南　華　大　學

資訊管理學系碩士論文

使用 KL 演算法來解決多頻道的無線廣播排程資料分割的問題

Using KL Algorithm for Multi-Channel Wireless Broadcast Partitioning

研 究 生：李建磐

指導教授：吳光閔　博士

中華民國　九 十 四 年 六 月

使用 KL 演算法來解決多頻道的無線廣播排程資料分割的問題

Using KL Algorithm for Multi-Channel Wireless Broadcast Partitioning

研　究　生：李建磐　　Student : Chien-Pan Lee

指導教授：吳光閔博士　Advisor : Dr. Guang-Ming Wu

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

使用 KL 演算法來解決多頻道的無線廣播排程

資料分割的問題

研究生：李建盤

經考試合格特此證明

口試委員：蔡德讓

吳光閔

何陳勇

指導教授：吳光閔

系主任(所長)：資訊管理系系主任吳光閔

口試日期：中華民國　94　年　6　月　30　日

誌　　謝

寫誌謝表示碩士論文已經順利完成，首先要感謝我的父母對我的勉勵及栽培，讓我沒有後顧之憂全心全意的在學業上衝刺；再者要感謝指導教授 吳光閔老師，在這二年來對學生熱心的指導，無論是在報告的方式、研究領域上的指導，從討論的過程中能夠獲益良多，老師也分享很多學習上的經驗和生活上的處事態度，帶給學生很深刻的體會，從論文的問題和解決的方法都仰賴老師細心的教導，使得論文可以順利地完成。此外，感謝口試委員蔡德謙教授及何漢彰教授在口試時，對學生的論文提出的一些建議和改進的地方，使得本論文能夠更加完善。

在這二年研究生涯當中，也要感謝翟本瑞老師對學生的鼓勵，還有徐立群老師無論在學業上或生活上，不斷地給學生勉勵及關心，並且在學業上給學生很大的協助。也特別感謝室友文天及宗聖在生活上、學業上的互相勉勵及幫助，還有感謝研究室的夥伴如卿、元安、美倫、俊杰、閔皓、乾訓、明哲、建榮，以及無線廣播小組的學弟妹們，在生活上和學業上一起學習、成長。另外還要特別感謝鈺明在學業上給我加油打氣，並且花時間幫我校稿，在此由衷感謝她，最後感謝資管所的同學和學弟妹們，有幸可以和大家一起在同一個環境中成長，獻上最真摯的祝福給所有認識的朋友，能夠認識以上各位真好。

李建磐 僅識

于南華大學

民國九十四年六月

使用 KL 演算法來解決多頻道的無線廣播排程資料分割的問題

研究生 ： 李建磐　　指導教授 ： 吳光閔 博士

南華大學資訊管理學系碩士班

摘要

隨著行動科技時代的來臨，無線通訊網路與行動裝置技術正以飛快的速度前進與發展，使行動使用者隨時隨地都能方便地獲得想要的資訊。在無線網路的環境之下，以目前的技術來看，無線網路技術的頻寬仍比有線網路技術的頻寬小很多；因此，廣播是一項節省頻寬浪費的有效技術！

在多頻道的無線廣播環境中，將行動使用者所要求的資料項目平均分配到數個頻道上面，這樣一來可以縮短伺服器的廣播週期。當行動使用者所要求的資料集合分配在不同的頻道上，且位於相同的廣播時間點就會發生資料衝突。因為，發生資料衝突時，行動使用者必須等到下一個廣播週期才能再接收所要的資料項目，如此一來便會增加行動使用者資料項目的總存取時間。

在本文中，我們將 KL 演算法[18]運用於多頻道的無線廣播環境中，平均分配用戶端所請求的多重資料於多頻道上，以降低資料衝突發生的機率。從實驗結果顯示我們所使用的 KL 演算法可有效地降低資料發生衝突的機率：均勻分配比隨機分割法提升 27%的效能、常態分配提升 77%的效能，指數分配平均提升 82%的效能。

關鍵字: 無線網路、無線廣播

Using KL Algorithm for Multi-Channel Wireless Broadcast Partitioning

Student：Chien-Pan Lee　　Advisors：Dr. Guang-Ming Wu

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

ABSTRACT

With the coming time of mobile technologies, the Wireless Communication Network and the Mobile Unit Techniques are going fast step and amazingly development which make mobile technique user can get the desired information anywhere and anytime conveniently. Under the environment of wireless network, taking an overview of nowadays techniques, the bandwidth of wireless network techniques remain less than the wire network. Therefore, broadcast is a much more efficiency technique to save the use of bandwidth!

In the environment of multi-channel wireless broadcast, configured the required data items on several channels averagely can make the interval of broadcast shorten. Whenever the data integration of users' desired been configured on different channel but on the same time-point of broadcast will the data conflict occur. Since the data conflict occurs, the mobile users have to wait for the next broadcast interval may them receive the desired data items. This will increase the total access time of data items of mobile users'.

In this context, we take use of KL algorithm [18] expression at multi-channel wireless broadcast environment. Configuring the multiple data of client’s required on multi-channel to decrease the rate of data conflict occurring. From the result of experience shows the KL algorithm expression what we take can effectively decrease the rate if data conflict: raising the performance of 27% at equalized configuration than random split, 77% at normalized configuration, and 82% at log configuration.
Keywords: Wireless Network, Wireless Broadcast

目　錄

表　　目　　錄

圖　　目　　錄

第一章 緒論

在網際網路的發展與進步之下，使人們在資訊的交流及傳遞上更為方便，而網路的發展更從有線網路發展到無線網路的通訊環境，帶給人們超乎所想的便利和不受限的操作空間，只要有無線通訊設備和行動裝置就可以利用無線網路與各個區域進行通訊，方便人們隨時隨地接收所需資訊。無線通訊的使用已相當普及，也進一步地成為人們日常生活中的必需品，扮演著相當重要的角色。

第一節 無線廣播的環境介紹

在無線廣播的環境中，依網路的架構可以分成 *Ad Hoc*[5, 6, 11, 13, 19, 29]和*Client-Server*[1, 2, 3, 4, 7, 8, 9, 10, 12, 14, 15, 16, 17, 18, 20, 21, 22, 23, 24, 25, 26, 27, 28, 30, 31, 32, 33]的網路環境。

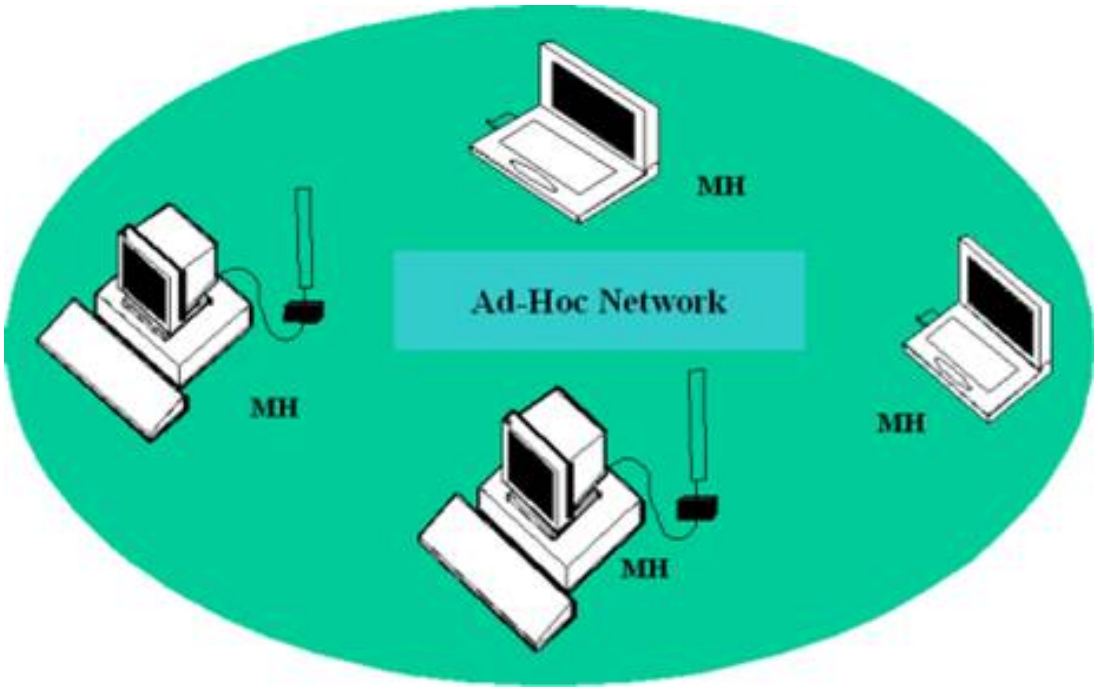

圖1-1 Ad Hoc的網路環境

上圖1-1是 Ad Hoc的網路環境架構圖，Ad Hoc 無線網路是由一群可任意移動的行動主機(MH)組成，每一個MH負責類似路由器的工作，並且可以接收或者傳送網路上的封包。因此Ad Hoc 無線網路在傳遞訊息還是廣播資料時，完全不需要架設固定式的基地台設備，也不會受限於地形而需佈署線路的一切成本花費。這樣的網路環境就可以克服傳統無線網路需要建置基地台的成本費用，也不會造成因無法建置基地台而無法通訊的遺憾。

Ad Hoc 的網路是一種能夠在沒有事先建置網路基礎架構的環境下，由無線行動裝置所臨時組成的網路。由於Ad Hoc 網路具有自我組織的能力，可以簡化網路的管理，提高其健全性和適應性；更能在處於動態的條件:如移動性和無法預測流量負載的既定基礎結構下有效的利用資源。Ad Hoc 無線網路的優點在於它完全不需要任何有線網路的架構或設備的支援，所以這種網路架構的應用是非常廣泛的例如:

(1) 個人或家庭區域網路:

諸如行動電話、無線耳機裝置、無線滑鼠及無線鍵盤、筆記型電腦、PDA、掌上型電腦、家裡保全系統、電燈、冰箱、微波爐、電視、洗衣機、電扇等家電，透過藍芽的元件就可以將這些設備整合在一起，而藍芽的技術也算是小型且小區域範圍的Ad Hoc 無線網路應用。

(2) 軍事上的用途:

在軍事基地的通訊上，倘若仍然依賴基地台來傳遞訊息，那是非常不可靠的，因為一旦基地台被敵方摧毀還是傳遞訊息受到干擾，都會導致通訊過程受到阻斷，因此在軍方早已利用Ad Hoc 無線網路的環境與設備來進行軍事上的運作。

(3) 視訊會議通訊:

在一些視訊會議的場合中，因應實際上的需要，行動裝置的使用者可

以不經由基地台的轉接及切換，直接與其他行動使用者利用Ad Hoc無線網路的環境參與視訊會議的討論和溝通，如此一來也可以節省通訊上的成本。

(4) 緊急救災及搜救行動:

在偏遠的山區還是比較偏僻的地方，業者基於成本上的考量，可能不易在偏遠山區架設基地台，在此情況下會導致一般通訊無法正常運作。若在緊急救災及搜救行動的過程當中利用Ad Hoc無線網路的環境進行通訊，以行動裝置跟其他人保持連絡將有助於整個搶救過程得以順利進行。

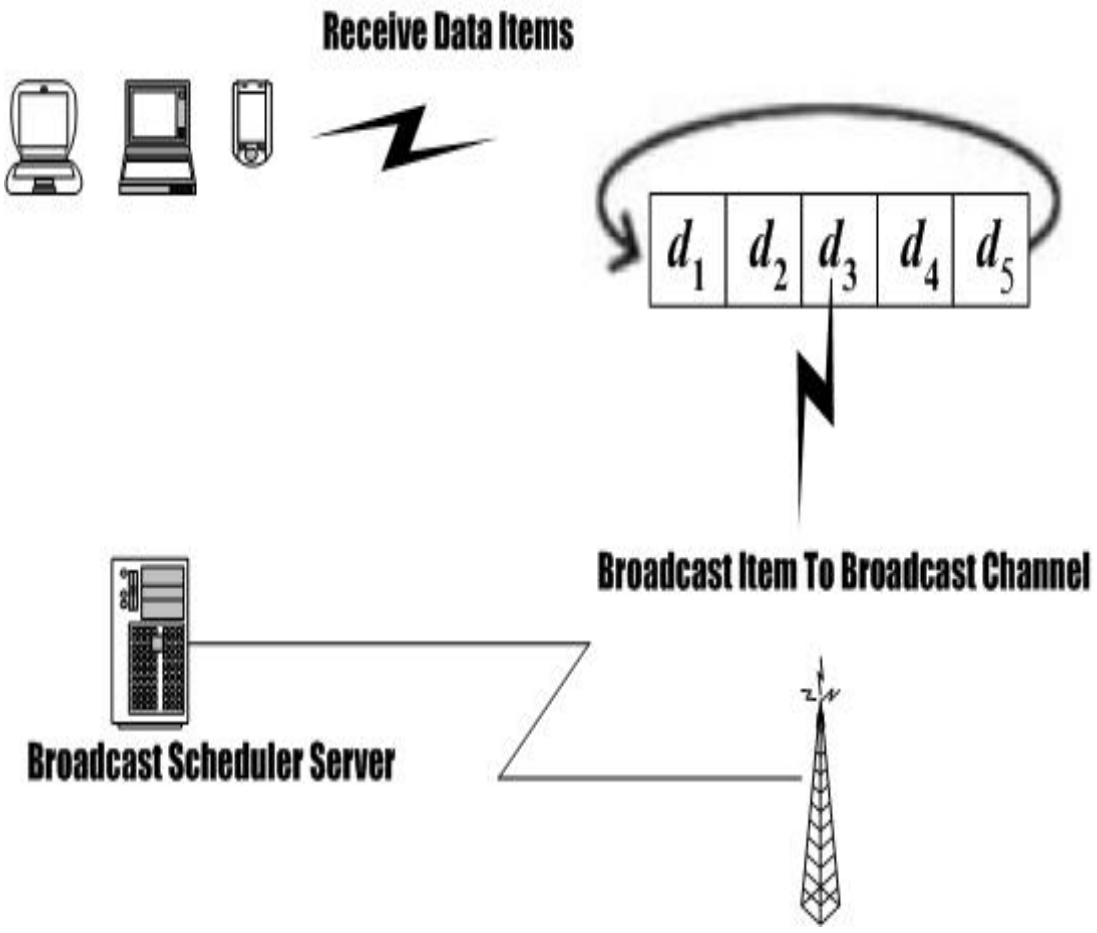

圖1-2 Client-Server的網路環境

上圖1-2 是Client-Server網路環境的架構圖，行動使用者會對伺服器提出資料的請求，而伺服器則根據用戶端請求資料的歷史記錄，這些歷史記錄可能是累積一段期間內用戶端所請求的資料，然後伺服器會根據這些資料決定一個廣播的排程，將這些資料放到廣播頻道上來廣播，有需要的行動使用者在進入廣播頻道監聽，並且接收他們所需要的資料，如此一來不同的行動使用者請求了相同的資料，只需要到廣播頻道上來接收即可，伺服器的負載量也不會太大，就可以有效率的利用無線網路的頻寬。在*Client-Server*的網路環境裡，依廣播模式來分類，又可以分成三種資料廣播模式，拉模式(pull-based model) (參見圖1-3)[3, 12, 25, 28, 30, 31]，推模式(push-based model) (參見圖1-4) [1, 2, 4, 7, 8, 9, 10, 15, 16, 17, 18, 20, 21, 22, 23, 24, 26, 32, 33]，和混合模式(hybrid model) (參見圖1-5)[14, 27]。

