南　華　大　學

資訊管理研究所

碩士論文

有向性資料廣播問題串接的啟發式解法

Greedy Heuristic for Problem of Scheduling Directed Wireless Data

研 究 生：陳珊珊

指導教授：蔡德謙　博士

中華民國 九十六 年 六月

南　華　大　學

資訊管理學系

碩 士 學 位 論 文

有向性資料廣播問題串接的啟發式解法

研究生：凍 珊 珊

經考試合格特此證明

口試委員：

指導教授：蔡德謙

系主任(所長)：

口試日期：中華民國　96　年　6　月　8　日

誌　　謝

這一路走來在整個求學的過程中，要感謝的人實在太多了，只能說自己何德何能。身邊那些曾經支持我、幫助我、鼓勵我的人，總讓我在最無助、彷徨時開導我，給我無比的信心。承蒙 蔡德謙老師二年來的耐心指導，尤其是老師嚴謹的治學態度和圓融的處事方法，仍是我努力學習的對象。老師，真的謝謝您了!當然，也給予 吳光閔老師及 吳金山老師表達誠摯的謝意。感謝在口試時提供學生精闢的見解和分析建議，讓學生對於論文的思考和寫作都有相當的幫助，使得學生的論文更具完善。還有我的家人，該是我回饋給予的時候了，心中的感動是無法言語。

除了家人和師長外，朋友們的援助也不容忽視。無線廣播小組的大家庭成員們，不管是嘉明、小童、士颺、昭勳、淑玲這些活寶庫學長姐們，還是淳淳、詠生和慶豐，都能適時的給予協助與勉勵。當然不能忘的雅君、小賴與岳勳，我們一同成長、學習，而你們的學習精神同時也是我效法的對象。

在完成論文的同時，也結束學生的求學階段。學生心懷感激和滿滿回憶走向另一個人生的階段。在此，謹以本論文向所有給過我援助的朋友們致上最真誠的感謝和感動。

有向性資料廣播問題串接的啟發式解法

研 究 生: 陳 珊 珊　　指 導 教 授: 蔡 德 謙 博士

南 華 大 學　資 訊 管 理 學 系　碩 士 班

摘　　要

隨著網路使用者的日漸增加，網路資訊的大量充斥，用戶端對於網頁資訊的取得也要求更快速、更有效率，但礙於現實網路環境頻寬有限及無線資訊產品能源限制，因此在整個無線網路的環境中，資料項的廣播儼然扮演重要的角色及應用。

本研究著重於無線網路廣播資料排程問題，以廣播資料項存在的相依性為主要的考量，藉由有向無循環圖形(Directed acyclic Graph ,DAG)的資料存取模式及其表示順序限制作為基礎，在資料排序路徑的搜尋上，運用各種不同的貪婪拓撲排序之策略，經由各個策略下所產生的有向性線性序列，運算其頂點之間的平均路徑長度，並且相互比較彼此優劣。其目的是試圖讓用戶端在查詢相關性資料時，可以降低讀取相關性資料平均等待查詢時間，找出最佳廣播資料排序，縮短資料項間的相依長度，確保聯網的服務品質(Quality of Service ,QoS)。

經本研究實驗結果數據顯示，所提出的連續性雙向演算法，確實比

其他以貪婪拓撲排序的策略之演算法，更能獲得較短的平均查詢讀取時間。

關鍵詞:資料排序、平均查詢讀取時間、貪婪演算法

Greedy Heuristic for Problem of Scheduling Directed Wireless Data

Student: Shan -Shan Chen Advisors：Dr. Der chian Tsaih.

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

ABSTRACT

Since the resource of transmission bandwidth and the power of portable devices are much limited under the wireless network environment, an efficient and effective data broadcasting algorithm is important in disseminating information to mobile clients in wireless broadcast environment.

In this paper, we consider the problem of efficiently generating the broadcast schedule when there are possible data dependence exist between each broadcasted data item. By arranging the vertex set by topological sort with greedy strategy, one can generate a directed optimal linear broadcast schedule. One key issue related to data broadcasting is the technique used for server to organize data for clients accessing the broadcasted information more efficiently, which is in terms of the average response time. The short average response time can ensure the quality of service.

Our result shows that by using the proposed schemes one can achieve a better broadcast schedule which reduces average response time.

Keywords: Wireless Broadcast、Greedy Algorithm、Average Response Time

目　錄

表　　目　　錄

圖　　目　　錄

第一章 導論

1.1 研究背景與動機

由於無線網路（wireless network）科技日趨成熟，網際網路的使用早就融入生活之中，因此造就資訊膨脹世代，而快速且多元化取得資訊，已成為網際網路基本的需求。但是因處於無線網路環境下網路頻寬尚有不足之處，加上無線資訊產品日益普及，著重考量方便性、易攜性之情況下，無線資訊產品的能源限制也成為無線廣播發展的難題之一。面對無線網路的頻寬限制和無線資訊產品的能源限制，近年來學者專家們正極力思維如何能運用較佳的資料排程廣播資訊，讓資訊的傳輸更有效率，以解決目前無線網路所面臨的各種缺失狀況。現實中無線網路的環境，伺服端透過廣播機制，提供大量公共資訊，例如:電子的即時新聞、股票價格及氣象等資訊，並利用資料排程系統決定出一組廣播序列(broadcast cycle)，經由廣播頻道(channel)傳遞給予眾多的用戶端。使得用戶端能於廣播序列上，以最短時間內取得所需資訊降低獲取資訊之成本。

1.2 研究目的

近年來學者專家們對於無線網路廣播資料排程的範疇中，所探討的廣播資料主要都是針對於資料項之間的獨立性[21]。但現實的環境中，有許多真實運用，資料項之間是存在相依性[5,7,12]，例如用戶端瀏覽網站時，網頁上所呈現的音效檔和影像檔，這些資料項都是在瀏覽網頁時同時被請求的，而這些音效檔和影像檔便相依於上一個主網頁。

在無線廣播問題範圍中，為研究學者們爭相議論其分別著眼於各層面，其主要包括廣播資料庫設置、廣播序列播放控制、節省用戶端的能源等。

過去針對無線廣播之研究中有學者提出，將資料項受歡迎程度，做為安排廣播序列的考量。亦也對熱門資料項提出運用 pure-push-based技術，而冷門資料項則由 pure-pull-based技術來處理。

除此之外，應用於能源節制議題上[10,13,16]，大多以自動選取調頻為基礎概念，行動用戶端由廣播頻道上接收額外的資訊(如廣播索引:需等待多少時間後請求資料項會在廣播頻道上廣播)，此時行動裝置都處於低耗電的休眠模式中，直到傾聽到所請求的資訊時，行動裝置才恢復活動狀態。 Indexing和hashing 便是用以改善調頻時間(tuning time)之方法[13,14]。

本研究著重於資料項之間的相依關聯性，運用各種貪婪策略加入節點

階層值經由排序後，依所產生廣播線性排序並利用此排序運算出資料項間相依性長度，希望藉此能找出較佳廣播序列，以求降低其用戶端之平均查詢讀取時間。

1.3 研究架構

有關本研究架構說明如下:

第一章、說明本研究之背景、動機以及研究目的。

第二章、簡述無線網路環境及架構，並整理本研究相關之文獻，其中包含圖形結構(DFS 與 BFS)、貪婪演算法、混合式演算法、等。

第三章、詳細說明本文欲改善之廣播問題描述以及符號的定義與基本假設。

第四章、旨在說明我們所應用雙向貪婪演算法、應用階層式合併凝聚叢集演算法，試圖找出較佳之廣播序列。

第五章、本研究的實驗結果，以 JAVA 程式語言撰寫程式進行模擬，針對本研究所發現之問題，使用其演算法解決的效能分析。

第六章、結論，依據實驗結果所得加以分析和建議。

第二章 背景探討

2.1 無線廣播環境

2.1.1 無線網路

自美國電機電子工程學會(IEEE)於 1997 年 6 月頒佈 IEEE802.11 標準協定後，無線網路便展開新歷程。歷經多年科技快速演進，網路使用者的增加與應用範圍擴大，無線網路不再是新鮮話題。人們可以不受“線”制，隨時隨地使用筆記型電腦瀏覽網頁、收發郵件或用 MSN 與朋友談心聊天；在自助旅行時再也不怕迷路，只要利用具無線功能的 PDA 或筆記型電腦，就能輕鬆查詢到鄰近景點並建議旅遊路線；開車前往市區也不用邊開邊找停車位，只要透過網路既可查詢停車場的空車位剩幾個或位於哪裡。

依據無線網路連接的方式可分為，利用無線基地台和有線區域網路連結的主從式無線網路(Infrastructure Network); 不需要無線基地台的對等式無線區域網路(Ad-Hoc Network)。

主從式無線網路(Infrastructure Network)，行動用戶端使用行動裝置透過固定式基地台(Base Station, BS)或存取點（Access Point, AP）和現有網

路做連結。在此模式下，基地台為無線端的主控，負責無線網路與有線網路之間的封包傳遞與訊息控制 。如有數個獨立的網路則可透過無線基地台將其串接，各獨立網路即可互相做連結。

