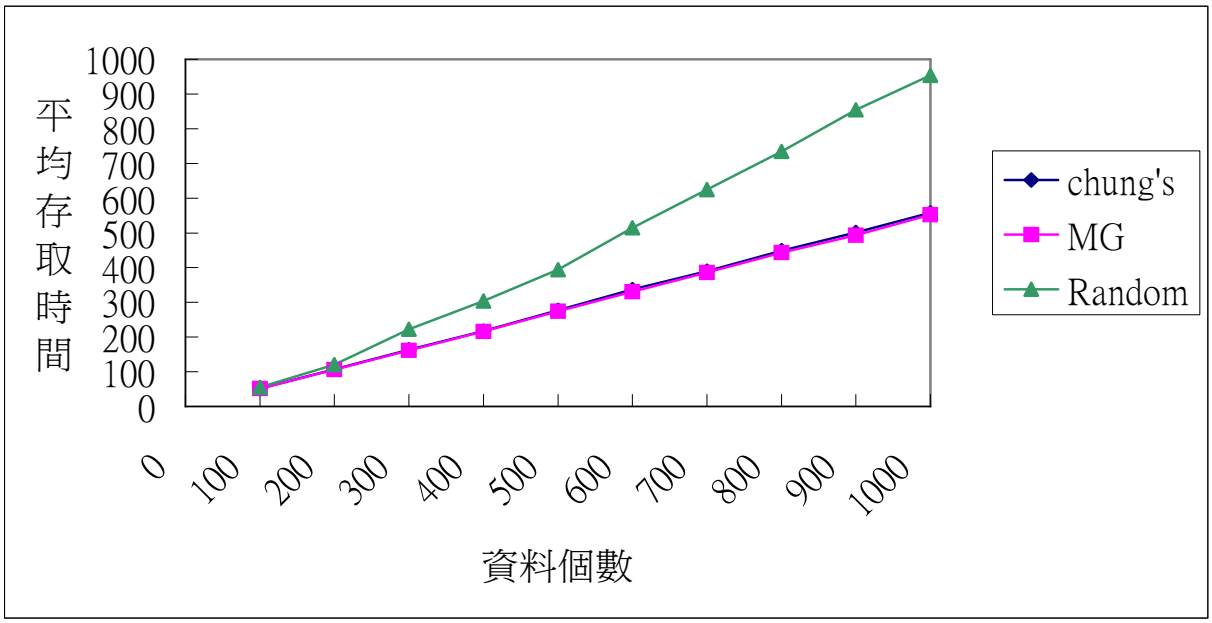


圖 5-4　依據不同資料個數下的平均存取時間

在圖 5-4 顯示平均存取時間的結果，在資料個數不同分別為 100、200、300、400、500、600、700、800、900、1000 筆和客戶端查詢 100 筆，每筆查詢所包含資料個數 20 個，存取偏斜程度則改為 2.0 的實驗環境下，結果顯示出我們所提出的最小空隙法比 chung's 所提出的叢集法在實驗數據看來，效果是較好的。

表 5-7 是顯示出在改良後的最小空隙法下，所有的區斷相似度總和(包含建立完整的 SAM 及 ISAM 及更新區斷相似度)能大幅度的縮短。我們使用 100 個查詢存取 20 筆資料記錄在查詢存取頻率為 2.0 下進行模擬。結果顯示出在處理合併步驟之前，計算相似度數

量及更新數量會少於建立完整的相似度矩陣。

表 5-7　依據不同的資料個數所計算的區斷相似度總和

資料個數	100	200	400	600	800	1000
建立完整的 SAM 及 ISAM	2E4	8E4	3.2E5	7.2E5	1.2E6	2E6
改良後最小空隙法	1556	3748	12338	26794	39972	49314

在改良後的最小空隙法裡，經過每次合併後的搜尋次數會少於在相似度矩陣中已被決定的值，所以在每次合併時搜尋最大數也會少於在相似度矩陣中所需的記憶體空間。

表 5-8 顯示出當資料個數越多時，建立完整的 SAM 及 ISAM 矩陣時需較大的記憶體，若能使用改良後的最小空隙法能大大降低記憶體空間。

表 5-8　依據不同的資料個數所需記憶體空間

資料個數	100	200	400	600	800	1000
建立完整的 SAM 及 ISAM	2E4	8E4	3.2E5	7.2E5	1.2E6	2E6
改良後最小空隙法	91	98	153	378	465	496

第六章 結論

在本文，我們建立一個叢集的方法來合併無線廣播資料裡多個客戶端查詢的資料項。先前使用的區段相似度需評估不同區斷間每對資料項的相似度。然而，我們主要的核心是沿用資料相似度方法再發展如何降低查詢的平均存取時間及合併每對資料的權重值，我們的區斷相似度模式是從資料相似度所修改後的模式但不需考慮每個介於不同區斷間的所有資料項彼此的相似度。透過各種實驗比較，我們所建立的區斷相似度均能更有效率地找出較佳的資料項排程配置。