在下圖1-3是一種拉模式的廣播方法，又可以稱為要求式廣播(on-demand Broadcast)。拉模式是由每一個用戶端可以將資料的請求直接送到伺服器，然而在某一個時間點，某一個用戶端提出了自己所請求的資料，伺服器就會依據該請求的資料來播出，但是在這一個用戶端請求資料的同時，或者是在一段時間間隔之內，也有其他的用戶端，提出了相同或者不相同的資料請求，這時候伺服器就可能要依據請求時間的先後順序，或者是用戶端等待時間的長短，來決定整個廣播排程中資料先播及後播的問題，使得用戶端的平均等待時間能夠達到最少。這種模式有考慮到公平性的問題，也就是說當用戶端請求比較冷門的資料或者提出請求的時間點較早，這些問題在此模式都會被考慮到，不會因為等待過久而產生饑餓的情況發生，因此可以適用於當資料量變動較大的情況，也就是存取頻率變動較大的情形。

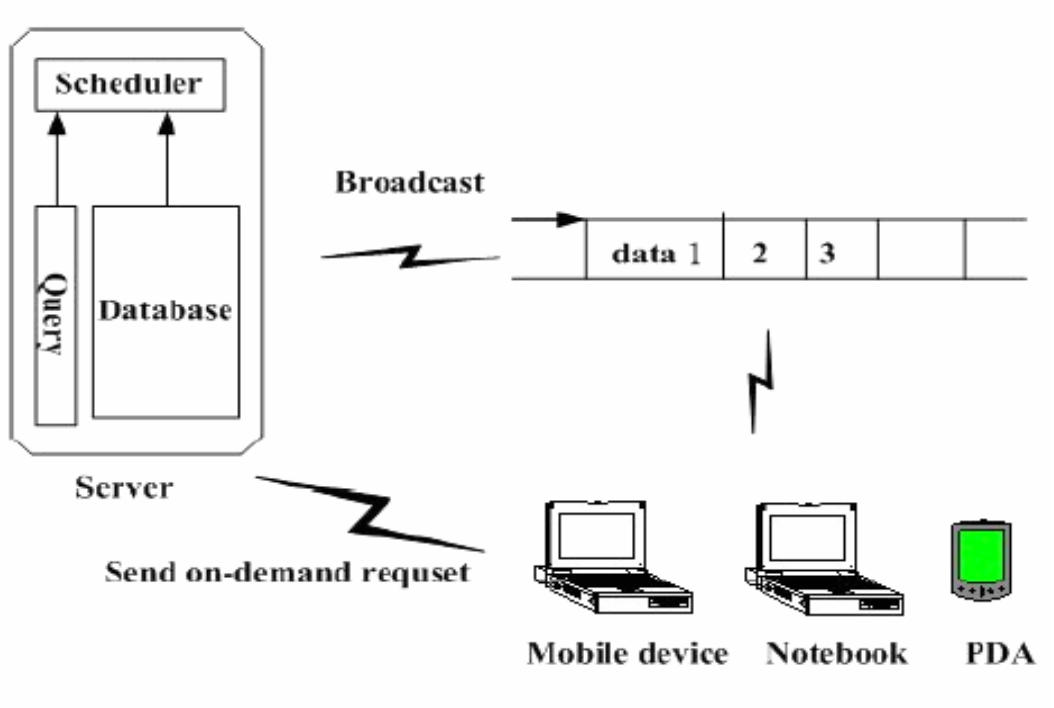

圖1-3 廣播環境中的拉模式

在拉式模式下，有幾個比較有名的演算法如下，Earliest First Request(EFR)[3]，這一個演算法是依據比較公平的方式，將用戶端先請求的資料優先播放，是根據請求的先後順序來播放，但是這樣一來造成的缺點，就是用戶端的平均等待時間會比較差一點。Most Request First(MRF)[3]，伺服器會去統計出用戶端請求最多次的資料優先播出，但是這種方法所產生的缺點，較少用戶端請求的冷門資料可能會一直無法被廣播，導致某一些用戶端可能會一直等待不到他們所請求的資料。Longest Wait First(LWF)[3]，這個方法是伺服器會去記錄所有用戶端請求資料的時間，並且找出被用戶端請求最久的資料，而這些資料用戶端一直未接收到優先播出，但是伺服器要隨時去計算每筆資料被等待的時間，將會造成伺服器的負擔會比較大一點。Request times Wait(R×W)[3]，這一個方法同時考慮到用戶端對資料的請求時間，以及資料被請求的熱門程度，並且從中找尋出一個最適當的平衡點來廣播資料。

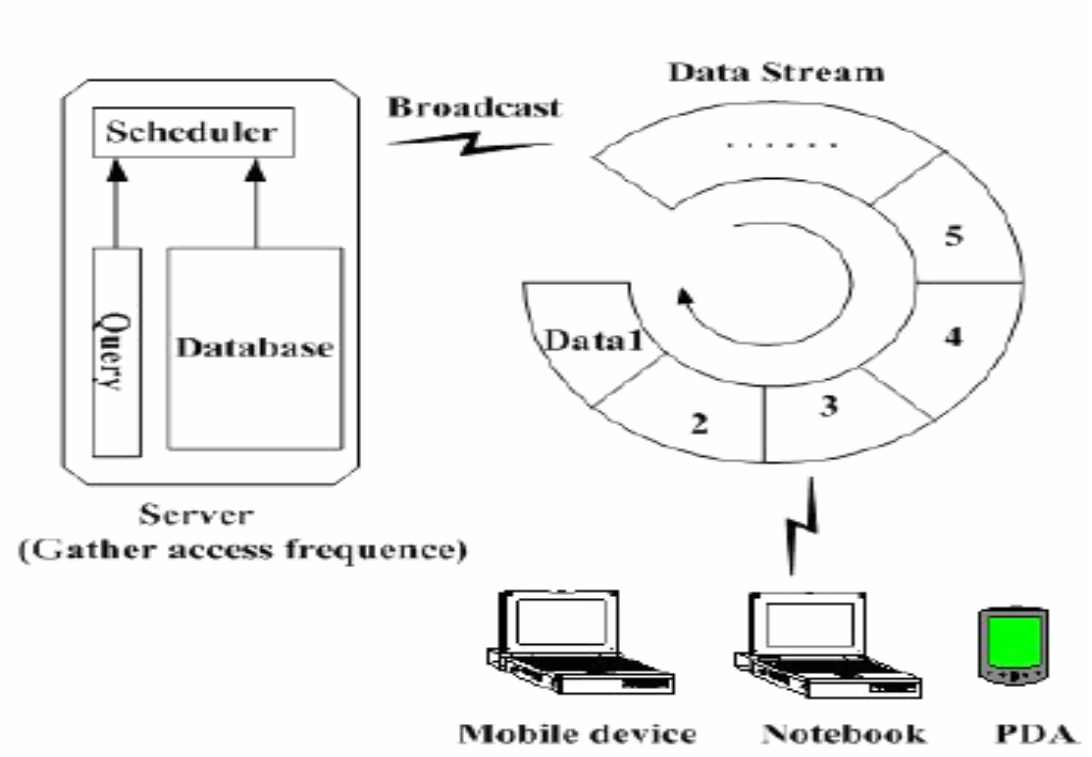

圖1-4 廣播環境中的推模式

在圖1-4是廣播環境中的推模式，此一模式就是伺服器先將用戶端所請求的資料集合全部收集，因此用戶端請求的資料已經是伺服器內的歷史記錄，伺服器會將這些歷史記錄的資料經過統計，譬如找出經常被用戶端請求的熱門資料，將所有資料組成一個資料排程，並且在一段固定的時間內，把這些資料放在廣播頻道上循環廣播，使得所有用戶端所請求的資料，都可以進到廣播頻道上來接收他們所需要的資料。然而有些情況為了熱門資料容易被用戶端來接收，會針對整個廣播排程裡頭的資料做一些不同的調整，而對於比較冷門的資料，可能在等待時間上的重視程度會比較低。這個模式下伺服器可以只廣播一份資料，而用戶端可以隨時進入廣播頻道上接收所需要的資料，用戶端進入廣播頻道上的時間點也可能會不同，但是這些被廣播的資料是一直由伺服器循環廣播的，所以當用戶端進入廣播頻道時，所需要的資料已經錯過的時候，只要等到下一個廣播週期就可以接收所需要的資料了。此模式的主要好處

是所有用戶端能夠存取資料，在同一時間的廣播頻道上，而不會增加伺服器的工作量及額外的負擔。

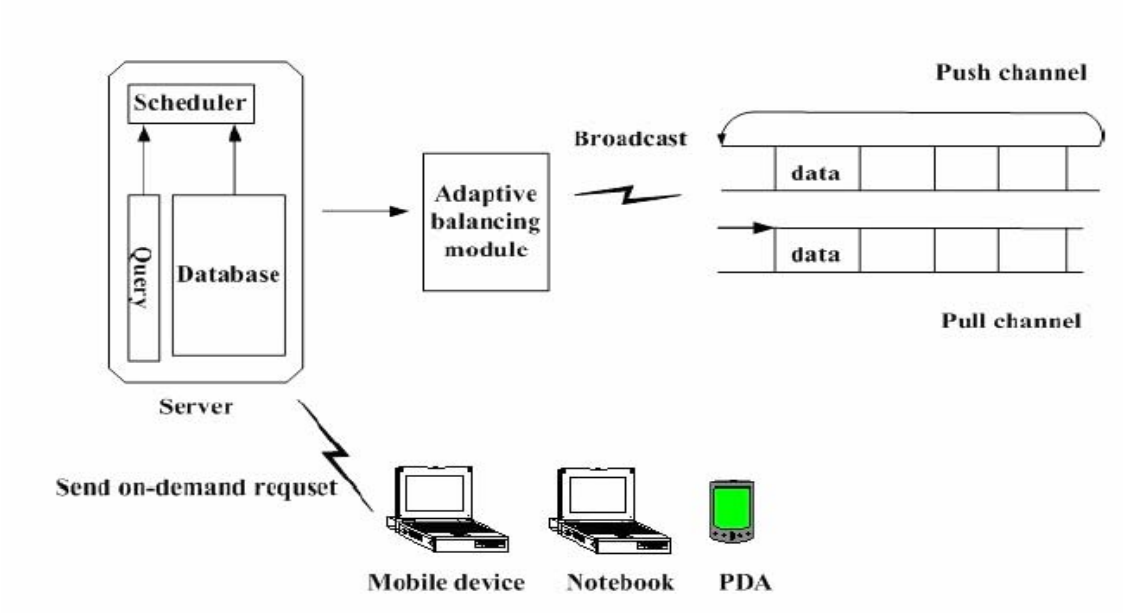

圖1-5 廣播環境中的混合模式

在圖1-5則是拉模式和推模式並用的混合模式。利用混合模式的廣播方法，可以有效率的互補拉模式及推模式的缺點，如果只使用拉模式的話，當伺服器遇到大量的用戶端請求資料時，會導致伺服器在單位時間內無法馬上決定要廣播的資料內容，使得一部分廣播頻道上的時間槽(time slot)是空的未被加以運用，因此而產生頻寬未完全被利用。如果僅使用推模式的話，當一些用戶端請求的是比較冷門的資料，因為這些資料在資料庫的歷史記錄中出現率不高，會被伺服器的廣播排程來廣播的機會相對的降低很多，甚至於可能沒被伺服器排入廣播排程當中，這樣一來請求冷門資料的用戶端就無法等到資料，必須經由on-demand的廣播頻道才能夠接收冷門的資料。因此使用混合模式的廣播，更能夠有效率的滿足提出各種請求的用戶端。在[14]，提出一個adaptive balanced scheme (ABS)的資料廣播混合模式在多頻道的廣播環境。伺服器會將比較熱門的資料放在push頻道上週期性的循環廣播，而比較冷門的資料則放到pull

頻道上，經由用戶端的請求來廣播。在[27]，提出一個混合排程的演算法，將比較熱門資料歸類成push data，而比較冷門的資料歸類成pull data，此方法預期達到用戶端的等待時間能夠達到最少。

在推模式的廣播方式又可以分成*real time*[2, 22, 24]和*non-real time*[1, 4, 7, 8, 9, 10, 15, 16, 17, 18, 20, 21, 23, 26, 32, 33]。*real time*是在即時的環境下來廣播資料，最常被應用在隨選視訊網路影音的即時廣播環境，還有線上即時電視節目的播出，由即時播放的頻道來提供廣播的服務。由於一些議題在探討平均等待的時間過長，以及大量的請求卻無法即時處理的問題，因此有些研究著重在平均等待時間的縮短[24]，縮短平均等待時間才能夠讓即時反應(*real time*)的環境可以建立。為了降低平均等待時間，很多處理方式是將比較熱門的資料多次且循環廣播，這樣可以服務到大多數用戶端的需求，但是產生的問題則是讓一些請求冷門資料的用戶端無法等到他們請求的資料，導致可能變成饑餓（Starvation）的情況發生。在[24]，提出一個有效率的演算法在*real time*的廣播排程上，演算法的目的在使得平均存取時間達到最小化。

在*non-real time*中，又有兩個重要的議題，其中第一個議題是考慮行動裝置電源消耗的問題，因為行動裝置的電源無法保持連續供應及不容易隨時取得，所以如何節省電源的消耗是相當重要的問題，然而做好資料的排程可能會對節省電源方面有些作用，用戶端可能因為比較好的廣播排程，使得等待的時間因此而縮短，進而達到節省電源的目的。為了節省電源消耗，首先要對廣播的資料利用索引(*Index*)技術，來記錄資料廣播的資訊在廣播頻道上。因為先對要廣播的資料做索引之後，這些索引便可以記錄資料在多久之後會被伺服器來廣播，這時候當用戶端進入廣播頻道之後，只要先接收到索引資訊，就可以知道所請求的資料在多久之後的時間點被廣播，然後用戶端就可以進入休眠模式(*doze mode*)，

可以不用一直在廣播頻道上監聽資料，所以用戶端進入休眠模式損耗的電源會比活動模式(*active mode*)還要來的低，等到所請求的資料要出現之前的時間點恢復成活動模式，而用戶端在活動模式所花的時間就稱為*Turning Time*，並且接收用戶端所需要的資料即可，不用每個時間點都去監聽是否該筆資料是用戶端所請求的，這樣一來就可以節省了電源的浪費，下圖1-6會進一步詳細說明。