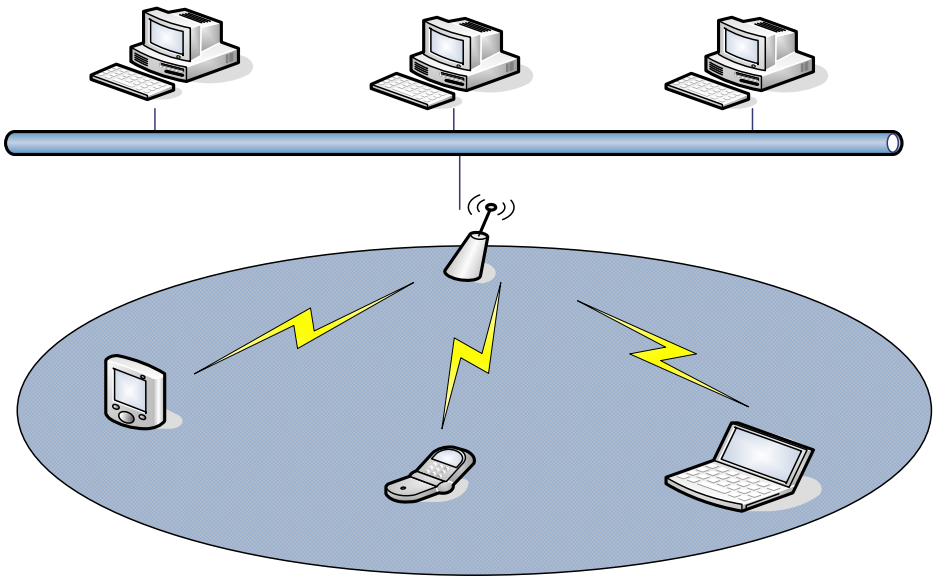


圖 1、主從式無線網路示意圖

對等式無線網路區域(Ad-hoc Network)，在此架構下利用無線網路卡構成無線區域網路，行動用戶端使用無線網路的資訊設備是運用點對點(Peer to Peer)模式互通。行動用戶端扮演著伺服端的角色，可以透過對方交換資訊，不須任何基地台或擷取點(access point)設備，便可快速架設無線網路環境。資訊設備進行無線傳輸時，不會因為連線之基地台改變位置而中斷連線，行動式資訊裝置(如:筆記型電腦、PDA)在使用中若有

跨越不同基地台的涵蓋範圍時，不使網路斷線(Roaming)。可適用於相同系統之各個無線基地台間的傳輸，目前現行之行動通訊皆屬於無線漫遊網路架構。

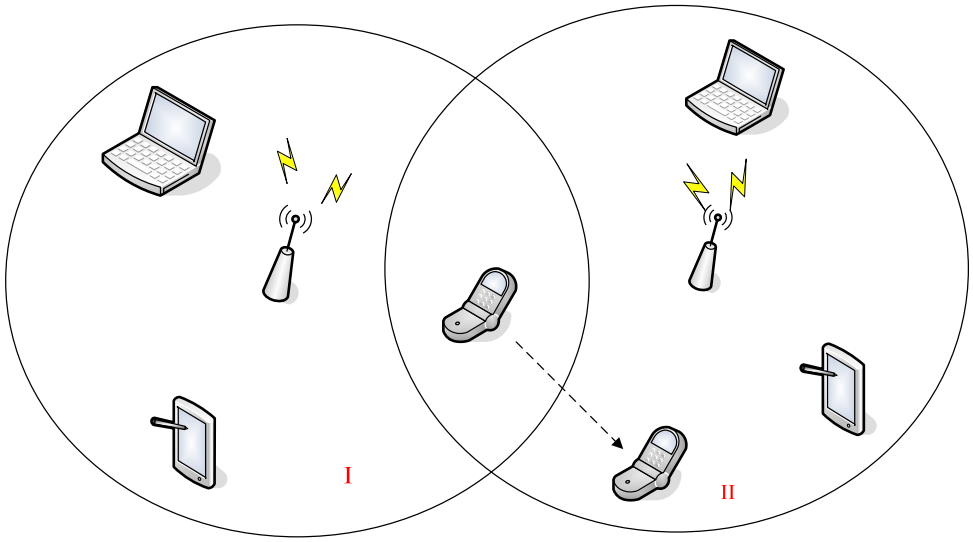


圖 2、對等式無線網路示意圖

2.1.2 無線資訊廣播

在傳統主從式 Client/Server 無線網路架構下，皆由一般行動用戶端 Client 向伺服器 Server 提出資料請求;伺服器在依據行動用戶端的請求提供相關資訊。在資料廣播領域裡，主要區分為二種基本模式。一為推式廣播系統(Push-based Broadcasting)[4,22];此廣播模式下由伺服器分析某

條件之歷史紀錄，決定好資料排序在主要廣播頻道(main channel)上不斷地循環式(Broadcast Cycle)廣播資料，而行動用戶端則進入頻道監聽頻道上的資料，便取得所需要的資料，完成查詢動作。二為拉式廣播系統(Pull-based Broadcasting)[4,11,20,22,25]；又稱請求式廣播(On-demand Broadcast)。當所需求的資料皆不在廣播頻道上廣播時，行動用戶端便會利用上傳頻道(uplink channel)向伺服器端提出請求，伺服器端根據請求將所請求資料經由次要頻道(sub channel)廣播給行動用戶端。

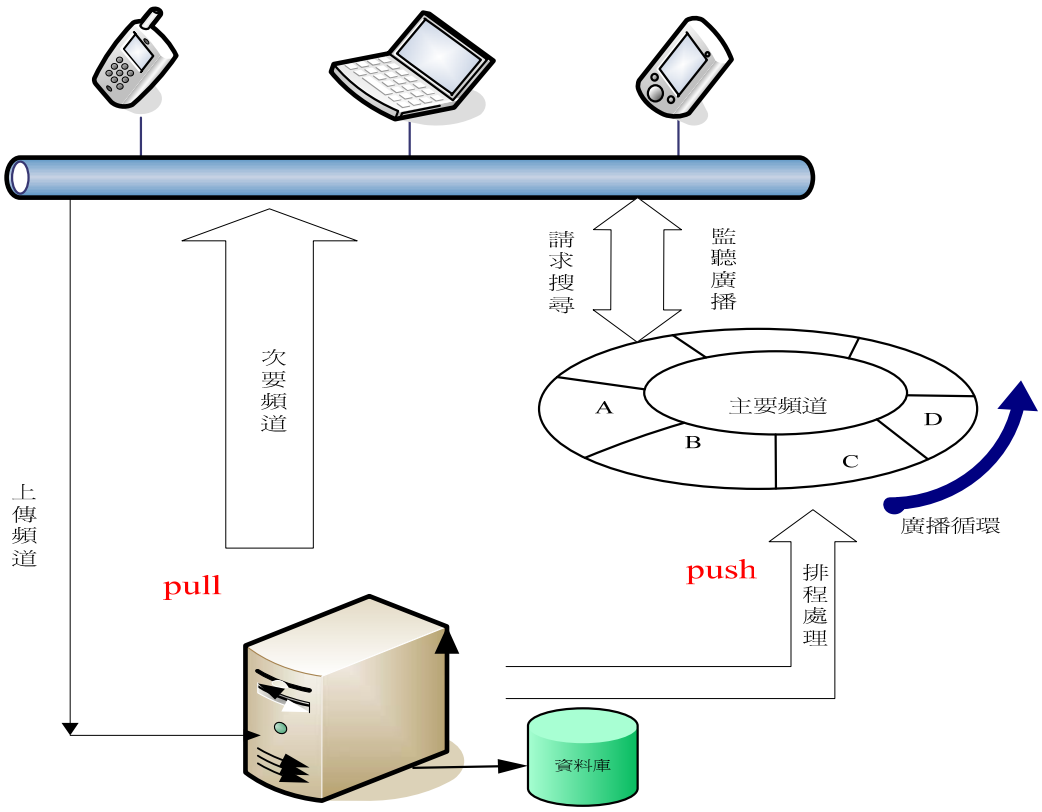


圖 3、資料廣播系統示意圖

另一種不需用無線基地台的對等式無線區域網路(Ad-Hoc Network)，使用的是行動式基地台(mobile station, MS)，因此之間的連結則必須靠每一個行動用戶端 MH(mobile hot)做為路徑的選擇，所以沒有固定的拓撲。當來源 MH 與目的 MH 間距離太遙遠時，就經由多個 MH 之間以多次跳躍(multi-hop)來傳送資訊到達目的 MH。此模式下的繞送策略可分為繞徑表導向(Table- Driven)、需求導向(On-Demand)和混合式(Hybrid)三種。

以 Table-Driven 繞送方法而言，各個 MH 本身都存有將繞送資訊的路由表(Routing Table)，並週期性固定和鄰近 MH 更新維護整個網路環境最近的繞送資訊。若 MH 有資料請求時，便可馬上依據 Routing Table 中的路徑和資料訊息來做廣播。但缺點是 MH 必需要不斷的去交換 Routing Table 的訊息，會耗用掉大量的網路頻寬和 MH 的電力。