由於最小空隙法在計算區斷相似度時只考慮合併區間的相似度，所以在計算區斷相似度上時間複雜度能有效的降低。

為進一步降低時間與空間的複雜度，我們建立一個改良型的最小空隙法藉由部份建立相似度矩陣和使用一個虛擬鏈結串列來降低搜尋空間讓計算相似度的次數達到最少。透過我們的實驗結果顯示，使用改良型的最小空隙法確能有效地降低時間複雜度及空間複雜度。

參　考　文　獻

[1] S. Acharya, R. Alonso, M. Franklin, and S. Zdonik, "Broadcast disks: Data mangement for asymmetric communication environments," *Proceedings of ACM SIGMOD Conference*, Pages:199-210, 1995

[2] S. Acharya, M. Franklin, and S. Zdonik, "Disseminating updates on broadcast disks," *Proceedings of Very Large Data Bases Conference*, Pages:354-365, 1996

[3] Demet Aksoy and Mason Sin-Fai Leung, "Pull vs Push: A Quantitative Comparison for Data Broadcast," *Global Telecommunications Conference,* Volume 3, Pages:1464-1468, 2004

[4] A. A. Bertossi, M. C. Pinotti, and S. Ramaprasad, "Optimal multi-channel data allocation with flat broadcast per channel," *Proceedings of the 18th International Parallel and Distributed Processing Symposium*, Pages:18, 2004

[5] Y. C. Chehadeh, A. R. Hurson, and L.L. Miller, "Energy-Efficient Indexing on a Broadcast Channel in a Mobile Database Access System," *Proceedings of the The Internation Conference on Information Technology: Coding and Computing*, Pages:368-374, 2000

[6] Po-Jen Chuang and Ching-Yueh Hsu, "An Efficient Cache Invalidation Strategy in Mobile Environments," *Proceedings of the 18th International Conference on Advanced Information Networking and Application*, Pages:260 -263, 2004

[7] Yon Dohn Chung, Su Ho Bang, and Myoung Ho Kim, "An efficient broadcast data clustering method for multipoint queries in wireless information systems," *The Journal of Systems and Sofeware*, Volume 64, Pages:173-181, 2002

[8] Yon Dohn Chung and Myoung Ho Kim, "Effective Data Placement for Wireless Broadcast," *Distributed and Parallel Databases*, Volume 9, Pages:133-150, 2001

[9] Y. D Chung and M. H. Kim, "An index replication scheme for wireless data broadcasting," *Journal of Systems and Software*, Volume 51, NO.3, Pages:191-199, 2000

[10] Omer Erdem Demir and Demet Aksoy, "Energy-Efficient Broadcast-based Event Update Dissemination," *Pertormance*, *Computing*, *and Communications*, 2004 *IEEE International Conference on* 2004, Pages: 477-482

[11] Qiu Fang, Susan V. Vrbsky, Yu Dang, and Weigang Ni, "A Pull-Based Broadcast Algorithm that Considers Timing Constraints," *Proceedings of the 2004 International Conference on Parallel Processing Workshops*, Pages:46-53, 2004

[12] Chih-Hao Hsu, Guanling Lee, and Arbee L. P. Chen, "A Near Optimal Algorithm for Generating Broadcast Programs on Multiple Channels," *Proceedings of the tenth International Conference on Information and Knowledge Management*, Pages:303-309, 2001

[13] Chih-Hao Hsu, Guanling Lee, and Arbee L. P. Chen, "Index and Data Allocation on Multiple Broadcast Channels Considering Data Access Frequencies," *Proceedings of the Third International Conference on Mobile Data Management*, Pages:87-93, 2002

[14] Chih-Lin Hu and Ming-Syan Chen, "Adaptive Balanced Hybrid Data Delivery for Multi-Channel Data Broadcast," *Proceedings of the IEEE International Coference on Communication* (ICC-2002), Volume 2, Pages:960-964, 2002

[15] Qinglong Hu, Wang-Chien Lee, and Dik Lun Lee, "Indexing Techniques for Wireless Data Broadcast under Data Clustering and Scheduling," *Proceedings of the Eighth International Conference on Information and Knowledge Management*, Pages:351-358, 1999

[16] Hiun-Long Huang and Ming-Syan Chen, "Broadcast Program Generation for Unordered Queries with Data Replication," *Proceedings of the 2003 ACM Symposium on Applied Computing*, Pages:866-870, 2003