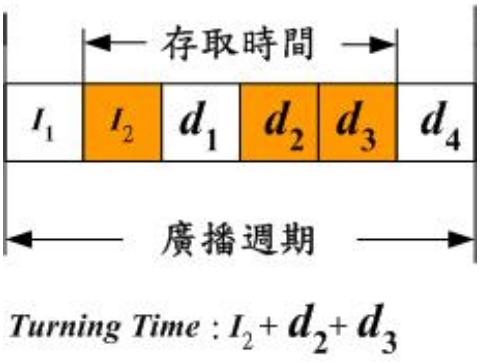

圖 1-6 說明 Access Time 和 Turning Time

在圖1-6中，I_1和 I_2個代表廣播排程的索引資訊。假設用戶端要求資料d_2和d_3，其中從I_1到d_4是伺服器的一個廣播週期。*Turning Time*代表的意思是用戶端從I_2的時間點進入到廣播頻道上，讀取索引資訊I_2之後，知道所要求的資料d_2和d_3在何時會被伺服器來廣播，然後就會進入休眠模式，等到資料d_2和d_3出現的時候，才進入活動模式將資料d_2和d_3接收下來；因此*Turning Time* 包括讀取I_2的時間加上接收資料d_2和d_3的時間。有一些研究在解決*Turning Time*，利用索引技術在資料廣播上[7, 15, 23, 33]，此方法主要以減少電源的消耗為目的。另外一個議題是用戶端存取時間的問題，我們簡單定義存取時間(*Access Time*)，當用戶端進入廣播之後，直到所要求的全部資料都接收完畢的整段時間，我們就稱為存取時間。我們

以上圖1-6來說明，假設用戶端從I_2的時間點進到廣播頻道上，而該用戶端所請求的資料是資料d_2和d_3，從I_2的時間點開始計算直到接收完資料d_2和d_3的整段時間，我們就稱之為存取時間。在很多篇研究中提出解決存取時間的問題[1, 4, 8, 9, 10, 16, 17, 20, 21 , 26, 32]，期望達到存取時間愈短愈好。在[1][26]把資料配置在伺服器的磁碟上，他們將比較熱門的資料表示成比較高的存取頻率，放在速度比較快的磁碟機上面；比較冷門的資料表示成比較低的存取頻率，放在速度比較慢的磁碟機上面。他們想要有效率的改善資料存取時間在資料廣播的環境。有關資料存取時間的議題，又可以把用戶端請求分成單一資料[4][17]和多重資料[9, 10, 16, 20, 21]。在[4][17]中，作者提出的演算法主要在解決用戶端請求單一資料，期望達到平均存取時間最小化，而且他們的方法都有達到最佳化的目標，這兩篇研究都是在多頻道的廣播環境。用戶端請求多重資料又可細分成單一頻道[9, 10, 20, 21]和多頻道[16]。在[21]文章中，他們提出幾個不同Data-Oriented 的方法，將query set中的資料分別拆開來看，對每一個用戶端請求的資料累計其各自的頻率，並將較高的與次高的放在一起，最後使得最常被要求的資料擺放的愈集中，而更容易在較短的時間內，滿足較多用戶端的要求。在[9]，他們有效率的先將資料分群之後，再分配到單一頻道上。在[10]，作者改良了之前被提出的方法，再加上資料探勘的技術，使得存取時間能更有效率的減少。在[16]作者們利用基因演算法，從用戶端請求的多重資料，產生一個最佳解的適應性函數，使得平均存取時間達到最小。

第二節 本篇論文的目標及架構

在本文中我們使用Kernighan-Lin 演算法(簡稱為 KL 演算法)[18]，來進行廣播頻道的分割，將用戶端請求的多重資料平均分割在多個廣播頻道上。在本篇探討的環境上，是把用戶端請求的資料集合平均分配在

二個廣播頻道上，如此一來用戶端請求的資料集合，極有可能被分配在不同的廣播頻道上，這樣的情況我們稱為資料對抗，一旦資料被分配在不同廣播頻道上，將來伺服器端在安排廣播排程時，用戶端想要接收的資料，就有可能在相同時間點上被廣播，而這樣的情況我們稱為資料衝突。資料分割的目的，減少用戶端請求的資料集合發生資料對抗的次數，使得將來伺服器端在進行廣播排程時，資料會發生衝突的機率就會降低很多。當用戶端進入廣播頻道接收資料時，減少在相同時間點上要接收的資料發生衝突的機率，預期能夠盡量降低用戶端的資料存取時間。KL演算法一開始是先將用戶端請求的資料集合，把資料與資料之間的存取次數記錄下來，並且每一筆資料轉換成相對應的頂點，產生一個有關聯的請求資料關係圖。因此，用戶端任二個資料之間請求的次數，在圖形上便是頂點與頂點所連成的線，每條線都代表著任二筆資料的請求次數。知道用戶端請求資料的次數後，每一次都從二個廣播頻道上各挑選一個資料互相交換，而交換之後得到的結果，使得資料被分配在不同頻道的次數減少，一直不斷做資料兩兩互換的動作，直到KL 演算法的終止條件式，即判斷再怎麼交換都無法減少請求的資料被分配不同頻道的次數，馬上停止互相交換的動作並且輸出結果。

在實驗結果及效能分析，我們的方法能有效率減少用戶端請求的資料集合，發生資料對抗的次數。我們使用的KL演算法，在均勻分配時平均提升27%的效能，在常態分配時平均提升77%的效能，在指數分配時平均提升82%的效能。本文的大綱如下:第二章將針對要解決問題的核心進一步描述和說明。第三章為使用的KL演算法和舉實例說明演算法流程。第四章是本文使用方法的實驗結果和效能評估。最後一章為結論及未來展望。

第二章　問題描述

第一節 單一頻道和多頻道的差異

我們所要探討的問題，就是把每一個用戶端請求的多重資料放在多頻道上，使得整個廣播的週期縮短，首先舉兩個簡單的例子說明單一頻道與多頻道的差異(如圖 2-1 所示)。伺服器廣播資料項的集合 $D = \{d_1, d_2, d_3, d_4, d_5, d_6\}$，假設每個資料項大小都相同，存取時間均為一個時間單位，表 2-1 為本文中所定義的符號表。

表 2-1 符號定義

符　號	符號意義
d_i	要廣播的資料項
D	廣播的資料集合
$\|D\|$	廣播資料的數量
q_i	某一個用戶端提出的資料請求
Q	用戶端的集合
$DS(q_i)$	q_i 請求資料的集合
C_i	第 i 個廣播頻道　$i \in \{1,2,...,k\}$
$DS(C_i)$	被分配在 C_i 上的資料所成的集合
I_i	I_i 值代表同一個廣播頻道上的資料之間的關聯
E_i	E_i 值代表跨不同廣播頻道上的其他資料之間的關聯
h_i	E_i 減去 I_i 的值
g_{xy}	資料 d_x 與資料 d_y 互換之後得到獲利的值 其中 $d_x \in DS(C_1), d_y \in DS(C_2)$
g_i	代表第 i 次交換所對應的 g_{xy} 值

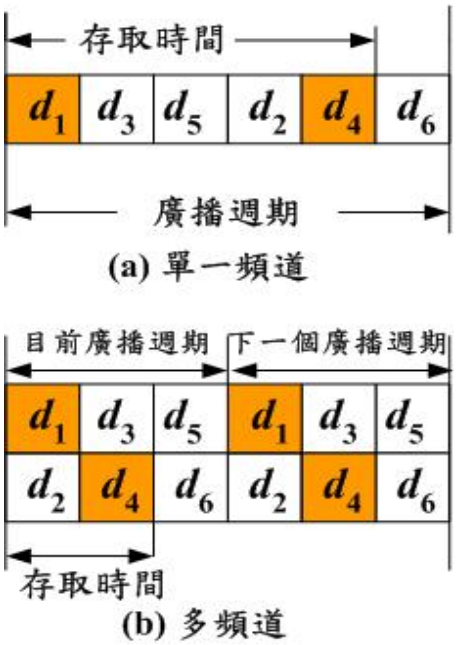

圖 2-1 單一頻道和多頻道的差異

用戶端從 d_1 的時間點進入廣播頻道上接收資料，用戶端要接收的資料是$\{d_1, d_4\}$，在圖 2-1(a)中的是單一頻道的廣播環境，這時候用戶端必須花五個時間單位的存取時間，才能夠很順利地把資料接收完。在圖 2-1(b)中的是多頻道的廣播環境， 此刻用戶端只需要兩個時間單位的存取時間，就能夠接收完所請求的資料。從這個例子我們可以看得出來，在圖 2-1(a)中，伺服器在單一頻道的廣播環境下，要被廣播的資料有 $d_1, d_2, d_3, d_4, d_5, d_6$ 這六筆，而要被廣播的資料是依照 $d_1, d_3, d_5, d_2, d_4, d_6$ 的順序來廣播，伺服器會依此順序的廣播週期來循環廣播，在單一頻道上一個廣播週期是六個時間單位；在圖 2-1(b)中，伺服器在多頻道的廣播環境下，要被廣播的資料有 $d_1, d_2, d_3, d_4, d_5, d_6$ 這六筆，然而在廣播頻道 1 上，伺服器要廣播資料 $d_1, d_3,$ 和 d_5，並且按照 d_1, d_3, d_5 的廣播順序來廣播資料，然後依據這樣的方式週期性的廣播；而在廣播頻道 2 上伺服器要廣播資料 d_2, d_4 和 d_6，並且按照 d_2, d_4, d_6 的廣播順序來廣播資料，然後依據這樣的方式週期性的廣播，跟圖 2-1 (a)中的單一頻道的廣播環境比較之下，整

個廣播的週期縮短了一半，因此當廣播頻道的數量愈多，將會使得整個廣播頻道的時間週期愈短，就可以讓用戶端縮短資料的存取時間。在圖2-1(b)的多頻道廣播環境中，由於廣播週期縮短使得每一筆資料出現的次數愈頻繁，用戶端能夠在比較短的時間內接收所需要的資料，在資料的存取時間上可以相對地減少。

第二節 多頻道資料衝突

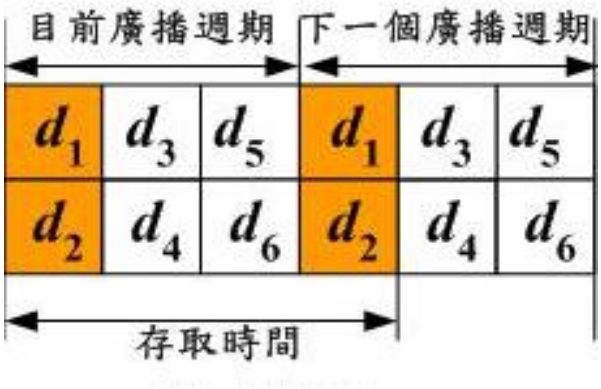

圖 2-2 多頻道發生資料衝突

在多頻道的廣播排程中，可能會發生的問題就是資料衝突，所謂資料衝突就是用戶端所請求的資料，被伺服器在同一個時間點來廣播，我們舉一個例子來說明發生資料衝突的情況(如圖 2-2 所示)。在圖 2-2 的例子中的是多頻道的廣播環境，假設用戶端從 d_1 的時間點進入廣播頻道上接收資料，用戶端要接收的資料是$\{d_1, d_2\}$，要被廣播的資料有 $d_1, d_2, d_3, d_4, d_5, d_6$ 這六筆，然而在廣播頻道 1 上，伺服器要廣播資料 $d_1, d_3,$ 和 d_5，並且按照 d_1, d_3, d_5 的廣播順序來廣播資料，然後依據這樣的方式週期性的廣播；而在廣播頻道 2 上伺服器要廣播資料 d_2, d_4 和 d_6，並且按照 d_2, d_4, d_6 的廣播順序來廣播資料，然後依據這樣的方式週期性的廣播，由於用戶端所請求的資料是 d_1 和 d_2，發生了被廣播的資料在同一時間點上的資料衝突，所以用戶端只能夠在同一時間點上接收 d_1 或 d_2 其中一筆資料，必須等到下一個廣播週期才能接收完所需要的資料，因此在廣播排程上一

旦發生資料衝突的情形，勢必使得資料的存取時間變得更長了。

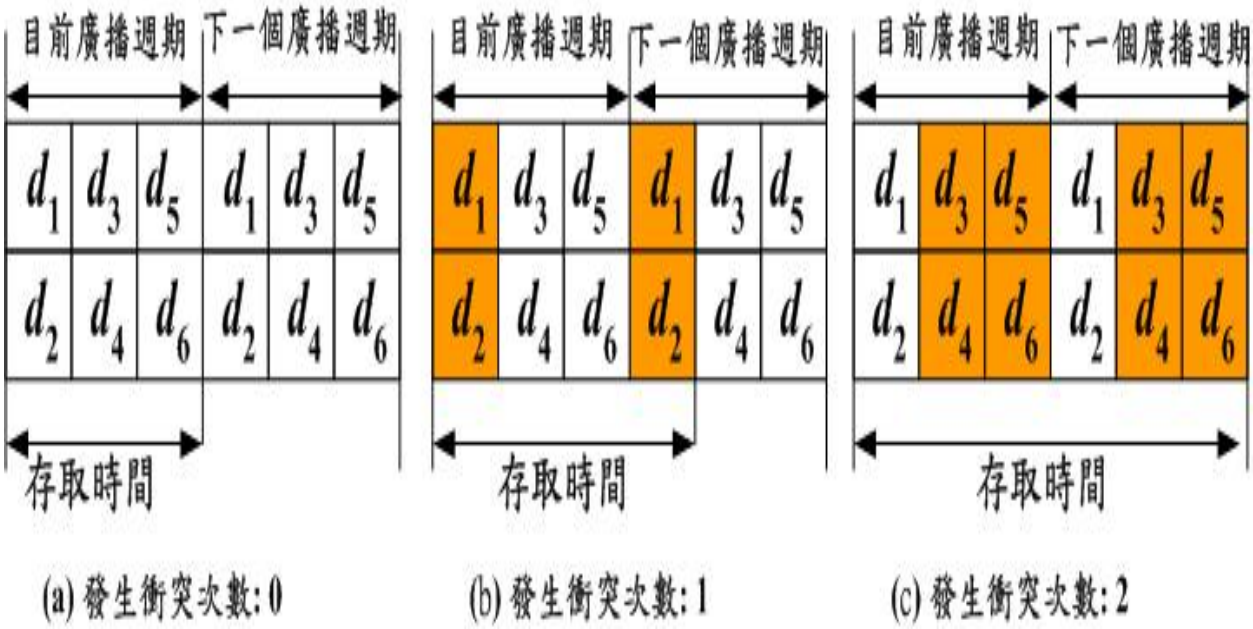

圖 2-3 資料衝突所發生的次數

我們再舉一個例子說明用戶端所請求的資料集合，如果發生資料衝突的情況對資料存取時間的影響。假設 $D = \{d_1, d_2, d_3, d_4, d_5, d_6\}$，並且將資料分割到 k 個廣播頻道上(在這個例子，$k = 2$ 如圖 2-3 所示)。當用戶端請求資料集合如圖 2-3(a)所示，$DS(q_i)=\{d_1, d_3, d_5\}$，在圖 2-3(a)中，有二個廣播頻道，廣播頻道 1 中伺服器要廣播資料 $d_1, d_3,$ 和 d_5，並且按照 d_1, d_3, d_5 的廣播順序來廣播資料，然後依據這樣的方式週期性的廣播；而在廣播頻道 2 中伺服器要廣播資料 d_2, d_4 和 d_6，並且按照 d_2, d_4, d_6 的廣播順序來廣播資料，然後依據這樣的方式週期性的廣播，假設一個用戶端提出請求的資料是 $d_1, d_3,$ 和 d_5，這些資料都被分配在同一個頻道上，因此沒有發生資料對抗的情形，當他在 d_1 這個時間點上進入廣播頻道上，就可以非常順利的花三個時間單位，接收完他所需要的資料，並且在接收資料的過程也沒有發生資料衝突的情況。