On-Demand 繞送策略下，MH 並不會事先建立 Routing Table，而是當 MH 有資料請求時，來源 MH 發送尋找目端 MH 路徑的封包(package)，中間的 MH 收到此封包時，會記下回來源 MH 的路徑，直到目的 MH 收到封包時，依所記錄之路徑回另一訊息封包給來源 MH，因而建立起傳送資訊路徑。其缺點會增加資訊傳送時間上的延遲，所以此方法較適合於小規模的網路。Hybrid 式繞送方法，便是結合 Table-Driven 與 On-Demand。

2.2 圖形理論

圖形理論的討論，回溯於 1736 年[1,2]，由瑞士數學家尤拉（Leonhard Euler, 1707-1783）利用圖形來解決 Koenigsberg 七橋樑問題。探討由點及線所構成的結構，即是任何一條線，兩邊一定有點。研究的重點在於點與線的連結關係，而不是距離或者方向的問題。尤拉定義與節點所連接的邊之個數為邊分支度，從某一節點出發，經由各個邊後再回到出發的節點。從此圖形理論便廣泛地應用在各個方面，包含最短路徑、拓撲排序、專案規劃等。

運用在資料結構時，圖形結構並沒有像樹結構有著階層關係，其每個節點之間皆可以相互連結。基本定義為，圖形 ***G=(V,E)***，是由 ***V*** 是點(vertices)所構成有限非空集合，兩節點間的連線 ***E*** 是邊(edges) 所構成有限非空集合來表示，連結於節點 u 上的邊數稱為 u 的內分支度(degree)，其總邊數稱為 u 的總入分支度(In_degree); 節點 u 向外連結的總邊數稱為 u 的向外總分支度(Out_degree)。主要可以分成有向圖形結構 (Directed Graph)和無向圖形結構(Undirected Graph)二種。本研究則著重於有向無循環圖形上的探討，藉由有向無循環圖形的資料存取模式及其表示順序限制作為基礎。

2.2.1 有向圖形結構(Directed Graph)

所謂有向圖形意指在圖形 G 中任一節點上的邊皆有方向性。如每一個邊線(edge)皆有箭號標示著方向表示點和點間的順序性，其限制為圖形裡不能有相同指向同一目的的連結、圖形裡不能有指向自己邊的連結。排除這兩點限制後，圖形的有向邊構成一個循環，稱之為有向循環圖形 $G=(V,E)$，$(u,v)\in E$ 且 u≠v，表示圖形上最少有一條路徑(Path)，節點 v 到節點 u 是屬於圖形上的邊 E (P_{vu})，如圖 4a。反之稱為有向無循環圖形 $G=(V,E)$[18]，如圖 4b 是個簡單的有向無循環圖形。其點(vertices)的集合和邊(edges)的集合，如下所示:

$V(G) = \{v_1, v_2, v_3, v_4\}$

$E(G) = \{<v_1, v_2>, <v_1, v_4>, <v_2, v_4> <v_3, v_4>\}$

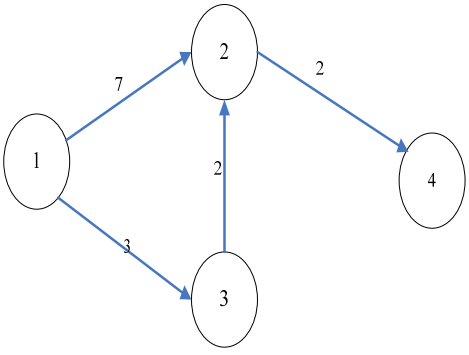


圖 4a、簡單有向循環圖形

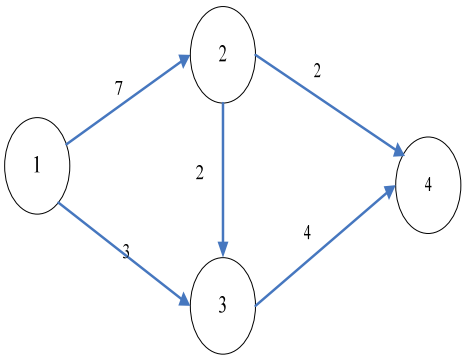


圖 4b、簡單有向無循環圖形(DAG)

2.2.2 無向圖形結構(Undirected Graph)

無向圖形意指在圖形中 *G=(V,E)*任兩節點間的邊無方向性。其限制是兩節點間不能以兩線相連。圖 5a 其節點 3 沒有直接連結到節點 4;節點 2 也沒有直接連結到節點 3 等以上條件，因此無法形成迴路，為無向無連通圖形。圖 5 b 在圖形裡任兩節點上都存在一個邊，並且形成一個迴路，為無向完全連通圖形。

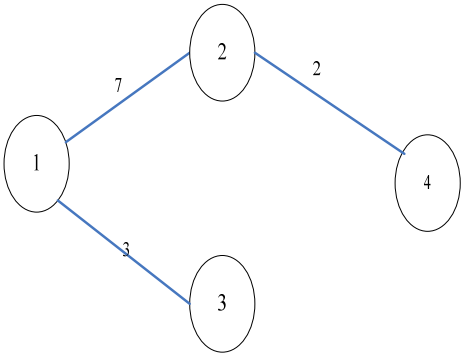


圖 5a、簡單無向無連通圖形

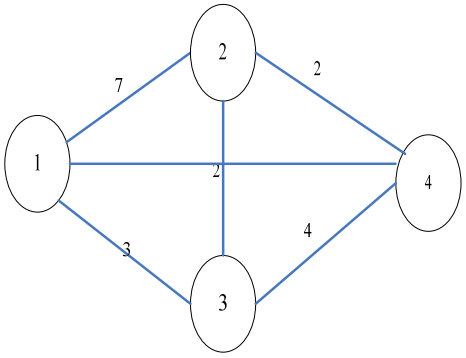


圖 5b、簡單無向完全連通圖形

2.3 混合式演算法(Hybrid heuristic)

由學者 R. Sakellariou 與 H. Zhao[23]在 2004 年所提出運用在有向無循環圖形裡的一個混合式演算法，主要是解決連續性獨立作業的排程問題。在有向無循環圖形上利用各個不同的方法去計算節點和邊的權重值，所產生出的序列平均讀取長度比起以往的方法較有明顯的改善。其演算法步驟如下:

步驟一:假設在圖形中的每個節點和邊都有權重值，藉由這些權重值計算出每個節點的階層值。經由本研究修改後的公式如圖公式(1)所示，節點 u 的階層值為 r[u]:

$$r[u] = \max_{(u,v)\in E}\{W[u][v] + r[v]\} \qquad \text{公式(1)}$$

W[u][v]是從節點 u 到節點 v 的邊權重值，節點 u 和 v 必需是屬有向無循環圖形上的節點。

步驟二:將其階層值由大排到小放入陣列中。學者提出由此陣列中取出第一個節點加入第 0 個群集(group)，接著再依序和陣列中的節點比對彼此關係是否為獨立。如果彼此之間為獨立關係就加入相同的群集裡。反之節點彼此間是有關聯的就新增另一個群集，把此節點放入新的群集中，並重覆比對的動作，會得到一個或一個以上的群集。

本研究藉由所產生的群集，於每個群集裡的節點運用貪婪拓撲排序，便產生新的序列。再藉由此序列計算出節點間的平均路徑長度，並與多種貪婪拓撲排序策略相互比較其優劣。

2.4 拓撲排序(Topological Sort)

利用圖形結構呈現事件的前後次序，按照固定次序，在完成某項或某幾項事件後才進行下一個事件，任兩項事件無法同時進行。在圖形 G 中之節點代表工作或一個事件，邊則代表事件之間的先後關係，例如瀏覽特定購物網站時必須成為會員，才能購物(加入會員->登入->進行購物->結帳)，這所形成的圖形 G 稱為頂點作業網路(Activity On Vertex Network)或稱 AOV 網路。

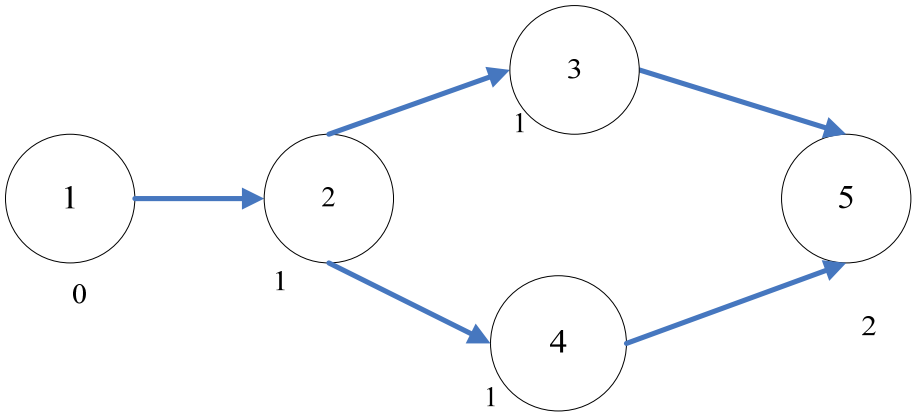


圖 6、AOV 網路圖例

如圖 6 是一個 AOV 網路，其節點代表網頁。節點 3 需等節點 1 和 2 完成加入會員和登入賣場後才可以進行購物的動作。節點 1 為其它點的先行者，節點 2 為節點 3 和 4 的立即先行者。節點 5，必需等節點 3 和節點 4 完成後才能進行，為圖形中的後繼者。