[17] Yan Huang and Yann-Hang Lee, "AN EFFICIENT INDEXING METHOD FOR WIRELESS DATA BROADCAST WITH DYNAMIC UPDATES," *Communications*, *Circuits and Systems and West sino Expositions*, IEEE 2002 International Conference on Volume 1, Pages:358-362, 2002

[18] Jiun-Long Huang and Ming-Syan Chen, "Dependent Data Broadcasting for Unordered Queries in a Multiple Channel Mobile Environment," *IEEE Transaction on Knowledge and Data Engineering*, Volume.16, NO.9, Pages:1143-1156, 2004

[19] Jiun-Long Huang, Ming-Syan Chen, and Wen-Chih Peng , "Broadcasting Dependent Data for Order Queries Without Replication in a Multi-Channel Mobile Environment," *Proceedings of the 19th International Conference on Data Engineering*, Pages:692-694, 2003

[20] Jen-Jou Hung and Yungho Leu, "Efficient Index Caching Schemes for Data Broadcasting in Mobile Computing Environments," *Proceedings of the 14th International Workshop on Database and Expert Systems Applications*, Pages:139-143, 2003

[21] Jen-Jou Hung and Yungho Leu , “An Energy Efficient Data Reaccess Scheme for Data Broadcast in Mobile Computing Environments,” *Proceedings of the 2003 International Conference on Parallel Processing Workshops*, Pages:5-12, 2003

[22] T. Imielinski, S. Viswanathan, and B. R. Badrinath, “Data on air: Organization and access,” *IEEE Transactions on Knowledge and Data Engineering*, Volume 9, NO.3, Pages:353-372, 1997

[23] T. Imielinski, S. Viswanathan, and B. R. Badrinath, ”Energy efficient indexing on air,” *Proceedings of the 1994 ACM SIGMOD international conference on Management of data*, Pages:25-36, 1994

[24] Kuen-Fang Jea and Ming-Hui Chen, “A Data Broadcast Scheme Based on Prediction for The Wireless Environment,” *Proceedings of the Ninth international Conference on Parallel and distributed Systems*, Pages:369-374, 2002

[25] Ji Yeon Lee, Yon Dohn Chung, Yoon Joon Lee, and Myoung Ho Kim, “Gray Code Clustering of wireless data for partial match queries,” *Journal of Systems Architecture*, Volume 47, Pages:445-458, 2001

[26] Guanling Lee and Shou-Chih Lo, “Broadcast Data Allocation for Efficient Access of Multiple Data Items in Mobile Environments,” *Mobile Networks and Applications*, Volume 8, NO.4, Pages:365-375, 2003

[27] SangKeun Lee and Masaru Kitsuregawa, “Efficient Processing of Wireless Read-only Transactions in Data Broadcast,” *Proceedings of the 12th Int’1 Wrkshp on Research Issues in Data Engineering: Engineering e-Commerce/e-Business Systems*, Pages:101-111, 2002

[28] Guanling Lee, Meng-Shin Yeh, Shou-Chih Lo, and Arbee L. p. Chen, “A Strategy for Efficient Access of Multiple Data Items in Mobile Environments,” *Proceedings of the Third International Conference on Mobile Data Management*, Pages:71-78, 2002

[29] Sung-Hwa Lim and J.-H. Kim, “Real-time broadcast algorithm for mobile computing,” *The Journal of Systems and Sofeware*, Volume 69, Pages:173-181, 2004

[30] Chi-Wai lin, Haibo Hu, and Dik-Lun Lee, “Adaptive Real-time Bandwidth Allocation for Wireless Data Delivery,” *Wireless Networks*, Volume 10, NO.2, Pages:103-120, 2004

[31] Shou-Chih Lo and Arbee L.P Chen , “An adaptive Access Method for Broadcast Data Under an Error-Prone Mobile Environment,” *IEEE Transactions on Knowledge and Data Engineering*, Volume 12, NO.4, 2000

[32] Shou-Chih Lo and Arbee L. P. Chen, “Optimal Index and Data Allocation in

Multiple Broadcast Channels," *Proceedings of the 16th International Conference on Data Engineering*, Pages:293-302, 2000

[33] Weigang Ni , Qiu Fang, and Susan V. Vrbsky , "A Lazy Data Request Approach for On-demand Data Broadcasting," *Proceedings of the 23rd International Conference on Distributed Systems Workshops*, Pages:790-796, 2003

[34] Wen-Chih Peng and Ming-Syan Chen , "Developing Data Allocation Schemes by Incremental Mining of User Moving Patterns in a Mobile Computing System," *IEEE Transactions on Knowledge and Data Engineering*, Volume 15, NO.1, Pages:70-85, 2003

[35] Wen-Chih Peng and Ming-Syan Chen, "Efficient Channel Allocation Tree Generation for Data Broadcasting in a Mobile Computing Environment," *Wireless Networks*, Volume 9, NO.4, Pages:117-129, 2003