當用戶端請求資料集合如圖 2-3(b)所示 $DS(q_2)=\{d_1, d_2, d_4, d_6\}$，在圖 2-3(b)中，有二個廣播頻道，廣播頻道 1 中伺服器要廣播資料 d_1, d_3, 和 d_5，並且按照 d_1, d_3, d_5 的廣播順序來廣播資料，然後依據這樣的方式週期性的廣播；而在廣播頻道 2 中伺服器要廣播資料 d_2, d_4 和 d_6，並且按照 d_2, d_4, d_6 的廣播順序來廣播資料，然後依據這樣的方式週期性的廣播，假設一個用戶端提出請求的資料是 d_1, d_2, d_4 和 d_6，當他在 d_1 這個時間點上進入廣播頻道上，在同一個時間點上同時廣播資料 d_1 和 d_2，而這兩筆資料同時都是用戶端所提出請求的，此刻就發生了資料的衝突，用戶端只能接收 d_1 或者 d_2 其中一筆資料，然後用戶端可以順序地接收 d_4 和 d_6 這兩筆資料，接著用戶端要等到下一個廣播週期方能接收完所有的資料，因為發生了一次存取資料的衝突，導致用戶端無法在一個廣播週期的時間接收完所有的資料，使得用戶端在資料接收的時間拉長，也就是增加了用戶端的存取時間。

當用戶端請求資料集合如圖 2-3(c)所示 $DS(q_3)=\{ d_3, d_4, d_5, d_6\}$，在圖 2-3(c)中，有二個廣播頻道，廣播頻道 1 中伺服器要廣播資料 $d_1, d_3,$ 和 d_5，並且按照 d_1, d_3, d_5 的廣播順序來廣播資料，然後依據這樣的方式週期性的廣播；而在廣播頻道 2 中伺服器要廣播資料 d_2, d_4 和 d_6，並且按照 d_2, d_4, d_6 的廣播順序來廣播資料，然後依據這樣的方式週期性的廣播，假設一個用戶端提出請求的資料是 d_3, d_4, d_5 和 d_6，當他在 d_1 這個時間點上進入廣播頻道上，在下一個時間點上同時廣播資料 d_3 和 d_4，而這兩筆資料同時都是用戶端所提出請求的，此刻就發生了資料的衝突，用戶端只能接收 d_3 或者 d_4 其中一筆資料；在下一個時間點上同時廣播資料 d_5 和 d_6，而這兩筆資料同時又剛好是用戶端所提出請求的，此刻又發生了一次資料上的衝突，用戶端只能接收 d_5 或者 d_6 其中一筆資料，因為發生了二次存取資料的衝突，導致用戶端無法在一個廣播週期的時間接收完所有的資

料，用戶端必須要等到下一個廣播週期方能接收完所有的資料，使得用戶端的存取時間又變得更長了。

第三節 資料衝突與資料對抗的關係

在多頻道的資料排程方面，我們可以把它分成二個部分來做，第一個部分先決定用戶端所請求的資料集合，將它們分配在那一個頻道上的問題，這個部分做完以後再進行第二個部分的實際資料排程，在本篇論文我們所做的只是第一個部分，而沒有做第二個部分的實際資料排程。我們所做的第一個部分，以一個簡單的例子來說明之。

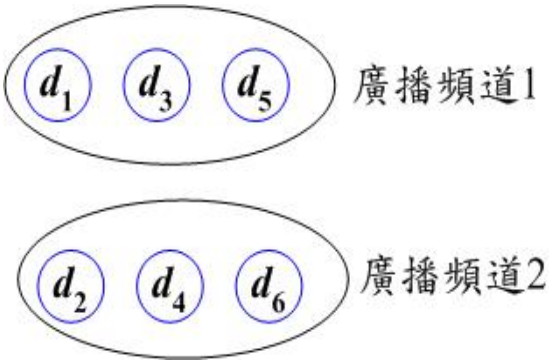

圖 2-4 發生資料對抗的情況

在圖 2-4 中，就是我們要做的第一個部分，假設 $D = \{d_1, d_2, d_3, d_4, d_5, d_6\}$，並且將資料分割到 2 個廣播頻道上，其中我們把資料 $d_1, d_3,$ 和 d_5 放在廣播頻道 1 上，另外的資料 d_2, d_4 和 d_6 放在廣播頻道 2 上，因為沒有做第二部分的實際資料排程，所以廣播頻道上的資料是沒有前後的順序。假設用戶端同時所請求的資料集合，某一些資料放在廣播頻道 1，而另外的某一些資料被分配在廣播頻道 2，我們把這樣的情況定義成資料對抗。我們所做的第一個部分的目的，是為了第二個部分來著想，因為把用戶端請求的資料集合放在多頻道上，就會產生資料對抗的問題，將來在做第二個部分的實際資料排程時，就可能會發生同一個時間點上資料衝突的問

題，因此如果把用戶端請求的資料集合，盡量放在同一個廣播頻道上的話，也就是說如果沒有產生資料對抗的問題，將來在做實際資料排程時，就不會發生資料衝突。

我們要解決的問題，就是把所有的$d_i \in D$，平均分配到k個頻道$(k \geq 2)$，使得用戶端所請求的資料集合，減少發生資料對抗的次數，將來伺服器在進行實際廣播資料排程的時候，被廣播的資料在同一個時間點上會發生資料衝突的機率，能夠盡可能降低。

第三章 KL 演算法

第一節 KL 演算法定義

在本段中我們討論如何將資料分割，並且把用戶端請求的資料平均分配到多頻道上，期望能夠有效率的降低資料對抗的次數，將來伺服器在進行廣播排程時，使得資料在同一時間點上被廣播而發生資料衝突的機率降低，讓用戶端在資料的存取時間盡量減少的目的。$DS(q_i)$表示某一個用戶端請求資料項的集合，如果將$DS(q_i)$轉換成一個圖形的觀念，每一個資料項建立成圖形上的每一個頂點。對某一個資料$d_i \in DS(q_i)$, $q_i \in Q$，我們就建立一個v_i來對應d_i，我們用 V 來代表所有的頂點所成的集合。對於任意兩個頂點$v_x, v_y \in V$，頂點v_x, v_y對應到$d_x, d_y \in DS(q_i)$, $q_i \in Q$，如果被用戶端提出請求，我們就必須建立一條e_{xy}。每一條e_{xy}代表有多少個用戶端同時請求d_x和d_y這兩筆資料，而W_{xy}表示用戶端請求資料d_x和d_y的存取次數。所以把用戶端請求的資料集合處理完畢之後，轉換成一張用戶端請求資料的關聯圖。

我們的演算法，一開始把用戶端請求的資料，先隨機將一半的資料放在廣播頻道 1；另外一半的資料放在廣播頻道 2。然後從廣播頻道 1 中挑選特定資料；同時也從廣播頻道 2 挑選特定資料，互換之後得到獲利的值定義為g_{xy}，其中g_{xy}代表對應的$\widehat{g_i}$值(i=1，表示第一次互換；其他以此類推)。在計算獲利值g_{xy}之前，先分別把廣播頻道上的每一個頂點，計算內部線關係值(以符號I_i表示)，I_i值代表同一個廣播頻道上的頂點之間的關聯；計算外部線關係值(以符號 E_i表示)，E_i值代表跨不同廣播頻道上的其他頂點之間的關聯；然後外部線關係值減去內部線關係值(以符號h_i表示)。h_i代表的意義是廣播頻道上某一個頂點v_i，搬到另外一個頻道上的話，將會獲利的值，也就是只考慮單一頂點的移動。而計算g_{xy}值則是同時考慮任兩個頂點v_x, v_y，從不同的頻道上互換會產生的獲利值。

接下來把廣播頻道 1 和廣播頻道 2 的所有資料，每兩個資料配對的組合在交換之後，所產生的獲利值在計算完以後記錄下來，要交換的順序從獲利值最大的，也就是哪一對交換賺最多獲利值的先交換，交換完之後將那兩個資料鎖定，其他的資料在更新完獲利值之後，按照這樣的順序來交換。第一次全部交換完之後，所有的資料都鎖定住了，就可以計算這一次交換獲利最多的值。把第一次獲利最多的值，連同會獲利的那些資料記錄起來，然後解除所有資料的鎖定，並且將會獲利的那些資料做真正互換；互換以後，二個廣播頻道上會產生新的資料順序，再重新計算得到新的獲利最多的值。再進行第二次資料真正交換，以此類推一直做交換。每一次都是根據資料交換之後，會得到最大的獲利值(Maximum partial sum)，當最大的獲利值不是 0 的時候，演算法就會一直做交換，並且把每一次交換得到最大的獲利值累加，累加到最大的獲利值為 0(見下面公式 1)。當最大的獲利值為 0，即再分割和交換廣播頻道上資料都無法獲利，因此演算法就停止並輸出結果，產生較佳的資料分割在多頻道上。

$$\text{最大的獲利值} = \left(\max \sum_{i=1}^{k} \widehat{g_i}\right) = 0, 1 \le k \le \frac{|D|}{2} \qquad (1)$$

第二節 KL 演算法流程

我們以簡單的範例來說明整個演算法運算過程。假設有 3 個用戶端請求資料如下:$q_1=\{d_3, d_4, d_5\}$, $q_2=\{d_2, d_4, d_5\}$, $q_3=\{d_1, d_2, d_6\}$，$D=\{d_1, d_2, d_3, d_4, d_5, d_6\}$，轉換成圖形上的頂點，$V=\{v_1, v_2, v_3, v_4, v_5, v_6\}$，例如 $d_4, d_5 \in DS(q_1)$，同時 $d_4, d_5 \in DS(q_2)$，故資料 d_4 和 d_5 就會建立一條 $e_{4,5}$ 的線，$e_{4,5}$ 就代表有二個用戶端請求資料 d_4 和 d_5，而用戶端請求資料 d_4 和 d_5 的權重值 $w_{4,5}=2$，在把用戶端請求的資料集合處理完畢之後，就轉換成用戶端請求資料的關聯圖(如圖 3-1 所示)。

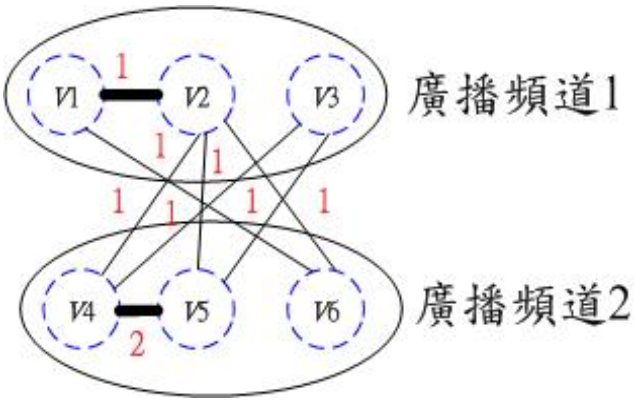

圖 3-1 用戶端請求資料的關聯圖

在圖 3-1，頂點{v_1,v_2,v_3,v_4,v_5,v_6}是用戶端請求資料對應圖形上的頂點，虛線表示頂點還沒有被交換過，資料是未被鎖定的狀態，任意兩個頂點形成的線，在同一個廣播頻道上我們稱為內部線關係(以粗線條表示)；跨不同廣播頻道形成的線，我們稱為外部線關係(以細線條表示)。為了要計算所有配對的g_{xy}值，所以要先計算下表 3-1 的I_i, E_i和h_i的值，就可以算出每兩個配對的g_{xy}值。我們先將所有的資料分在二個廣播頻道上，$DS(C_1)$= {v_1,v_2,v_3}；$DS(C_2)$= {v_4,v_5,v_6}。分別計算每一個頂點的內部線關係值(以符號I_i表示)，以及外部線關係值(以符號E_i表示)，外部線關係值減去內部線關係值(以符號h_i表示)，計算的結果如表 3-1:

表 3-1 計算頂點 v_1, v_2, v_3, v_4, v_5, v_6 之間的關係值

值 / 頂點	I_i	E_i	h_i
v_1	1	1	0
v_2	1	3	2
v_3	0	2	2
v_4	2	2	0

v_5	2	2	0
v_6	0	2	2

表3-1中，以頂點v_4做簡單的說明。頂點v_4在廣播頻道2，跟頂點v_5的權重值為2，故內部線關係值I_4=2。頂點v_4跨廣播頻道1，跟頂點v_2和v_3各有權重值為1，故外部線關係值E_4=2，而h_4=E_4 – I_4=0。h_4所代表的意義，當頂點v_4從廣播頻道2搬到廣播頻道1，會獲利的值(以單一頂點的觀點來看)，表示當頂點v_4從廣播頻道2搬到廣播頻道1，會損失跟頂點v_5的權重值2，但是換到廣播頻道1的時候，會獲利頂點v_2和v_3的權重值2，所以獲利的權重值一增一減之下變成0。

現在要將廣播頻道1上的其中一個資料項，$d_x \in DS(C_1), DS(C_1)= \{v_1, v_2, v_3\}$；和廣播頻道2上的其中一個資料項，$d_y \in DS(C_2), DS(C_2)= \{v_4, v_5, v_6\}$，互相做交換的動作，我們定義下面的公式2:

$$g_{xy} = h_x + h_y - 2w_{xy} \quad (2)$$

g_{xy} 表示當我們把頂點 v_x 從廣播頻道 1 搬到廣播頻道 2；同時把頂點 v_y 從廣播頻道 2 搬到廣播頻道 1，我們可以得到的獲利值。w_{xy} 表示頂點 v_x 和 v_y 之間的權重值，h_i 值是以單一頂點的觀點計算(計算頂點 v_x 得到 h_x，計算頂點 v_y 得到 h_y)，現在是二個頂點互換，計算的結果如表 3-2，我們以下表 3-2 的 g_{26} 來說明公式 2。當頂點 v_2 和 v_6 互換，在表 3-1 的 h_2 值已經有計算過和頂點 v_6 的權重值；在表 3-1 的 h_6 值也已經有計算過和頂點 v_2 的權重值，所以要減掉 2 倍的 w_{26} 值，才是 g_{26} 交換獲利的值。

表 3-2 計算頂點 $v_1, v_2, v_3, v_4, v_5, v_6$ 互換的 g_{xy} 值

g_{14}	g_{15}	g_{16}	g_{24}	g_{25}	g_{26}	g_{34}	g_{35}	g_{36}
0	0	0	0	0	2	0	0	4

根據表 3-2，把每 2 個頂點互相交換計算出來的結果；當頂點 v_3 和 v_6 互換，得到最大的 g_{36} 值 4，我們就把獲利值最大的頂點 v_3 和 v_6 互換；其中 $\widehat{g_1}$ 代表對應的 g_{36} 值。我們第一次把 v_3 和 v_6 交換，把它表示成 $(\widehat{g_1} = 4)$。把頂點 v_3 和 v_6 互換之後，就暫時將頂點 v_3 和 v_6 鎖定，表示暫時不會再更動 v_3 和 v_6，鎖定的頂點以實線表示 (見圖 3-2)。