在無循環圖形中，若存在一條路徑由節點 v 至節點 u，則稱 v 為 u 的直接前行點，而 u 是為 v 的直接後繼點，必須等待節點 v 事件完成後方可進行節點 u 事件。因此在線性序列中節點 v 會排在節點 u 的前面，這種具有線性序列特性的排序方法稱謂拓撲排序[17,19]。在此圖形需先計算出節點的內分支度，其拓撲排序方法如下:

步驟一:在圖形中尋找總入分支度為零的節點，即為沒有前行者的節點。

步驟二:在這些無前行者的節點裡，挑選出一任意節點，再將其連接的相依邊皆刪除。

重覆此程序，直到所有的節點皆被挑選出為止。拓撲排序必需存在於無循環圖形中，所產生的拓撲排列會有一個或一個以上的解。

如上圖 6 所示，進行拓撲排序如下:

1. 找出所有總入分支度為零的節點 1
2. 挑選出節點 1 放入序列中，並將其連接的邊皆刪去
3. 在剩下的節點中，找出總入分支度為零的節點 2
4. 挑選出節點 2 放入序列中，前將連接的邊皆刪去
5. 在剩下的節點中，找出沒有總入分支度為零的節點 3 和節點 4
6. 任意挑選其中一節點 3 放入序列中，前將連接的邊皆刪去

最後把節點 4、節點 5 也放入序列中，拓撲線性排列為:(1)(2)(3)(4)(5)。其也有可能為:(1)(2)(4)(3)(5)。因此有可能會繞進較不佳的序列解。

針對有向無循環圖形，其圖形結構的走訪概念和樹的走訪概念相近，皆由某個頂點開始，經過一定的規則拜訪圖形中每個節點。圖形走訪的方法分為深度優先搜尋法(Depth First Search)和廣度優先搜尋法(Breadth First Search)兩種[1,2]。

本文則以關連性資料項為考量，因為運用有向無循環圖形的資料存取模式及其表示順序的限制作為基礎，所以選用圖形走訪中的深度優先法之特性運算出各個節點的階層值，利用各種不同的貪婪拓撲排序的機制，經由各個策略下所產生的有向性線性排序，運算其頂點之間的平均路徑長度，並且相互比較其優劣。

2.4.1 深度優先搜尋法(Depth First Search)

DFS 是使用遞迴的技巧、 stack 的概念，縱向優先的走訪，即是往路徑方向走訪。首先找尋點在結構中的鄰接點，令任一節點 v 為起點，在與節點 v 所有相鄰而未被走訪過的節點 u_1、u_2、u_3、....等 n 個節點中，以迴圈方式走訪。選擇其中一個節點 u_1 為新的起點。接著 u_1 以遞迴的執

行 DFS，當 u_1 當有未走訪過的節點時，由 u_1 的所有鄰接節點中再度以遞迴方式的執行；當此節點 u_1 的所有相鄰節點都已經被拜訪過，則退回到節點 v，針對與該節點相鄰且尚未被拜訪過的頂點執行 DFS。直到所有相鄰節點都拜訪過後，回溯到上一層繼續深度優先搜尋其他未被拜訪過的頂點。其主要演算法如下:

Algorithm *DFS-visited(u)*

Input: A graph G= (V, E) and a vertex v= {0, 1, 2, 3… n-1}

Output: labeling of the vertex of G as discovery in the connected component

```
1.      For all u∈V set Label
2.          visited(u) ← false
3.       For each i∈V do
4.              If vertex( u) is unvisited
5.          Then
6.              DFS(u)
```

Algorithm *DFS(u)*

Input: A graph G= (V, E) and a vertex v= {0, 1, 2, 3… n-1}

Output: labeling of the vertex of G in the connected component of u as discovery edges set Label (v, visited)

```
1.    visited(v) ← true
2.        output v
3.            For each vertex w adjacent to v
4.                If visited(w) = false
5.            Then
6.                DFS(w)
```

圖 7、DFS 演算法

此走訪搜尋法運用在有向無循環圖形 6 裡，並依據 DFS 走訪的特性，運用由學者 R. Sakellariou 與 H. Zhao[23]提出的混合式演算法中提到的階層值，試圖在走訪過程中，一併計算出每個節點的階層值。其演算法步驟如下：

1. 首先把所有節點的階層值設定為零並且準備一個鏈結陣列
2. 選出總入分支度為零的節點 1
3. 按箭頭順序縱向走訪往節點 2

4. 節點 2 有兩條向外分支度，任選其中一條節點 4 走訪

5. 接著按箭頭順序縱向走訪到節點 5，因為節點 5 沒有向外分支度，所以將節點 5 放入陣列中並標記為『已拜訪過』節點，並計算出節點 5 的階層值

6. 回到節點 5 的上一層節點 4，因為節點 4 沒有向外分支度，所以將節點 4 放入陣列中並標記為『已拜訪過』節點，並計算出節點 4 的階層值

7. 回到節點 4 的上一層節點 2，選擇另一方向節點 3 並且以縱向走訪往節點 5，因為節點 5 為『已拜訪過』節點，且節點 3 沒有其他向外分支度，所以將節點 3 放入陣列中並標記為『已拜訪過』節點，並計算出節點 3 的階層值

8. 回到節點 3 的上一層節點 2，因為節點 2 沒有其他向外分支度，所以將節點 2 放入陣列中並標記為『已拜訪過』節點，並計算出節點 2 的階層值

9. 回到節點 2 的上一層節點 1，因為節點 1 沒有其他向外分支度，所以將節點 1 放入陣列中並標記為『已拜訪過』節點，並計算出節點 1 的階層值

2.4.2 廣度優先搜尋法(Breadth First Search)

BFS 是 level-by-level，在此使用佇列及遞迴的概念。優先找尋點串列中的所有鄰接點。首先，任選以一節點 v 為起點開始走訪，接續走訪節點 v 相鄰但未走訪過的所有鄰接節點包含有 u_1、u_2、u_3、等數個節點，當此階層相鄰節點拜訪完畢，方才繼續拜訪下一階層的節點，直到所有的節點皆被走訪過。

本文根據零入分支度排序(zero in-degree sort)走訪的特性。首先設定一個 degree 矩陣，存放每個節點的前行者個數，因此在矩陣內為 0 的表示沒有任何前者行節點。挑出沒有前行者節點 i 標記為拜訪過的節點，並放入序列中，將其相鄰於 i 的有向邊刪除且重新計算節點 degree 數。接著從與節點 i 相鄰且未被拜訪過的節點中，選擇節點 j，並以節點 j 為起點重覆執行遞迴 BFS，直到所有的節點皆被拜訪過且被排入序列中。

在節點 i 至節點 j 的路徑上有一邊權重值，令節點 j 有入射總權重值，公式如下:

$$S_{in}[i] = \sum_{(j,i)\in E} W[j][i] \qquad \text{公式(2)}$$

依據上述公式(2)運算在序列中的每個節點，依順序計算每個節

點的 $S_{in}[\mu]$。

將走訪搜尋法運用在有向無循環圖形 6 裡。其演算法步驟如下:

1. 選出總入分支度為零的節點 1，將節點 1 標記為『已拜訪過』節點並往下一串列鄰接點節點 2 走訪
2. 將節點 2 標記為『已拜訪過』節點並往下一串列鄰接點節點 3 和節點 4 走訪
3. 將節點 3 和節點 4 標記為『已拜訪過』節點並往下一串列鄰接點節點 5 走訪

第三章 問題描述

整個無線網路的廣播環境中，為確保用戶端的聯網品質(QoS)，如何運用較佳的資料排程進行廣播資訊，以降低網頁資訊讀取的平均等待查詢時間，使網頁資訊的傳輸更有效率，以解決頻寬受限的狀況，因此資料排程等問題已成為學者們探討之議題。

本研究所討論的資料廣播排程問題，於用戶端所要求的資料項有存取順序的限制，針對前述問題進行改善。當如果遇到循環圖形時，則將造成有向循環圖形，因限於本研究所探討之有向無循環圖形，所以將此循環圖中各個的有向圖邊皆減掉循環圖中最小的權重值，以達到有向無循環的圖形。本章節將詳細介紹其資料廣播排程。

3.1 相關性資料廣播排程

本節著重於相關性資料廣播排程，行動用戶端所要求的資料項存取是有順序性限制。如:用戶端學生欲使用曠課查詢系統，查詢個人曠課情況，必須先進入系統登入網頁輸入個人資料後，才進行查詢各週的曠課狀況。

假設廣播頻道上有一組資料項正在廣播，包含 N 個資料項 $D=\{d_1,$

$d_2,d_3,\ldots,d_n\}$。行動用戶端對頻道的查詢為 q_i ，其 A 用戶端所要查詢為 $q_1=[d_2,d_3]$。以圖 8(a)廣播排程為 $D=\{d_1,d_3,d_2,d_4,d_5,d_6\}$，如第一筆查詢從目前廣播週期由 d_3 開始查詢，因為存取的限制先讀取到 d_2 ，則必需等待下個週期才能收到 d_3 的時間為 7 個資料項時間。