[36] Wen-Chih Peng, Jiun-Long Huang and Ming-Syan Chen, "Dynamic Leveling: Adaptive Data Broadcasting in a Mobile Computing Environment," *Mobile Networks and Applications*, Volume 8, NO.4, Pages:355-364, 2003

[37] Bryan Hin Cheung Poon, Kwok-Wa Lam, and Victor C. S. Lee, "Server-side Broadcast Transaction in Mobile Computing Environment," *Proceedings of the International Conference on Information Technology: Coding and Computing*, Volume 1, Pages:511-515, 2004

[38] OBV Ramanaiah and Hrushikesha Mohanty, "NICD: A Novel Indexless Wireless On-Demand Data Broadcast Algorithm," *Proceedings of the Internation Conference on Information Technology: Coding and Computing*, Volume 2, Pages:730-734, 2004

[39] Navrati Saxena, Kalyan Basu, and Sajal K. Das, "Design and Performance Analysis of a Dynamic Hybrid Scheduling Algorithm for Heterogeneous Asymmetric Environments," *Proceedings of the 18th International Parallel and distributed Processing Symposium*, Pages:223, 2004

[40] C. Su, L. Tassiulas, and V. J. Tsotras, "Broadcast scheduling for information distribution," *Wireless Networks*, Volume 5, NO.2, Pages:137-147, 1999

[41] K. Tan and J. X. Yu, "Generating broadcast programs that support range queries," *IEEE Transactions on Knowledge and Data Engineering*, Volume 10, NO.4, Pages:668-672, 1998

[42] Duc A. Tran, Kien A. Hua, and Ning Jiang , "A Generalized Air-cache Design for Efficiently Broadcasting on Multiple Physical Channels," *Proceedings of the 2001 ACM Symposium on Applied Computing*, Pages:387-392, 2001

[43] Shuoi Wang and Hsing-Lung Chen, "Near-Optimal Data Allocation Over Multiple Broadcast Channels," *Networks*, 2004.(ICON 2004). *Proceedings.*

12th IEEE International Conference on Volume 1, Pages:207-211, 2004

[44] Arthur C. S. Wong, Kwok-Wa Lam, Karmen K. M. Ho, and Victor C. S. Lee, "Using Lock-based Checking Protocol for Efficient Data Broadcast in Mobile Environments," *Proceedings of the 24th International Conference on Distributed Computing Systems Workshops*, Pages:484-489, 2004

[45] Xiao Wu and Victor C. S. Lee, "Preemptive Maximum Stretch Optimization Scheduling for Wireless On-Demand Data Broadcast," *Proceeding of the International Database Engineering and Applications Symposium*, Pages:413-418, 2004

[46] Jianliang Xu, Wang-Chien Lee, and Xueyan Tang , "Exponential Index : A Parameterized Distributed Indexing Scheme for Data on Air," *Proceedings of the 2nd International Conference on Mobile Systems, Applications, and Services*, Pages:153-164, 2004

[47] Jialiang Xu, Baibua Zheng, Wang-Chien Lee, and Dik Lun Lee, "Energy Efficient Index for Querying Location-Dependent Data in Mobile Broadcast Environments," *Proceedings of the International Conference on Data Engineering*, Pages:239-250, 2003

[48] Etsuko Yajima, Takahiro Hara, Masahiko Tsukamoto, and Shojiro Nishio , "Scheduling and Caching Strategies for Broadcast Correlated Data," *Proceeding of the 2001 ACM Symposium on Applied Computing*, Volume 9, NO.1, Pages:22-28, 2001

[49] Wai Gen Yee and Shamkant B. Navathe, "Efficient Data Access to Multi-Channel Broadcast Programs," *Proceedings of the Twelfth International Conference on Information and Knowledge Management*, Pages:153-160, 2003

[50] Jianting Zhang and Le Gruenwald, "Optimizing Data Placement Over Wireless Broadcast Channel For Multi-Dimensional Range Query Processing," *Proceedings of the 2004 IEEE International Conference on Mobile Data Management*, Pages:256-265, 2004

[51] Jianting Zhang and Le Gruenwald, "Efficient Placement of Geographical Data Over Broadcast Channel for Spatial Range Query Under Quadratic Cost Model," *Proceedings of the 3rd ACM International Workshop on Data Engineering for Wireless and Mobile Access*, Pages:30-37, 2003

[52] Ji Xu Yanmin Zhu, Jianliang Xu, and Bo Li Lionel M. Ni , "A Cooperative Caching Algorithm for Multi-Cell Data Broadcasting," *Communications*, 2004 *IEEE International Conference* on Volume 7, Pages:4072-4076, 2004