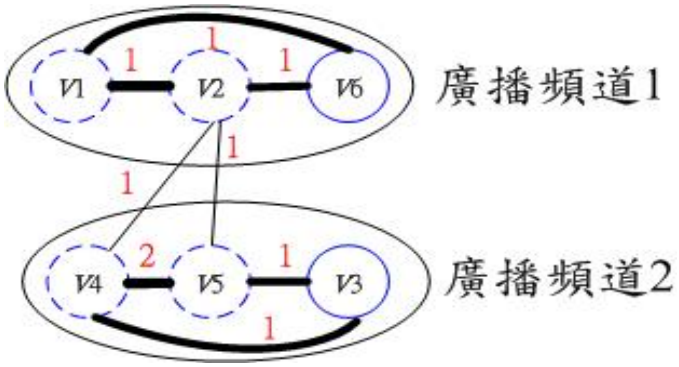

圖 3-2 模擬交換頂點 v_3 和 v_6 的結果

把頂點 v_3 和 v_6 互換之後，並且將頂點 v_3 和 v_6 變成暫時鎖定的狀態。因為頂點 v_3 和 v_6 互換之後，頂點之間的關聯因此改變，我們再繼續重新計算內部線關係值和外部線關係值如表 3-3:

表 3-3 重新計算頂點 v_1, v_2, v_4, v_5 之間的關係值

值 / 頂點	I_i	E_i	h_i
v_1	2	0	-2
v_2	2	2	0
v_4	3	1	-2
v_5	3	1	-2

計算完表 3-3 這些頂點的值，我們計算頂點互換的 g_{xy} 值，如下表 3-4:

表 3-4 重新計算頂點 v_1, v_2, v_4, v_5 互換的 g_{xy} 值

g_{14}	g_{15}	g_{24}	g_{25}
-4	-4	-4	-4

從表 3-4 計算的結果，獲利的 g_{xy} 值都是負數的。表 3-4 中所有頂點互相交換，都會獲得相同最大的值-4。因此任意兩個頂點互換都可以，我們就選擇把頂點 v_1 和 v_4 互換，把它表示成 $(\widehat{g_2} = -4)$。把頂點 v_1 和 v_4 互換之後，就暫時將頂點 v_1 和 v_4 鎖定，表示暫時不會再更動頂點 v_1 和 v_4 (見圖 3-3)。

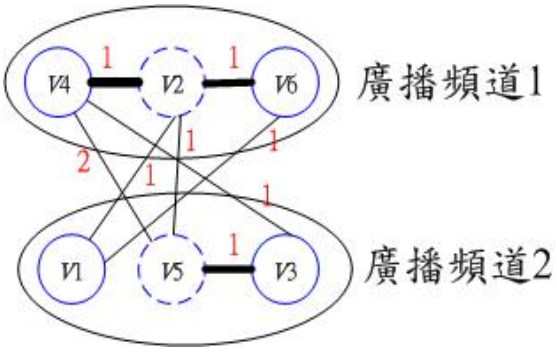

圖 3-3 模擬交換頂點 v_1 和 v_4 的結果

我們再重新計算頂點 v_2 和 v_5 互換得到的值，並交換的結果如下圖 3-4:

$g_{25} = 0$ $(\widehat{g_3} = 0)$

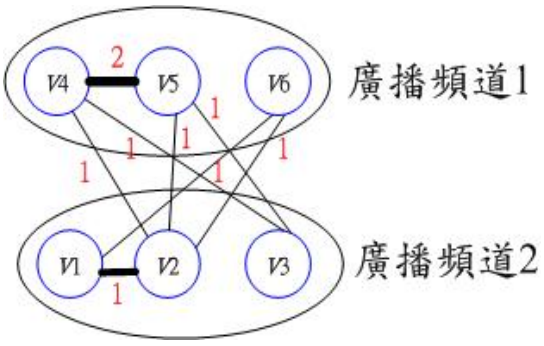

圖 3-4 模擬交換頂點 v_2 和 v_5 的結果

到目前這個步驟，我們已經將所有的頂點從廣播頻道 1 換到廣播頻道 2。但是實際上，前面這幾個互換的動作，只是模擬資料互相交換，目的就是要明確知道交換那些頂點，會使得獲利的值達到最大；然而並沒

有真正做交換的動作。在這一個例子中，我們得到一個結論，第一次交換頂點 v_3 和 v_6 得到 $\widehat{g_1} = g_{36} = 4$；第二次交換頂點 v_1 和 v_4 得到 $\widehat{g_2} = g_{14} = -4$；第三次交換頂點 v_2 和 v_5 得到 $\widehat{g_3} = g_{25} = 0$。

最大的獲利值= $(\max \sum_{i=1}^{k} \widehat{g_i}) = 4 \ (k = 1)$

上面這個式子是在計算用戶端請求的所有資料，在廣播頻道交換會得到獲利最大的值，也就是減少資料對抗的次數；用戶端跨不同廣播週期的次數愈少，就可以盡量降低用戶端的存取時間。以這個例子來說，廣播頻道上的頂點在第一次互換得到最大的獲利值是 4。從圖 3-1 的關聯圖，我們看到跨不同廣播頻道的有 6 單位的權重值。在圖 3-2 交換頂點 v_3 和 v_6 之後，跨不同廣播頻道的只剩下 2 單位的權重值，減少了 4 單位的權重值也就是最大的獲利值。換言之，從原本發生資料對抗次數的 6 次，降低成剩下 2 次發生資料對抗的次數。

根據模擬廣播頻道資料交換的結果，我們得知先把頂點 v_3 和 v_6 互換，會得到最大的獲利值。然後將模擬過程交換時，把暫時鎖定的所有資料解除鎖定，把頂點 v_3 和 v_6 互換的結果如圖 3-5。

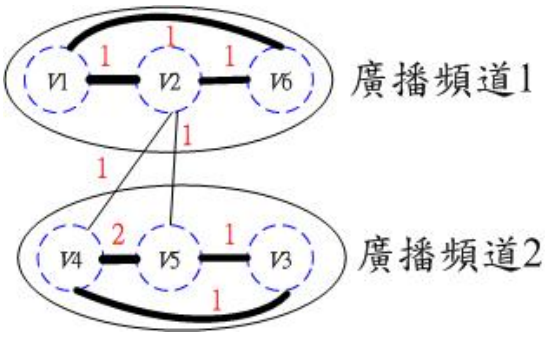

圖 3-5 真正交換頂點 v_3 和 v_6

圖 3-5 為第一次分割產生資料分割的結果，再重新進行第二次模擬廣播資料的交換，取得新的最大的獲利值。第二次模擬交換的結果為:

$\widehat{g_1} = g_{23} = -2$, $\widehat{g_2} = g_{14} = -2$, $\widehat{g_3} = g_{65} = 4$

即第一次交換頂點 v_2 和 v_3 得到值-2；第二次交換頂點 v_1 和 v_4 得到值-2；第三次交換頂點 v_6 和 v_5 得到 4。

最大的獲利值= $(\max \sum_{i=1}^{k} \widehat{g_i}) = 0$ $(k = 3)$

第二次模擬交換的結果顯示，把獲利值加總起來，產生新的最大的獲利值為 0，即第二次資料分割得到獲利為 0，因此本例子在進行第一次分割之後，得到的就是最後的結果，此時 KL 演算法就會終止並且輸出資料分割的結果，如上圖 3-5 所示。如果第二次分割，得到的最大的獲利值大於 0，KL 演算法就會繼續做下一次的資料分割，直到最大的獲利值小於等於 0 結束，下表 3-5 是 KL 演算法的演算步驟及說明。

表 3-5 KL 演算法的演算步驟

KL演算法
輸入:用戶端請求的資料集合
輸出:將資料有效率的分割放在廣播頻道1和廣播頻道2上，減少發生資料對抗的次數。
1.隨機分割用戶端請求的資料集合，平均放在廣播頻道1和廣播頻道2上。
2.找到二個未鎖定的資料 d_x, d_y ，其中 $d_x \in DS(C_1)$ 和 $d_y \in DS(C_2)$，互換之後得到最大的 g_{xy} 值。
3.將資料 d_x 和 d_y 模擬互換之後並且鎖定，並儲存獲利的值 $\widehat{g_i}$ 。

4.先更新過所有資料，然後重覆第2跟3的步驟，直到所有廣播頻道上的資料，都已經被交換過了(即全部資料都鎖定)。

5.找到k，使得$G_k=\sum_{i=1}^{k}\widehat{g_i}$獲利的值是最大的；其中$\widehat{g_i}$代表對應的$g_{xy}$值。

6.將所有資料解除鎖定

7.如果 $G_k>0$ ，將第5步驟會獲利的所有資料真正交換，再回到第2步驟重新模擬交換。

8.直到$G_k \le 0$

9.演算法結束並輸出有效率資料分割後的結果。

第四章 實驗結果和效能分析

本研究針對用戶端請求的資料集合，模擬均勻分配(Uniform Distribution)、常態分配(Normal Distribution)及指數分配(Exponent Distribution)的實驗環境。在模擬均勻分配的實驗環境中，用戶端提出請求的資料頻率都一致，也就是沒有熱門資料與冷門資料的區別，每一筆資料可能被用戶端請求的次數都一樣的情況。模擬常態分配的實驗環境中，用戶端請求資料的頻率就有所不同，比較熱門的資料被提出請求的頻率就比較高，也就是受到比較多的用戶端提出請求，比較冷門的資料被提出請求的頻率就比較低，也就是有一些資料只被少數人提出請求，但是在常態分配的實驗環境中，用戶端請求頻率高低的起伏並不會變動很大。在模擬指數分配的實驗環境中，用戶端提出請求的頻率變化極大，某一些極為熱門的資料，也就是被大多數用戶端提出請求的資料，資料的請求頻率就顯得特別高，而那些被極少數的用戶端請求的冷門資料，資料的請求頻率就顯得比較低，在頻率高低起伏明顯有很大的變化。我們以使用的KL演算法和隨機分割進行比較，在減少資料對抗次數的多寡進行效能上的分析和評估。隨機分割的方式，就是把用戶端請求的資料集合，隨機分割成兩半且平均分配在二個廣播頻道上。

表4-1 實驗模擬分配及參數設定

參數設定 / 模擬分配	請求頻率	廣播資料量	用戶端數量	選擇率
均勻分配	10	100~800	100~800	2%~6%
常態分配	1~35	100~800	100~800	2%~6%
指數分配	1~241	100~800	100~800	2%~6%

表 4-1 是實驗模擬分配及參數設定，在我們的實驗環境當中，主要要四個參數如下:請求頻率，廣播資料量，用戶端數量，選擇率。請求頻率表示資料項被用戶端存取的次數，就是資料的熱門程度，頻率愈高代表被請求的資料是愈熱門的，該筆資料同時被很多用戶端請求；相對地頻率愈低的話，表示該筆資料只被少數人請求的冷門資料。廣播資料量表示伺服器廣播資料給用戶端的數量。用戶端數量表示有多少個使用者進入廣播頻道上存取資料。選擇率表示廣播資料的數量與用戶端請求數量的比例，例如當廣播資料為 100，選擇率為 2%時，表示每個用戶端最多提出 2 個資料的請求。

第一節 以均勻分配模擬實驗

一開始先模擬均勻分配的實驗，我們改變伺服器廣播資料的數量，廣播資料的數量從 100~800，當用戶端的數量設為 200，選擇率設為 5%，本實驗的數據資料見表 4-2。

表 4-2 以均勻分配實驗廣播數量對資料對抗次數的影響

廣播資料量	**100**	**200**	**300**	**400**	**500**	**600**	**700**	**800**
隨機分割	322	838	1344	2256	2685	3525	4330	5272
KL 演算法	215	597	983	1829	1937	2503	2803	3617

在表 4-2 中，我們以均勻分配來進行實驗，用戶端的數量設為 200 及選擇率設為 5%，改變伺服器廣播資料的數量從 100 到 800，在用戶端的資料請求頻率都設定相同值 10，以隨機分割法和 KL 演算法進行資料對抗次數的比較。從實驗數據得知，當廣播資料量從 100 到 400 的時候，我們使用的 KL 演算法在資料對抗次數只有小幅度的減少，廣播資料量從 500 以後開始比較大幅度的減少，以這樣的趨勢來看，當伺服器廣播資料

的數量愈大的情況，我們的演算法能夠更有效率的降低資料對抗發生的次數，而且當伺服器廣播資料的數量很龐大之下，反而可以改善愈多。(如圖 4-1 所示)

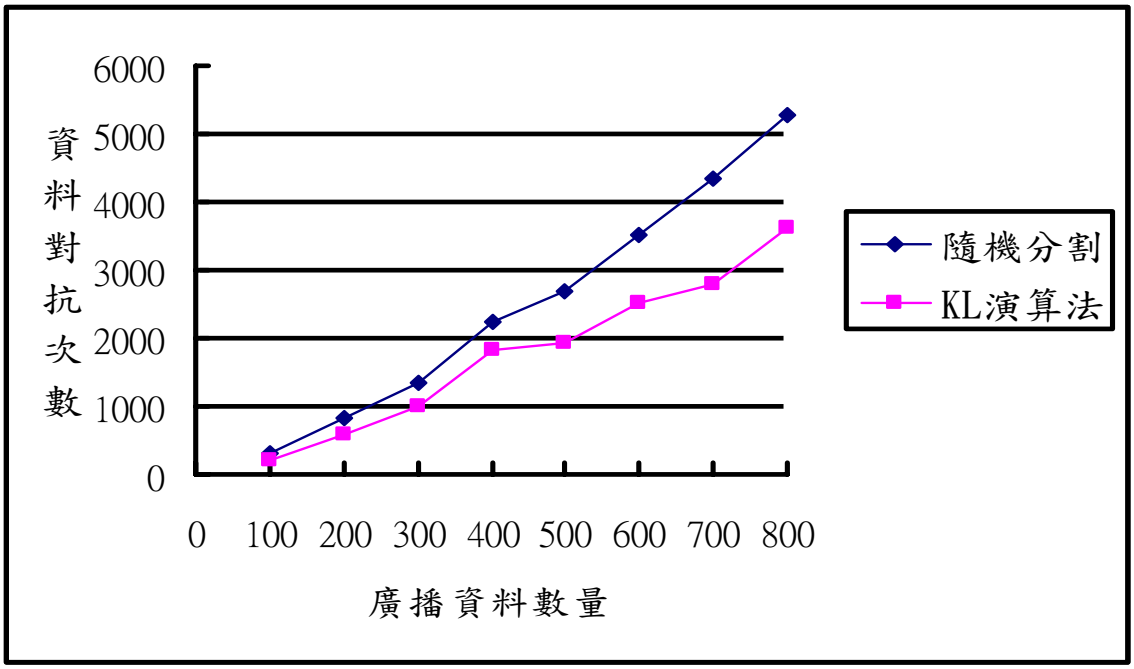

圖 4-1 廣播資料數量對資料對抗次數之影響(均勻分配)