廣播頻道上每個資料項視為節點，資料項查詢的頻率視為資料項間的相依性，將其對應成有向無循環圖形上的有向權重邊值。運用本研究所提出的排序機制將相依程度較高資料項 d_1 和 d_2 集中一起，如圖 8(b)其排程為 $D=\{d_1,d_2,d_3,d_4,d_5,d_6\}$。可得知此廣播排程下，查詢的時間為 2 個資料項時間。因此在廣播排程中，發現存取時間明顯降低了 5 個資料項時間。

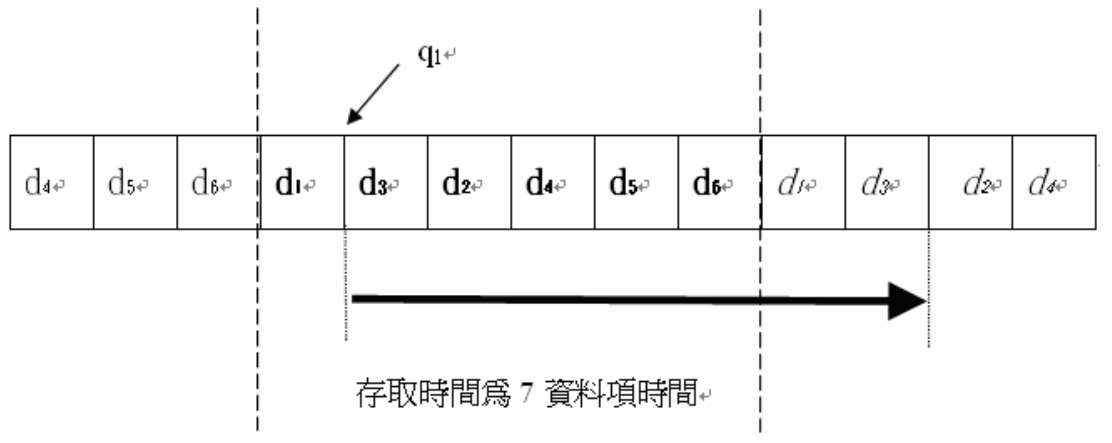


圖 8(a)、原廣播排程之讀取時間

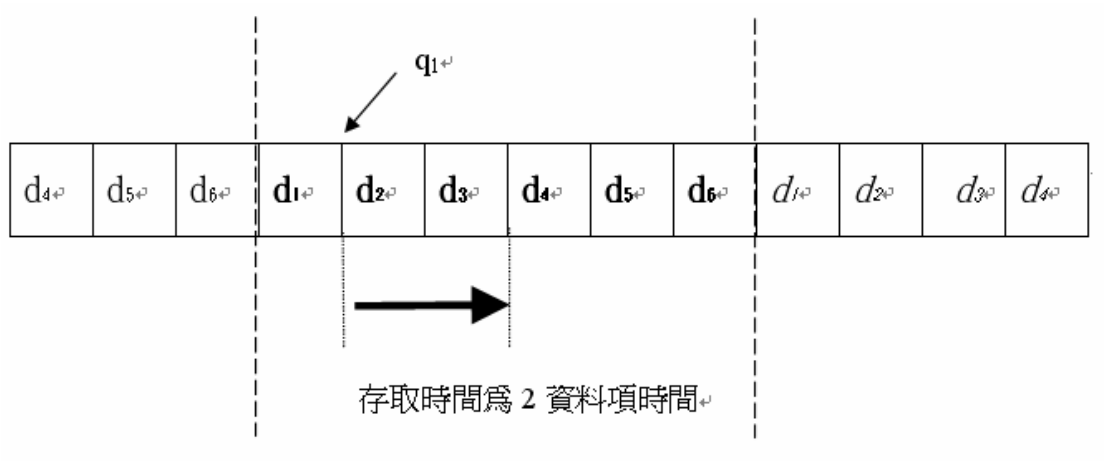


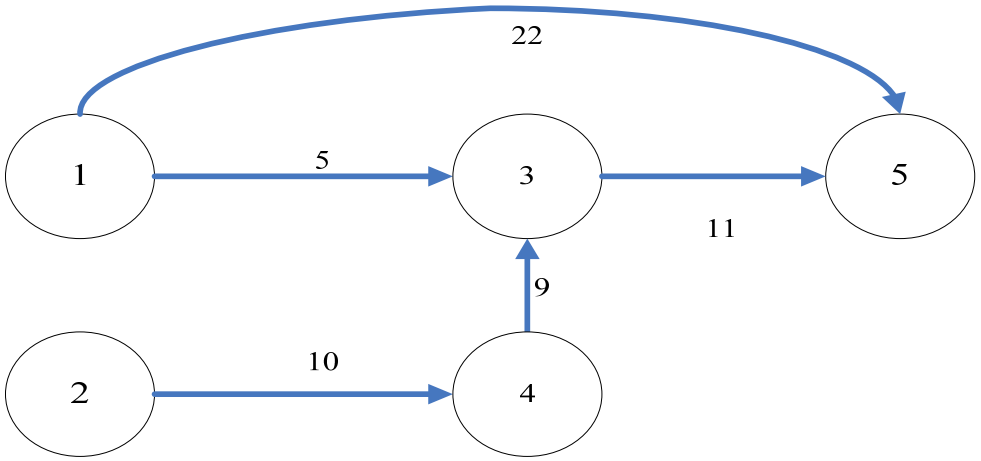
圖 8(b)、修改後廣播排程之讀取時間

3.2 有向線性排序的平均讀取時間

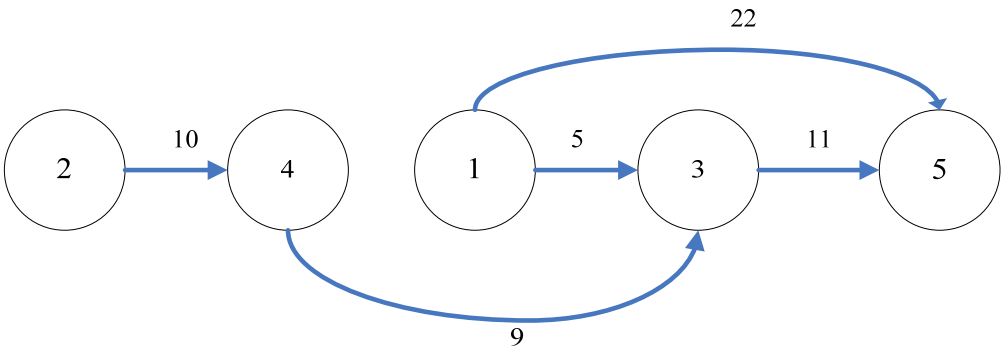
在無循環之有向圖形 G 中，節點代表工作或一個事件，邊則代表事件間的先後關係。如果有一路徑節點 i 到節點 j，則在序列中節點 i 就會排在節點 j 之前，其圖形裡邊的權重為一個恆正的常數值[3]。藉由排序後將有向無循環圖形轉成有向性之線性排序[7,24]，以利於運算。計算用戶端平均查詢讀取時間[16,27]如公式(3)所示，假設產生之廣播序列為 L, 在序列 L 中的第 i 個節點位置到相鄰的第 j 個節點位置之權重值為 *W[L[i]][L[j]]*; j-i 為收取資料項 i 到資料項 j 所花費的時間。

$$\sum_{j>i}(j-i)W[L[i]][L[j]] \Big/ \sum_{j>i}W[L[i]][L[j]] \qquad \text{公式(3)}$$

如圖 11(a)是一簡單的有向無循環圖，運用拓撲排序後產生一序列 L，其分別位於序列 L(1)= 2，L(2)=4，L(3)=1，L(4)=3 ，L (5)=5。便可套用上述公式算出平均讀取時間為: 1.54385965(10+9*2+5+22*2+11/57 = 88/57)。



(a)



(b)

圖 9、(a)簡單有向無循環圖範例一與(b)對應的拓撲排序圖

假設網頁資料項被要求的頻率為有向邊的權重，其權重值表示資料項之間相依程度。因此序列裡將相依程度較高者集中一起。便能縮短資料項之間的平均長度，以降低網頁資訊讀取的平均等待查詢時間。

第四章 雙向演算法

於無循環之有向圖形 G 中，拓撲排序雖然能有效利用於 DFS 排序與 BFS 之零入分支度排序。但在有向圖形應用於 DFS 排序時，無法找尋所有可能的節點排序，因而會減弱尋找最佳路徑的方向。為了解決相依性資料項廣播排程問題，本研究加入貪婪拓樸排序的各種策略與叢集各策略演算法，應用 DAG 以解決廣播排程的問題。

4.1 貪婪性演算法

貪婪演算法之特色，一種由上至下(Top-Down) 階段性尋找最佳解的方法，即在面臨選擇時，都以眼前最有利的條件為主。不斷的改進其方向，直到無法改進為止，因而這些解都不會違反路徑限制式，所以每次建構出的解都一定是區域性最佳解。此貪婪演算法常用來解決最佳化問題。例如櫃檯服務問題，某一個銀行出納櫃檯要服務 n 位顧客，銀行的目標是希望顧客在櫃檯等待的平均時間要最少，其解決方法為每次都從尚未服務的客戶中，選擇需要服務時間為最短的顧客服務，如此就可達到預期目標。