見圖 4-1，探討廣播資料數量對資料對抗次數之影響，我們所使用的 KL 演算法，在資料對抗次數方面都少於隨機分割的方式。從圖中可以很明顯的看出，當伺服器廣播資料數量愈多的時候，在資料對抗次數上相對地也會提高很多。但是對於用戶端而言，我們的方法在改善資料對抗次數方面，隨著廣播數量的增加，減少資料對抗次數明顯地有大幅度的減少。這樣一來將來伺服器在做實際資料排程的時候，使得發生資料衝突的機率也會降低，發生資料衝突的次數減少的話，當用戶端進入多頻道的無線環境，在接收多重資料的時候在資料存取的時間可以縮短。接下來以均勻分配實驗用戶端數量對資料對抗次數的影響，我們改變用戶端的數量，用戶端的數量從 100~800，將廣播資料的數量設為 600，選擇率設為 2%，實驗的數據資料見表 4-3。

表 4-3 以均勻分配實驗用戶端數量對資料對抗次數的影響

用戶端數量	100	200	300	400	500	600	700	800
隨機分割	1018	1076	1449	1982	2885	3940	4488	4921
KL 演算法	150	192	587	1130	2026	3036	3679	4085

在表 4-3 中，我們以均勻分配來進行實驗，伺服器廣播資料的數量設為 600 及選擇率設為 2%，即用戶端最多提出 12 筆資料的請求，我們改變用戶端的數量從 100 到 800，用戶端的資料請求頻率都設定相同值 10，以隨機分割法和 KL 演算法進行資料對抗次數的比較。由實驗的數據顯示，當伺服器廣播資料的數量都相同的情形下，我們的方法在減少資料對抗次數的比例都一致，並沒有隨著用戶端數量的增加而大幅度的提升，因此伺服器廣播資料的數量對於減少資料對抗次數影響比較大。當用戶端的數量增加時，用戶端提出的資料請求也隨著增加，所以發生資料對抗的次數也相對地變多。下圖 4-2 是用戶端數量對資料對抗次數之影響，我們使用的 KL 演算法對資料對抗次數減少的比例一致，整體而言我們的方法也比隨機分割的方式，都有減少資料對抗發生的次數。

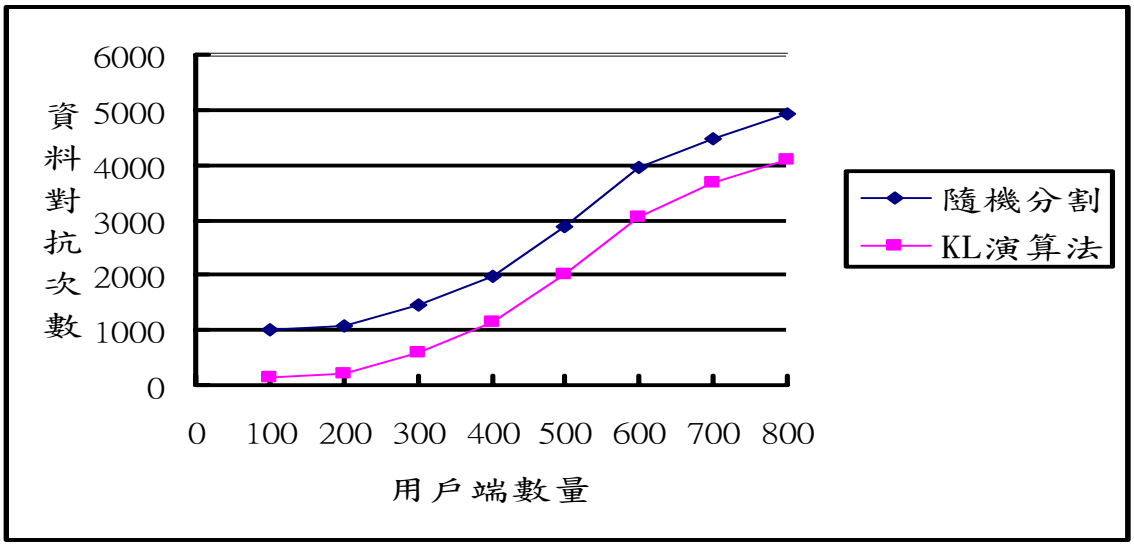

圖 4-2 用戶端數量對資料對抗次數之影響(均勻分配)

我們再以均勻分配實驗選擇率對資料對抗次數的影響，在進行模擬的實驗中，我們改變選擇率，選擇率從 2%~6%，伺服器廣播資料的數量設為 800，用戶端的數量設為 300。將選擇率設為 2%的時候，用戶端最多提出 16 筆資料的請求；將選擇率設為 3%的時候，用戶端最多提出 24 筆資料的請求；將選擇率設為 4%的時候，用戶端最多提出 32 筆資料的請求；將選擇率設為 5%的時候，用戶端最多提出 40 筆資料的請求；將選擇率設為 6%的時候，用戶端最多提出 48 筆資料的請求，實驗的數據資料見表 4-4。

表 4-4 以均勻分配實驗選擇率對資料對抗次數之影響

選擇率(%)	隨機分割	KL 演算法	效能提升 (%)
2	2016	905	+55
3	3606	2545	+29
4	6114	4728	+23
5	8656	7344	+15
6	11222	9779	+13
平均效能			+27

由表 4-4 可以看到當選擇率增加的時候，用戶端所請求的多重資料也跟著增加，因此所反應出來的資料對抗次數也相對地會持續增加。我們所使用的方法對於資料對抗次數，經過實驗數據統計的結果顯示，資料對抗次數平均比隨機分割減少了 1000 多次，效能也有很明顯的提升；從表 4-4 中，以均勻分配來進行實驗，我們的方法比隨機分割法平均可提升 27%的效能，因此當用戶端請求的多重資料量愈多的話，我們的方法也能在減少資料對抗的次數，達到不錯的效能。

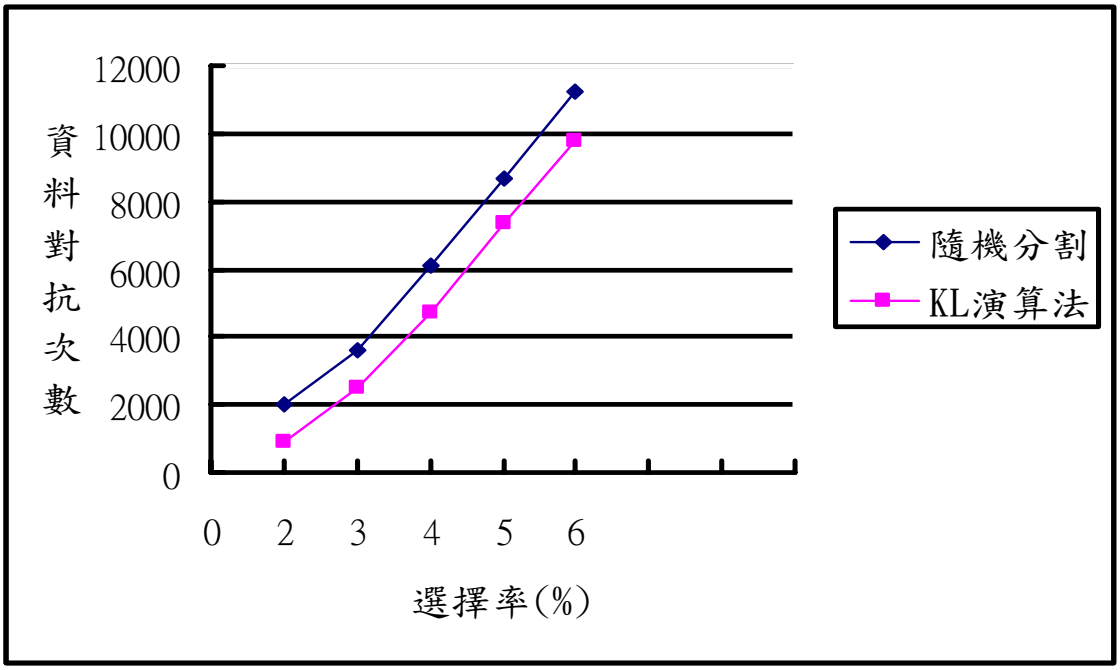

圖 4-3 選擇率對資料對抗次數之影響(均勻分配)

在圖 4-3 是在均勻分配的實驗環境中，選擇率對資料對抗次數之影響，我們使用的 KL 演算法對資料對抗次數減少的比例一致，整體而言我們的方法也比隨機分割的方式，都有減少資料對抗發生的次數。

第二節 以常態分配模擬實驗

接下來模擬常態分配的實驗，我們改變伺服器廣播資料的數量，廣播資料的數量從 100~800，將用戶端的數量設為 200，選擇率設為 5%，本實驗的數據資料見表 4-5。

表 4-5 以常態分配實驗廣播資料量對資料對抗次數之影響

廣播資料量	**100**	**200**	**300**	**400**	**500**	**600**	**700**	**800**
隨機分割	1197	2833	5303	7503	12065	10282	11493	13216
KL 演算法	345	1015	1611	2283	2268	2976	3123	3680

在表 4-5 中，我們以常態分配來進行實驗，在用戶端的資料請求頻率設定值介於 1 到 35 之間，以隨機分割法和 KL 演算法進行資料對抗次數

的比較。由表中的實驗數據顯示，當用戶端請求的資料頻率較高的時候，會造成資料對抗的次數相對會提高，因為隨機分割的方法是把一半的資料放在廣播頻道 1，另外一半的資料放在廣播頻道 2，所以當資料存取頻率很高且資料放在二個廣播頻道上的比例差不多時，資料對抗的次數就會提高很多，因此伺服器廣播資料數量在 300 之後，隨機分割法的資料對抗次數才會增加那麼多，而伺服器廣播資料數量在 500 的時候，剛好用戶端請求的很多筆資料的頻率都相當高，並且這些資料大部份都被分配在不同的頻道上，在資料對抗次數上達到 12000 多次。但是我們所使用的演算法，就是以用戶端請求資料的頻率優先考量，把這些頻率愈高的資料盡可能地放在相同的廣播頻道上，因此在資料對抗次數方面確實能很大幅度的減少。由常態分配的實驗中，用戶端的資料請求頻率設定值介於 1 到 35 之間，在請求頻率數值的變化並沒有很大，卻使得隨機分割對於資料對抗次數的變化明顯拉大。在圖 4-4 是廣播資料數量對資料對抗次數之影響，隨機分割法隨著廣播資料數量的增加，在資料對抗次數扶搖直上的變多，而我們的方法卻只是很小幅度的遞增。

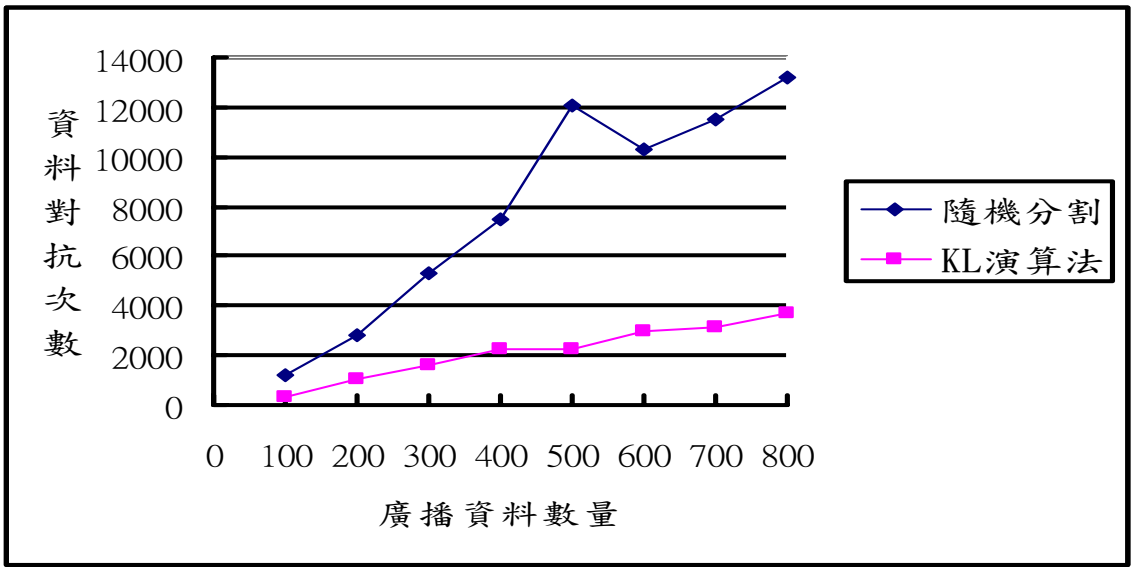

圖 4-4 廣播資料數量對資料對抗次數之影響(常態分配)

我們再以常態分配來實驗用戶端數量對資料對抗次數之影響，我們改變用戶端的數量，用戶端的數量從 100~800，將廣播資料的數量設為 500，選擇率設為 5%，實驗的數據資料見表 4-6。

表 4-6 以常態分配實驗用戶端數量對資料對抗次數的影響

用戶端數量	**100**	**200**	**300**	**400**	**500**	**600**	**700**	**800**
隨機分割	9338	11679	12974	14550	27421	31841	42419	34575
KL 演算法	1882	2902	6389	9201	13614	17378	26129	26059

在表 4-6 中，我們以常態分配來進行實驗，用戶端最多提出 25 筆資料的請求，我們改變用戶端的數量從 100 到 800，用戶端的資料請求頻率介於 1 到 35 之間，以隨機分割法和 KL 演算法進行資料對抗次數的比較。由實驗的數據顯示，用戶端的數量從一開始的 100 時，以隨機分割法的實驗結果在資料對抗次數高達 9000 多次，都遠比在均勻分配的次數高出很多，由此可見用戶端請求資料的頻率有很大的影響，隨著用戶端數量持續增多之下，資料對抗次數也是一直的增加。當用戶端的數量為 700 時，資料對抗次數是所有數據裡頭最多的，探討其原因則是因為用戶端請求的資料中，有很多筆資料被請求的頻率都介於 29 到 35 之間，也就是最熱門的資料比較多一點，所以造成的資料對抗次數也相對的比較多。而在我們使用的 KL 演算法，從一開始將用戶端數量設為 100 時，發生資料對抗次數比隨機分割法少了 7000 多次，在效能上就已經很有效率地減少資料對抗發生的次數。將用戶端數量設為 300 至 400 的時候，卻只比隨機分割法減少了 5000~6000 次左右，比用戶端為 100 時來得少一點，原因是用戶端請求的資料中，被請求的頻率介於 31~35 之間的資料，僅僅只有少數幾筆資料，也就是最熱門的資料比較少，但是次熱門的資

料仍然占一些比例。當用戶端數量在 500 以後，我們的方法都比隨機分割法減少了 10000 多次，表示我們把用戶端請求最熱門的資料，也就是存取頻率比較高的資料，都盡量擺放在同一個廣播頻道上，在效能上才會減少很多發生資料對抗次數。在圖 4-5 是用戶端數量對資料對抗次數之影響，一般而言當用戶端的數量變多時，表示同時請求的資料量也變得比較多，因此伺服器廣播的這 500 筆資料，而且同時被用戶端所請求，資料分配在不同廣播頻道上的可能性變高，所以圖 4-5 呈現出來的趨勢走向是很合理的。整體而言，我們的方法在減少資料對抗次數上仍然有比較大的減少，尤其是用戶端的數量在 500~700 時是改善最多的。

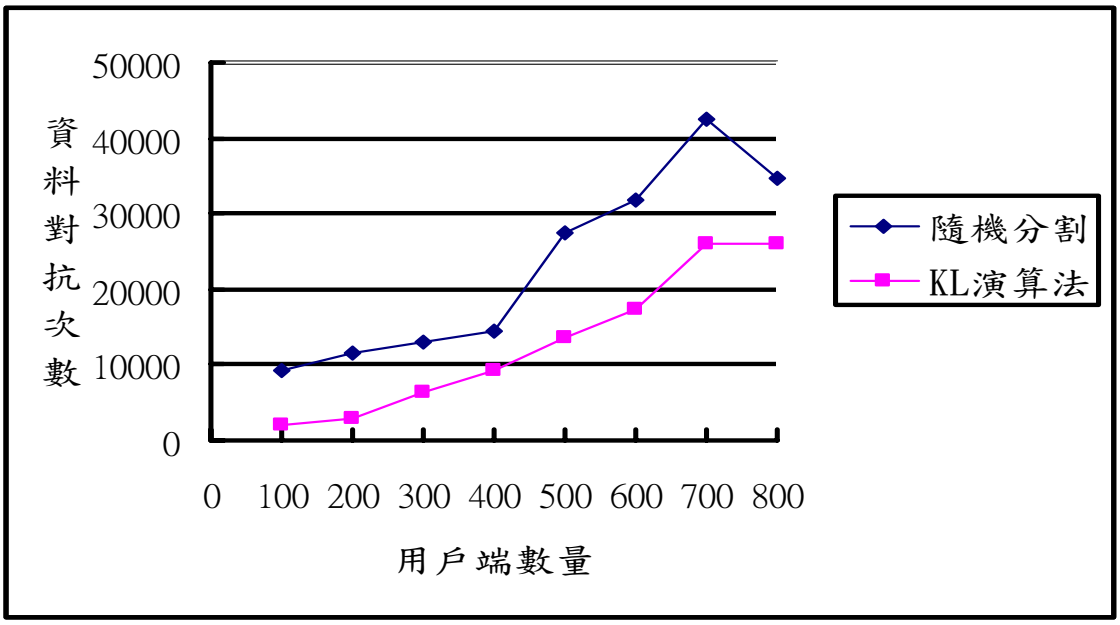

圖 4-5 用戶端數量對資料對抗次數之影響(常態分配)

以常態分配實驗選擇率對資料對抗次數的影響，在進行模擬的實驗中，我們改變選擇率這個參數，選擇率從 2%~6%，伺服器廣播資料的數量設為 500，用戶端的數量設為 200。選擇率為 2%的時候，用戶端最多提出 10 筆資料的請求；選擇率為 6%的時候，用戶端最多提出 30 筆資料的請求，實驗的數據資料見表 4-7。

表 4-7 以常態分配實驗選擇率對資料對抗次數之影響

選擇率(%)	隨機分割	KL 演算法	效能提升 (%)
2	2282	385	+83
3	4959	971	+80
4	5789	1432	+75
5	9634	2635	+73
6	14107	3416	+76
平均效能			+77

在表 4-7 中，我們以常態分配來進行實驗，分析選擇率對資料對抗次數之影響，在用戶端的資料請求頻率設定值介於 1 到 35 之間，以隨機分割法和 KL 演算法進行資料對抗次數的比較。由表中的實驗數據顯示，當選擇率從 2%~6%，也是就有 200 個用戶端提出資料的請求時，每一個用戶端最多提出 10~30 筆資料的請求。當選擇率增加時，資料對抗次數也跟著變多的趨勢，在隨機分割法所得到資料對抗次數的數據，跟以均勻分配實驗產生的數據可說是相同比例在增加，由此可見不同的分配對於選擇率的影響不大。但是比較值得關心的是，我們的方法在常態分配的實驗之下，在減少資料對抗次數方面卻有很明顯的改善，而且效能上遠比均勻分配的實驗結果來得好，會產生這樣的結果的主要原因，則是因為均勻分配的請求頻率都是 10，而常態分配的請求頻率在 1~35 之間，我們的方法把請求頻率愈高的資料都盡量放在相同頻道上，所以在常態分配的實驗結果才會表現較佳，而且當選擇率遞增的同時，我們的方法在資料對抗次數上只是小幅度的增加。從表 4-7 進一步來分析，當選擇率一直增加的情形下，我們的方法反而減少更多資料對抗次數，依照這樣的趨勢來看，用戶端請求的多重資料量愈多的話，我們的方法在減少資料

對抗的次數，反而能夠達到更好的效能，以常態分配來進行實驗的環境，我們的方法比隨機分割法平均可提升 77%的效能。

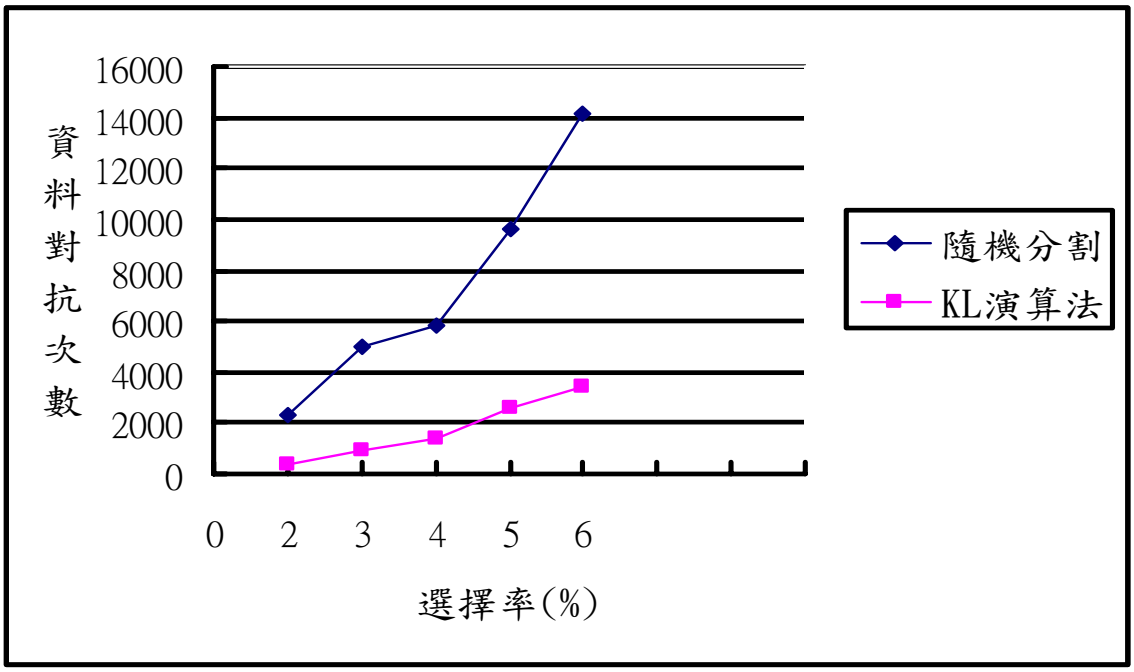

圖 4-6 選擇率對資料對抗次數之影響(常態分配)

在圖 4-6 是在常態分配的實驗環境中，選擇率對資料對抗次數之影響。從圖中可以很明顯看的出來，當選擇率從 2%~6%，隨機分割法的資料對抗次數增加的幅度愈來愈大，但是我們使用的方法卻增加不大，當選擇率在 3%以後，我們的方法跟隨機分割法比較之下，有愈來愈明顯拉大的趨勢，而且會相差甚大。

第三節 以指數分配模擬實驗

最後一個實驗模擬指數分配的環境，我們改變伺服器廣播資料的數量，廣播資料的數量從 100~800，將用戶端的數量設為 200，選擇率設為 5%，本實驗的數據資料見表 4-8。

表 4-8 以指數分配實驗廣播數量對資料對抗次數的影響

廣播資料量	**100**	**200**	**300**	**400**	**500**	**600**	**700**	**800**
隨機分割	1342	2862	8315	12382	9723	10866	11648	18146
KL 演算法	266	589	1426	1863	2859	2967	2922	3103

在表 4-8 中，我們以指數分配來進行實驗，在用戶端的資料請求頻率設定值介於 1 到 241 之間，以隨機分割法和 KL 演算法進行資料對抗次數的比較。在指數分配的實驗環境中，由於用戶端請求資料的頻率相差極大，一些特別熱門的資料被用戶端請求的頻率就特別高，所以在發生資料對抗的次數也會比較多。由表中的實驗數據顯示，隨機分割法在廣播資料量 200 以後，資料對抗次數就顯得很大，在伺服器廣播資料量 400 的時候，隨機分割法的資料對抗次數高達 12000 多次，突然在次數上暴增的原因，是因為用戶端請求最熱門的資料都被分配在不同的頻道上，資料請求的頻率從 179 到 241 之間的資料量不少，所以在資料對抗的次數才一下子變多。我們使用的方法在減少資料對抗次數方面，也一直很大幅度的減少，主要理由是我們已經把用戶端請求最熱門的資料，也就是資料請求頻率比較高的資料，都盡可能地放在同一個廣播頻道上，因此隨著伺服器廣播數量的遞增，而我們的方法在資料對抗次數上，卻只是增加一點點，並且伺服器廣播的資料量從 500~800，我們的方法在發生資料對抗次數都差不多。在圖 4-7 是以指數分配來進行的模擬，實驗廣播資料數量對資料對抗次數之影響，隨機分割法在伺服器廣播資料數量 200 以後，在資料對抗次數上就變化極大，而我們的方法跟隨機分割法相較之下，非常明顯的改善很多在資料對抗的次數，從圖形來看，我們的方法在發生資料對抗的次數只是微小的增加。

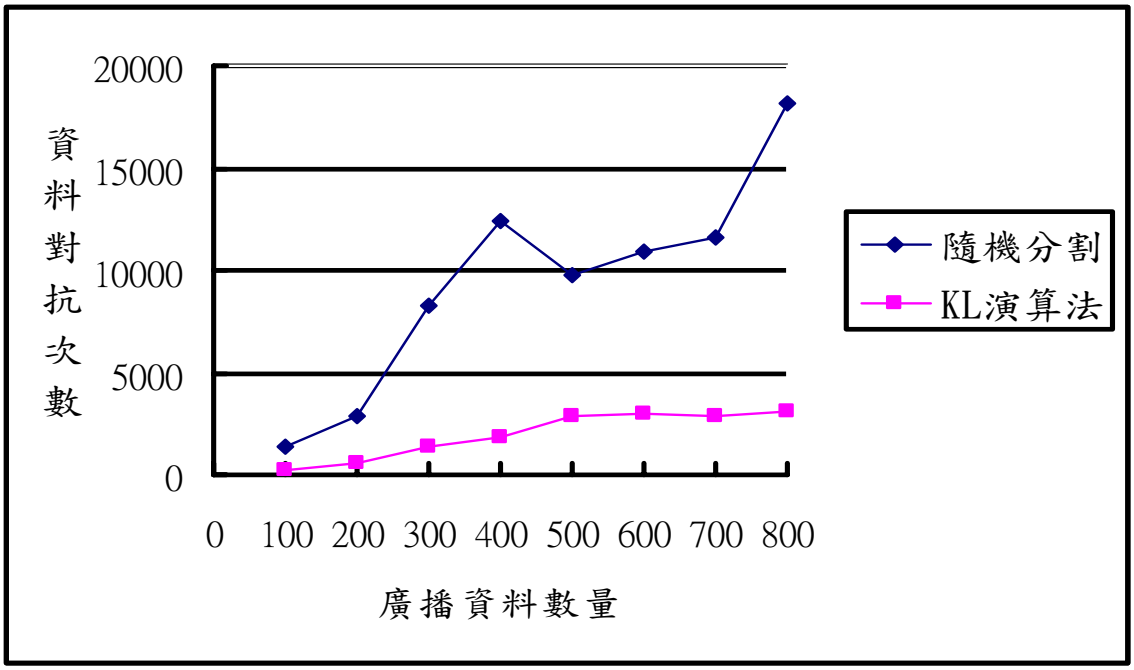

圖 4-7 廣播資料數量對資料對抗次數之影響(指數分配)

我們再以指數分配來實驗用戶端數量對資料對抗次數之影響，我們改變用戶端的數量，用戶端的數量從 100~800，將廣播資料的數量設為 500，選擇率設為 5%，實驗的數據資料見表 4-9。

表 4-9 以指數分配實驗用戶端數量對資料對抗次數的影響

用戶端數量	**100**	**200**	**300**	**400**	**500**	**600**	**700**	**800**
隨機分割	9036	30773	51123	57488	60936	73399	86231	97462
KL 演算法	1016	2173	4402	5280	13614	16736	22476	24579

在表 4-9 中，我們以指數分配來進行模擬實驗，用戶端最多提出 25 筆資料的請求，我們改變用戶端的數量從 100 到 800，用戶端的資料請求頻率介於 1 到 241 之間，以隨機分割法和 KL 演算法進行資料對抗次數的比較。從常態分配到指數分配，隨機分割法在用戶端數量 100 的時候，資料對抗的次數就已經相當的高，因為用戶端請求資料的頻率變大了，尤其在指數分配的請求頻率變動幅度又比常態分配來得大，因此資料對

抗的次數相當的也提高不少。隨機分割法在用戶端數量 300 的時候，資料對抗的次數高達 50000 多次，我們探究其原因是用戶端請求資料的頻率在 241 時，也就是有很多筆資料都是最熱門的，同時被很多用戶端所提出要求，而這些資料有一半放在廣播頻道 1 上，另外一半的資料放在廣播頻道 2 上，所以發生資料對抗的次數才會如此地高，經過我們的方法進行多重資料的分配，盡量將頻率特別高的資料放在同一個頻道上，實驗結果資料對抗的次數才發生了 4000 多次，在效能上確實減少了很多資料對抗的次數。隨機分割法在用戶端數量 500 以後，也是由於用戶端提出請求很多筆資料都是最熱門的，而這些資料在隨機分割的時候，都剛好被分配在不同的頻道上，所以發生資料對抗的次數一直增加。隨機分割法從用戶端數量 100~800，發生資料對抗的次數都是一直往上爬升，而我們的方法有很明顯的改善，在對抗的次數上只是逐漸的慢慢遞增的趨勢。