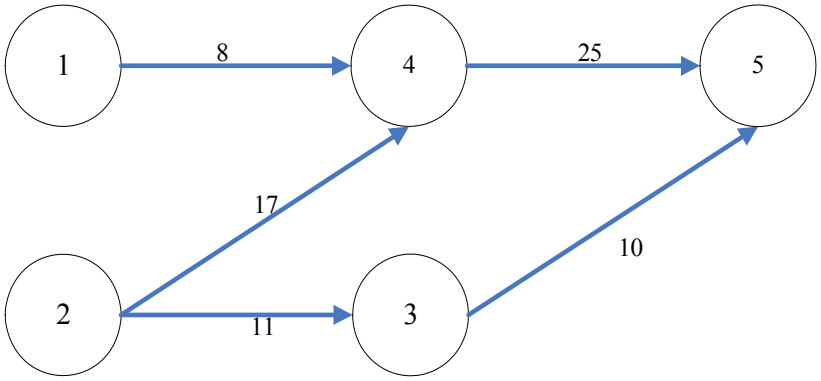


圖 10、有向無循環圖形(DAG)範例二

4.1.1 貪婪性零入分支度排序演算法(最大入射權重值選擇)

依據有向性線性排程序列上，每個序列中節點資料項的邊權重值並不會延伸到下一個節點資料項的邊權重值。在採用貪婪演算法面臨選擇時，則會選擇節點中較大的入射總權重值，因而在序列裡節點的入射權重值不會延伸到往後節點。

當加入應用零入分支度排序的貪婪演算法面臨選擇時，在總入分支度為零，即無前行者的節點中，則必須選擇節點的入射總權重值為最大者。其主要步驟如下:

步驟一: 在 DAG 中，尋找出所有總入分支度為零的節點，In_degree

＝0。

步驟二:由這些節點中挑選出入射總權重值為最大者節點 i，並放入廣播序列 L 中。

步驟三:對相依於節點 i 所有的入射權重邊刪去，並重新計算 In_degree 值。

重覆上述步驟，直到所有的節點都拜訪過為止，即可得到符合較佳的廣播拓撲序列解。以上圖 10 為例，使用貪婪性零入分支度排序演算法，其過程如下:

步驟一: 如圖 11，節點上方為節點 In_degree 值;有斜線標示的節點為 In_degree 為零者。首先計算出每個節點的入射總權重值，可得到 $S_{in}[1]=0$， $S_{in}[2]=0$， $S_{in}[3]=11$，$S_{in}[4]=8+17=25$，$S_{in}[5]=25+10=35$。可找到節點 1 和 2 的 In_degree 值同為零。

由節點 1、2，比較其入射總權重值。此時入射總權重值同為零，將從節點 1 與 2 中擇其一放入廣播序列，此處選擇節點 2。

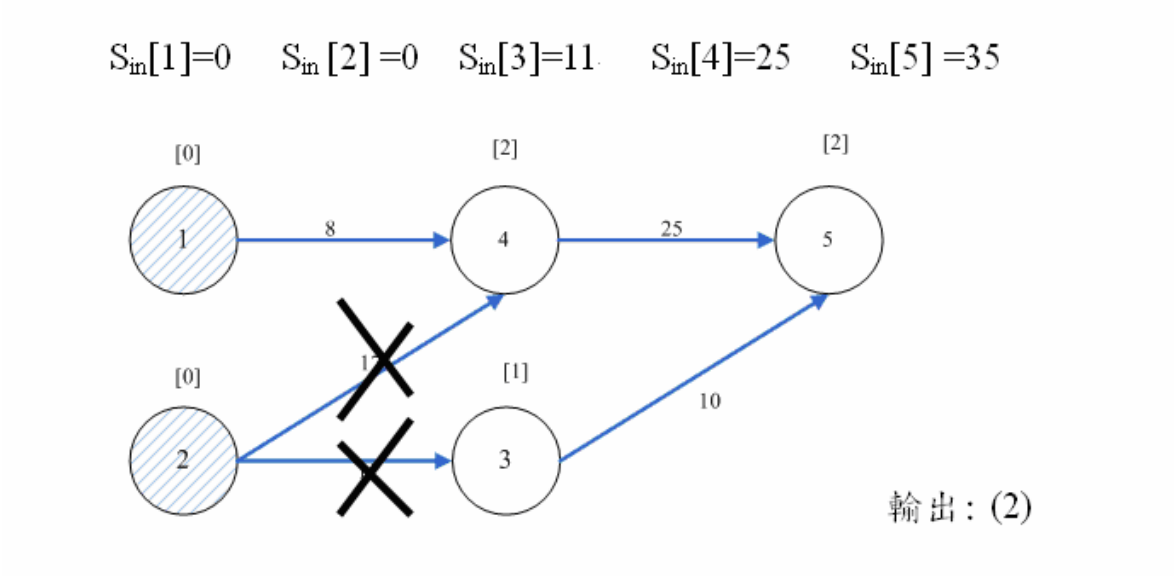


圖 11、範例二運用於最大入射權重值選擇，刪除節點 2

步驟二:把相依於節點 2 所有的入射權重邊都刪去，並重新計算 In_degree 值。節點 2 相依於節點 4 和 3，因此節點 3 的 In_degree 為 0、節點 4 的 in_degree 更新為 1。其中節點 1 和節點 3 的 In_degree 值為零，選擇入射總權重值較大者節點 3 放入廣播序列裡。

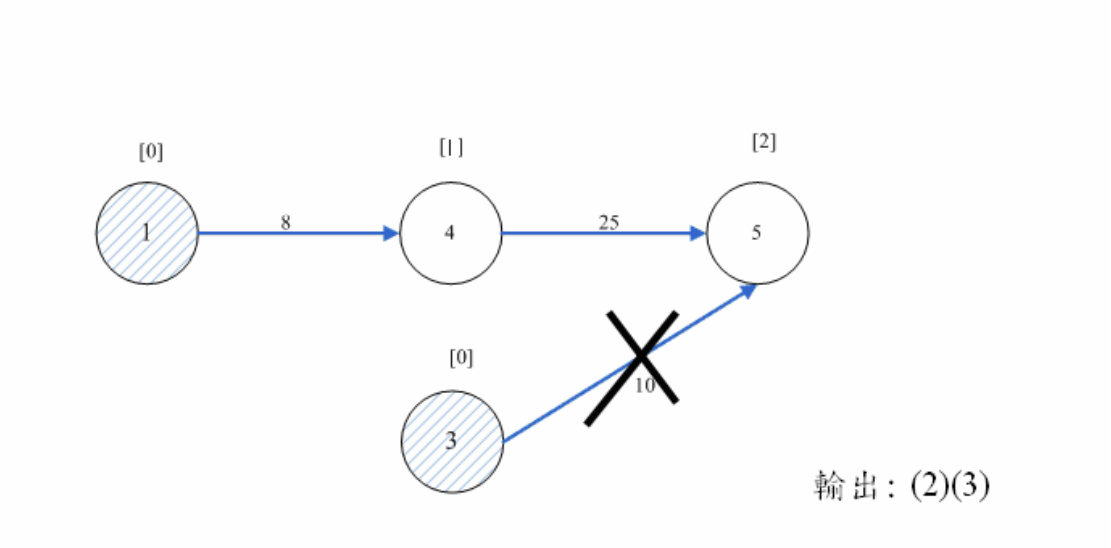


圖 12、範例二運用於最大入射權重值選擇，刪除節點 3

步驟三:把相依於節點 3 的有向入射邊刪除並重新計算 In_degree 值，更新節點 5 的 In_degree 值為 1。選擇 In_degree 值為零的節點 1 放入廣播序列中。

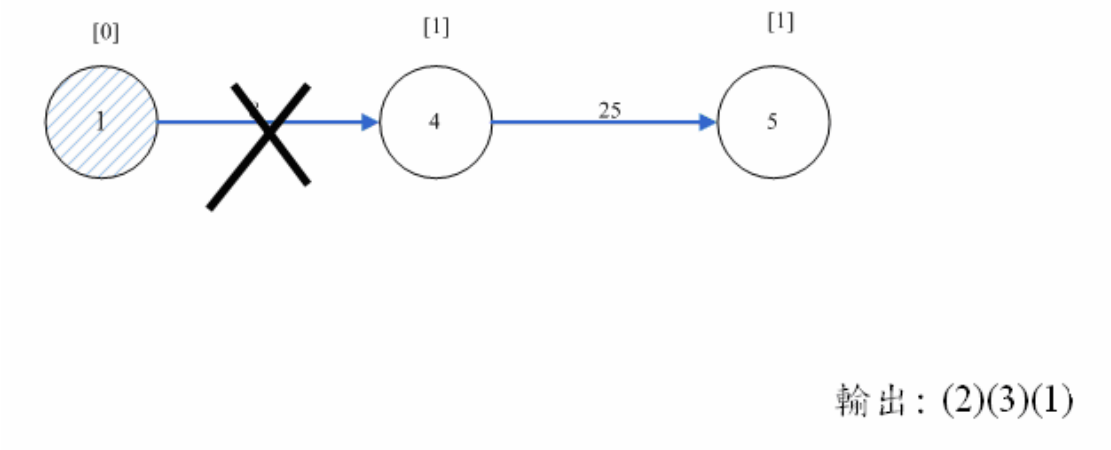


圖 13、範例二運用於最大入射權重值選擇，刪除節點 1

步驟四：刪除相依於節點 1 的入射權重邊，且更新節點 4 的 In_degree 為 0。選擇 In_degree 值為零的節點 4 放入廣播序列中。

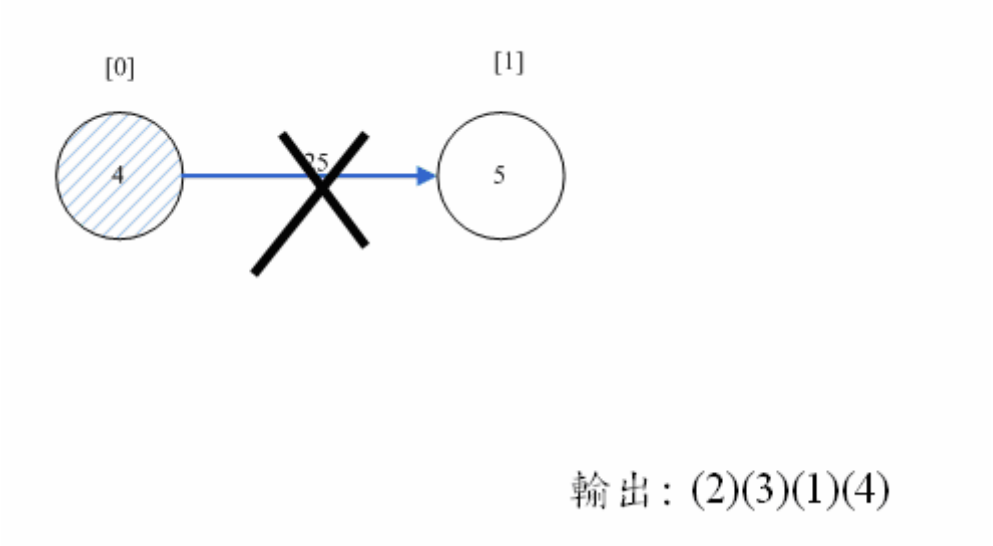


圖 14、範例二運用於最大入射權重值選擇，刪除節點 4

步驟五：刪除相依於節點 4 的所有入射權重邊，且更新節點 5 的 In_degree 為 0。選擇 In_degree 值為零的節點 5 放入廣播序列中，可以得到一廣播序列為：(2)(3)(1)(4)(5)。

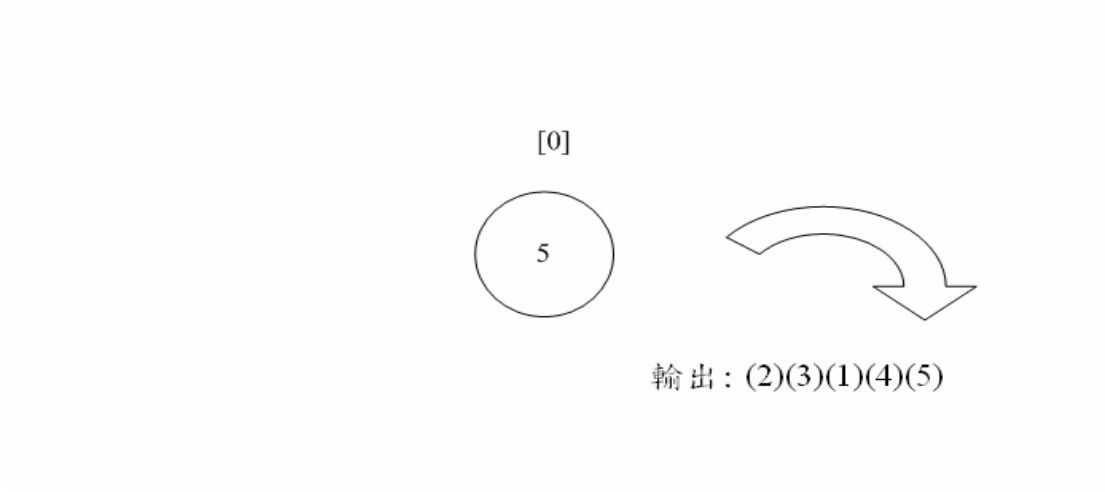


圖 15、範例二運用於最大入射權重值選擇，刪除節點 5

經由此廣播序列套用公式(3)，便可以得到此序列之平均讀取時間為 1.76056338(11+3*17+3*10+8+25/71=125/71)。

4.1.2 最小階層值選擇

應用加入零入分支度排序於婪貪演算法中，當所有的總入分支度為零時所得到的序列解，可能會繞進較不理想的解。因此在面臨選擇同時，不同於上一章節所述，取而代之的是由學者 R. Sakellariou 與 H. Zhao[23] 提出的混合式演算法中提到的階層值，在此選擇節點中階層值較小者。

當加入應用零入分支度排序的貪婪演算法面臨選擇時，所有總入分支度為零，即在無前行者的節點中，必須挑選節點中較小階層值。不同於上一章節所述之選擇最大入射權重值步驟如下，由圖 16 所示。

步驟一: 先計算出每個節點的 r[i]值，可得到 r[1]=33， r[2] =42，r[3]=10，r[4] =25，r[5] =0。可找到節點 1 和 2 的 In_degree 值同為零。

步驟二: 由節點 1 和 2 中挑選出 r[i]值為最小者節點 1，並放入廣播序列中。

步驟三: 對相依於節點 1 所有的入射權重邊刪去，並重新計算 In_degree 值。

重覆上述步驟，直到所有的節點都拜訪過為止，即可得到符合較佳的廣播拓撲序列解。其解為:(1)(2)(3)(4)(5)。

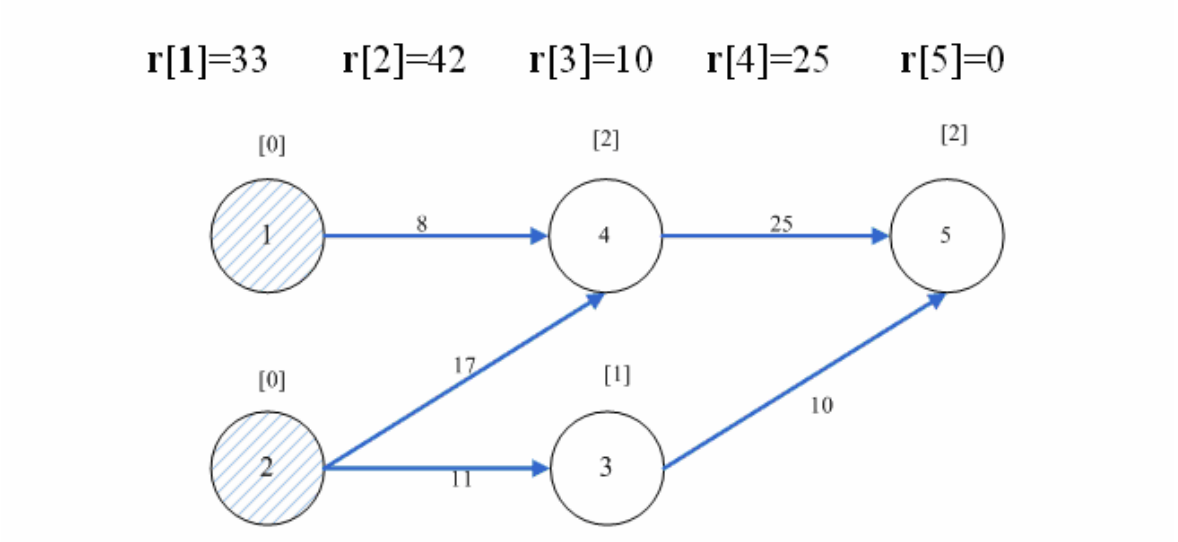


圖 16、範例二運用貪婪性零入分支度排序選擇最小階層值

經由此廣播序列套用公式 3，便可以得到此序列之平均讀取時間為

1.6056338(3*8+11+17*2+2*10+25/71=114/71)。

4.2 應用階層值於最大入射權重值之貪婪演算法

貪婪性零入分支度排序演算法雖然可以找出較佳的廣播拓撲序列，但當有兩個或兩個以上節點的 In_degree 值同時為零，且入射總權重值皆相同時，就有可能會出現一個以上的解，所得到的廣播序列就不一定是最佳路徑。因此在這一種情況出現時，可以利用每個節點的階層值，用來判斷加入廣播序列先後順序的條件。

當利用應用階層值加入貪婪演算法，有兩個或兩以以上節點的 In_degree 值同時為零，且入射總權重值也相同時，則必須選擇節點的階層值為最小者。其主要演算法分為以下步驟:

步驟一:在 DAG 中，尋找出所有總入分支度為零的節點 In_degree=0。由這些節點中挑選出入射總權重值為最大者，並放入廣播序列 L 中。當有二個或以上節點的入射總權重值為相同時，就比較其節點 r[u]大小。挑選 r[u]值較小者，並放入廣播序列 L 中。