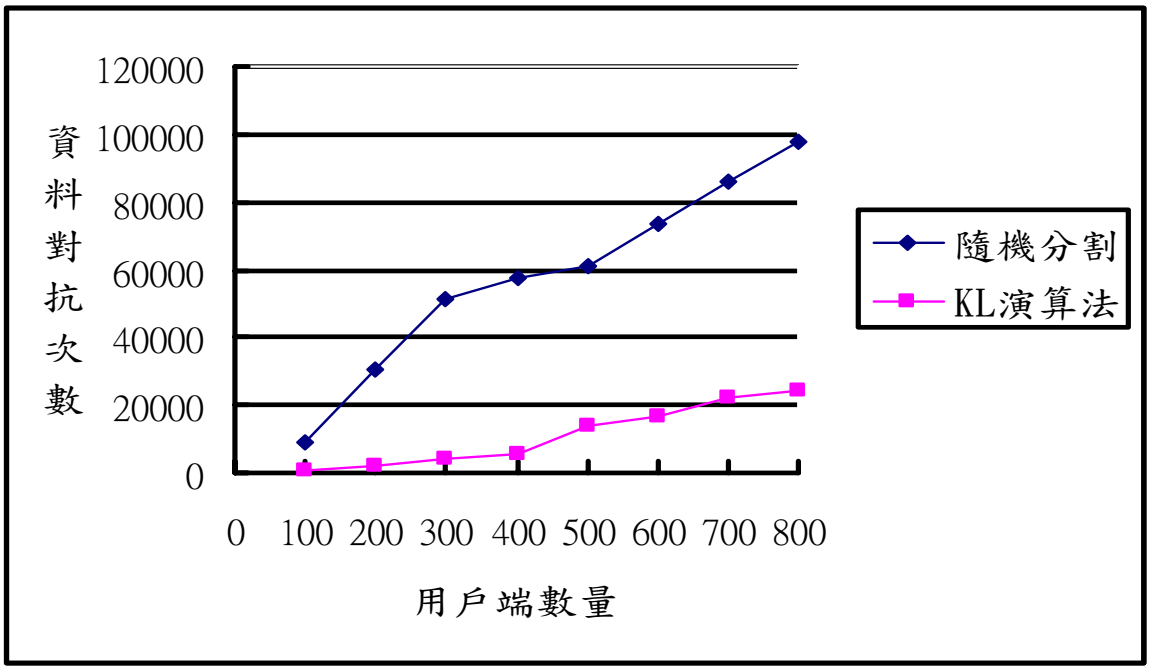

圖 4-8 用戶端數量對資料對抗次數之影響(指數分配)

在圖 4-8 是用戶端數量對資料對抗次數之影響,從圖形中可以很明顯的看得出來，用戶端的數量從 100~800，當用戶端的數量增加的時候，請求的資料量也伴隨著增加，因此資料對抗次數持續增加是合理的情況。在隨機分割方面，尤其在用戶端的數量為 200 以後，資料對抗次數就不斷地增加幅度很大，然而我們的方法只是小幅度的增加。從圖中我們可以得知，在指數分配的實驗環境中，由於受到用戶端資料請求頻率的影響，導致隨機分割法的變化極大，而我們的方法卻很緩慢的增加次數。

以指數分配實驗選擇率對資料對抗次數的影響，在進行模擬的實驗中，我們改變選擇率這個參數，選擇率從 2%~6%，伺服器廣播資料的數量設為 500，用戶端的數量設為 200。將選擇率為 2%的時候，用戶端最多提出 10 筆資料的請求；將選擇率為 6%的時候，用戶端最多提出 30 筆資料的請求，實驗的數據資料見表 4-10。

表 4-10 以指數分配實驗選擇率對資料對抗次數之影響

選擇率(%)	隨機分割	KL 演算法	效能提升 (%)
2	4455	370	+92
3	3261	670	+79
4	8530	1423	+83
5	9914	2927	+77
6	15207	3345	+78
平均效能			+82

在表 4-10 中，我們以指數分配來進行實驗，分析選擇率對資料對抗次數之影響，在用戶端的資料請求頻率設定值介於 1 到 241 之間，以隨機分割法和 KL 演算法進行資料對抗次數的比較。由表中的實驗數據顯

示，當選擇率從 2%~6%，也是就有 200 個用戶端提出資料的請求時，每一個用戶端最多提出 10~30 筆資料的請求。在隨機分割法的資料對抗次數方面，跟之前的均勻分配及常態分配比較後發現，在次數上並沒有很大的變化，因此用戶端請求資料的頻率對於選擇率的影響並沒有很顯著，但是在我們使用的 KL 演算法，在均勻分配時平均提升 27%的效能，在常態分配時平均提升 77%的效能，在指數分配時平均提升 82%的效能，後兩種分配因為資料請求頻率變化較大，而我們的方法是以資料頻率較高的優先配置，因此在效能上表現反而較好，指數分配又比常態分配好一點。在圖 4-9 是指數分配的實驗環境中，選擇率對資料對抗次數之影響。從圖中可以很明顯看的出來，隨機分割法的資料對抗次數增加的幅度極大，但是我們使用的方法只是緩慢的增加，我們的方法跟隨機分割法比較之下，有愈來愈明顯拉大的趨勢，而且會相差甚大。

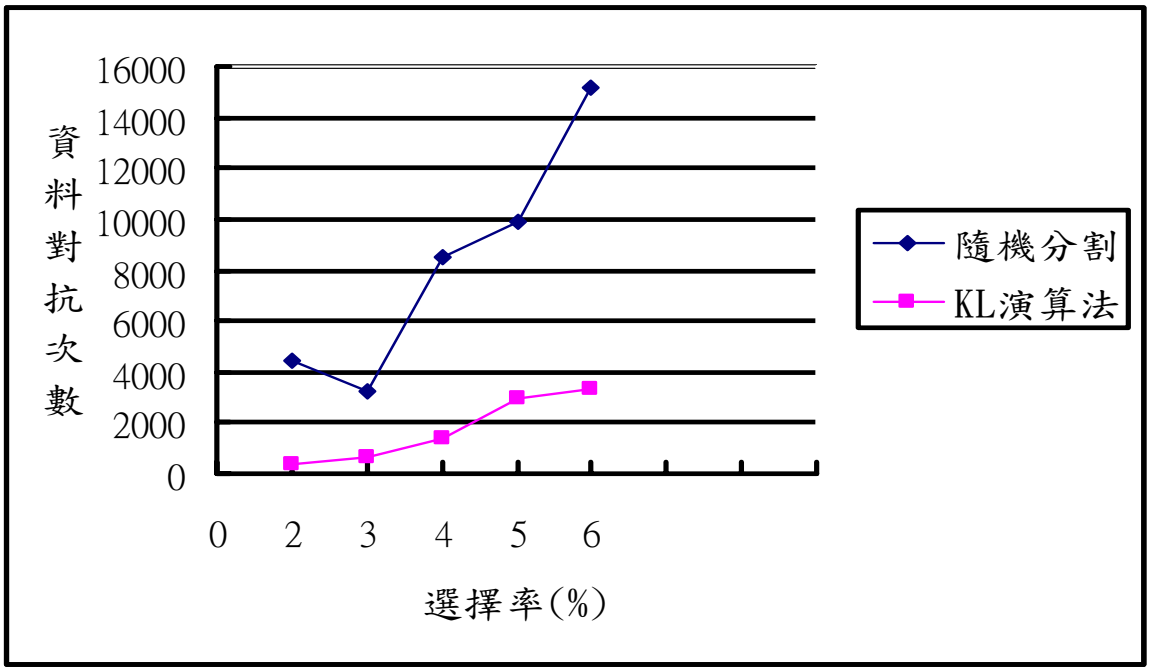

圖 4-9 選擇率對資料對抗次數之影響(指數分配)

第五章 結論和未來展望

爲了節省頻寬並將資源有效利用於無線網路的環境中，伺服器用廣播的方式廣播資料給用戶端，這樣一來伺服器可以服務大量的用戶端，就可以節省頻寬資源的浪費。在無線網路的環境中，在過去很多的研究中，確實被證明廣播的方法能有效率的解決頻寬不足的問題。由於先前的很多研究都在探討用戶端請求單一資料，而我們在本篇在探討用戶端請求多重資料，並且把這些多重資料分配在二個等長的廣播頻道上。在多頻道的資料排程上，我們可以把它分成二個部分，第一個部分先把用戶端請求的資料集合，決定資料應該放在那一個廣播頻道上；第二個部分是做實際的資料排程，在本篇我們所使用的 KL 演算法，只有做第一個部分，目標使得發生資料對抗的次數盡量減少，將來做實際的資料排程時，讓廣播的資料會發生資料衝突的機率能夠降低。在模擬實驗過程中，我們以均勻分配、常態分配及指數分配來進行，從實驗結果得知，我們使用的方法，能有效率的改善並且降低用戶端資料對抗的次數，並且我們的方法在常態分配及指數分配的模擬實驗中，分別提升了 77%及 82%。在未來的研究方向，我們將會探討 K 階分割演算法在多頻道的廣播環境，並且進一步分析對用戶端請求資料的存取時間之影響。

參考文獻

[1] Swarup Acharya, Rafael Alonso, Michael Franklin, and Stanley Zdonik, "Broadcast Disk: Data Management for Asymmetric Communication Environments," *Proceedings of ACM SIGMOD International Conference*, Pages: 199-210, 1995.

[2] Akio Ando, Toru Imai, Akio Kobayashi, Haruo Isono, and Katsumi Nakabayashi, "Real-Time Transcription System for Simultaneous Subtitling of Japanese Broadcast News Programs," *IEEE Transactions on Broadcasting*, vol. 46, no. 3, Pages: 189-196, 2000.

[3] D. Aksoy and M. Franklin, "Scheduling for Large-Scale On-Demand Data Broadcasting," *Proceeding of the 1998 IEEE INFOCOM Conference*, Pages: 651-659, 1998.

[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: 18-26, 2004.

[5] Joonho Cho, Seungtaek Oh, Jaemyoung Kim, Hyeong Ho Lee, and Joonwon Lee, "Neighbor Caching in Multi-Hop Wireless Ad Hoc Networks," *IEEE Communications Letters*, Pages: 525-527, 2003.

[6] Maggie Xiaoyan Cheng, Jianhua Sun, Manki Min, and Ding-Zhu Du, "Energy-Efficient Broadcast and Multicast Routing in Ad Hoc Wireless Networks," *Proceedings of the 2003 IEEE International on Performance, Computing, and Communications Conference*, Pages: 87-94, 2003.

[7] Ming-Syan Chen, Kun-Lung Wu, and Philip S. Yu, " Optimizing Index Allocation for Sequential Data Broadcasting in Wireless Mobile Computing," *IEEE Transactions on Knowledge and Data Engineering*, vol. 15, no. 1, Pages: 161-173, 2003.

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

[9] 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 Software*, Pages: 173-181, 2002.

[10] Ye-In Chang and Wu-Han Hsieh, "An Efficient Scheduling Method for Query-Set-based Broadcasting in Mobile Environments," *Proceedings of the 24th International Conference on Distributed Computing Systems Workshops*, Pages: 478-483, 2004.

[11] Carla-Fabiana Chiasserini, Imrich Chlamtac, Paolo Monti, and Antonio Nucci, "An Energy-Efficient Method for Nodes Assignment in Cluster-Based Ad Hoc Networks," *Wireless Networks*, Pages: 223-231, 2004.

[12] 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.

[13] Le Gruenwald, Muhammad Javed, and Meng Gu, "Energy-Efficient

Data Broadcasting in Mobile Ad-Hoc Networks," *Proceedings of the International Database Engineering and Applications Symposium*, Pages: 64-73, 2002.

[14] Chih-Lin Hu and Ming-Syan Chen, "Adaptive Balanced Hybrid Data Delivery for Multi-Channel Data Broadcast," *IEEE International Conference on Communications*, Pages: 960-964, 2002.

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

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

[17] C.H. Hsu, G. Lee, and A.L.P. Chen, "A Near Optimal Algorithm for Generating Broadcast Programs on Multiple Channels," *ACM International Conference on Information and Knowledge Management*, Pages: 303-309, 2001.

[18] B.W. Kernighan and S.Lin, "An Efficient Heuristic Procedure for Partitioning Graphs," *The Bell System Technical Journal*, vol. 49, no. 2, Pages: 291-308, 1970.

[19] Seung-Jin Kim, Woo-Jae Kim, and Young-Joo Suh, "Efficient

Broadcast Schemes with Transmission Power Control in Mobile Ad Hoc Networks,"*IEEE Communications Society*, Pages: 3859-3863, 2004.

[20] Guanling Lee, Shou-Chih Lo, and Arbee L.P. Chen, "Data Allocation on Wireless Broadcast Channels for Efficient Query Processing," *IEEE Transactions on Computers*, vol. 51, no. 10, Pages: 1237-1252, 2002.

[21] Guanling Lee and Shou-Chih Lo, "Broadcast Data Allocation for Efficient Access of Multiple Data Items in Mobile Environments," *ACM Mobile Networks and Applications*, Pages: 365-375, 2003.

[22] Chi-Wai Lin, Haibo Hu, and Dik-Lun Lee, "Adaptive Realtime Bandwidth Allocation for Wireless Data Delivery," *Wireless Networks*, Pages: 103-120, 2004.

[23] Shou-Chih Lo and Arbee L.P. Chen, "Optimal Index and Data Allocation in Multiple Broadcast Channels," *Data Engineering*, Pages: 293-302, 2000.

[24] Sung-Hwa Lim and J.-H. Kim, "Real-Time Broadcast Algorithm for Mobile Computing," *The Journal of Systems and Software*, Pages: 173-181, 2004.

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

[26] Wen-Chih Peng and Ming-Syan Chen, "Dynamic Generation of Data Broadcasting Programs for a Broadcast Disk Array in a

Mobile Computing Environment," *Proceedings of ACM International Conference on Information and Knowledge Management*, Pages: 38-45, 2000.

[27] 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: 223-229, 2004.

[28] Weiwei Sun, Weibin Shi ,and Bole Shi, "A Cost-Efficient Scheduling Algorithm of On-Demand Broadcasts," *Wireless Networks*, Pages: 239-247, 2003.

[29] Caimu Tang, Cauligi S. Raghavendra , and Viktor Prasanna, "Energy Efficient Adaptation of Multicast Protocols in Power Controlled Wireless Ad Hoc Networks," *Proceedings of the International Symposium on Parallel Architectures, Algorithms and Networks* , Pages: 80-85, 2002.

[30] O.B.V. and Mohanty H., "NICD: a Novel Indexless Wireless On-Demand Data Broadcast Algorithm," *Proceedings ITCC International Conference on Information Technology: Coding and Computing*, Pages: 730-734, 2004.

[31] Yiqiong Wu and Guohong Cao, "Stretch-Optimal Scheduling for On-Demand Data Broadcasts," *Proceedings Tenth International Conference on Computer Communications and Networks*, Pages: 500-504, 2001.

[32] Wai Gen Yee, Shamkant B. Navathe, Edward Omiecinski, and Christopher Jermaine, “Efficient Data Allocation over Multiple Channels at Broadcast Servers,” *IEEE Transactions on Computers*, vol. 51, Pages: 1231-1236, 2002.

[33] Wai Gen Yee and Shamkant B. Navathe , “Efficient Data Access to Multi-channel Broadcast Programs,” *ACM CIKM’03*, Pages: 153-160, 2003.