步驟二:對其所挑選出節點之所有相依的入射權重邊刪去，並重新計算 In_degree 值。

重覆以上步驟，直到所有節點都放入廣播序列中為止，便可以得到其符合較佳的廣播拓撲序列解。由上圖 10 使用應用階層值加入貪婪演算法，方法如下:

步驟一:同上零入分支度排序的貪婪演算法，計算入射總權重值並找出 In_degree 值為零的節點。另外計算每個節點的 r[u]值，r[1]=33，r[2]=42，r[3]=10，r[4]=25，r[5]=0。

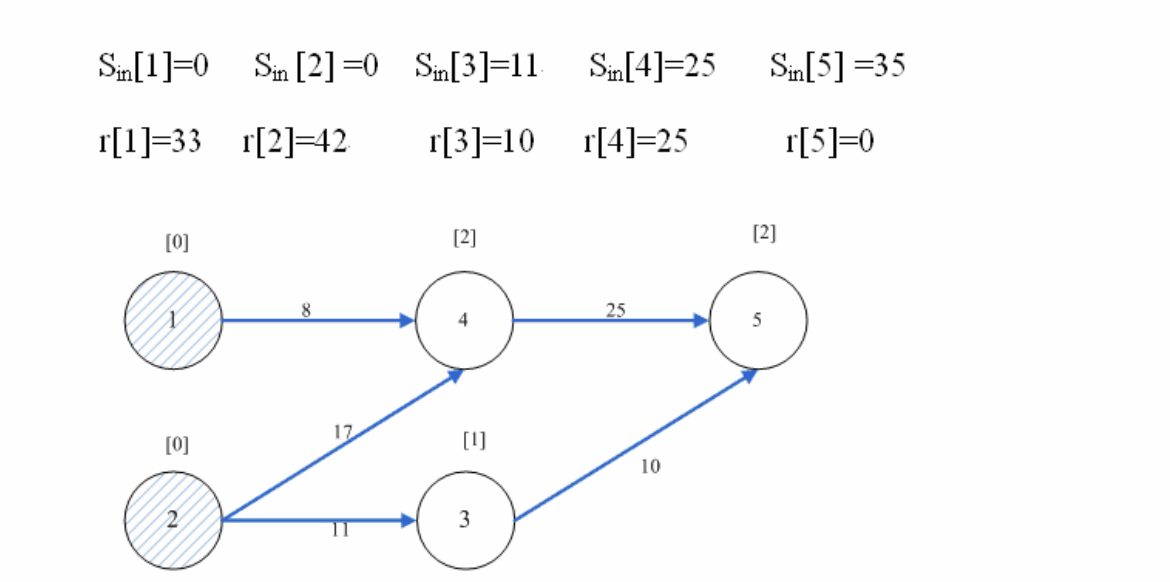


圖 17、範例二應用階層值於最大入射權重值之貪婪演算法

步驟二:其中節點 1 和節點 2 的入射總權重值同為零，因此比較其 r[u]值，挑選 r[u]值較小者節點 1，並放入廣播序列中。把相依於節點 1 所有的入射權重邊都刪去，並重新計算 In_degree 值，更新節點 4 的 In_degree 值為 1。

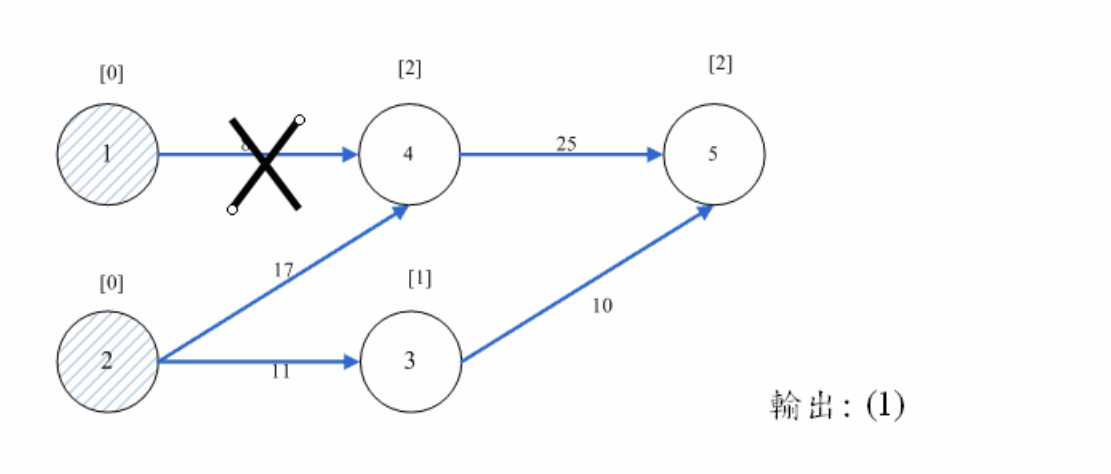


圖 18、範例二應用階層值於最大入射權重值之貪婪演算法，刪除節點 1

步驟三:接著從 In_degree 裡選出為零者，挑選節點 2 並放入廣播序列裡。然後在把與節點 2 相依的所有入射權重邊刪去，更新相依於節點 2 所有節點的 In_degree 值，節點 3 為 0、節點 4 為 0。

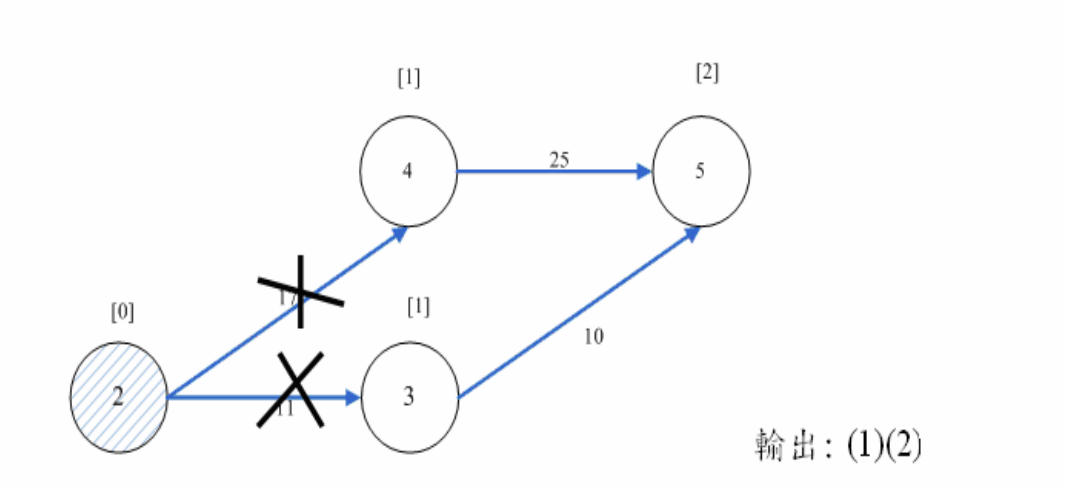


圖 19、範例二應用階層值於最大入射權重值之貪婪演算法，刪除節點 2

步驟四:此時節點 3 和節點 4 的 In_degree 值同時為 0，並比較其入射總權重值挑選其值較大者節點 4 放入廣播序列裡。刪除相依於節點 4 所有的入射權重邊，並重新計算其 In_degree 值，更新節點 5 的 In_degree 值為 1。

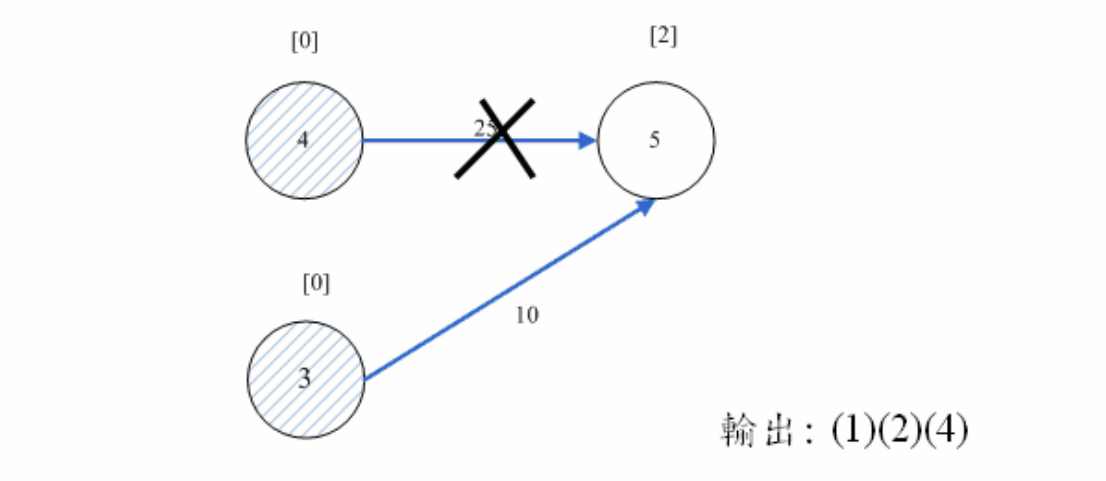


圖 20、範例二應用階層值於最大入射權重值之貪婪演算法，刪除節點 4

步驟五:選擇把 In_degree 值為零的節點 3 放入序列中。刪除節點 3 相依的所有入射權重邊，且更新節點 5 的 In_degree 值為 1。

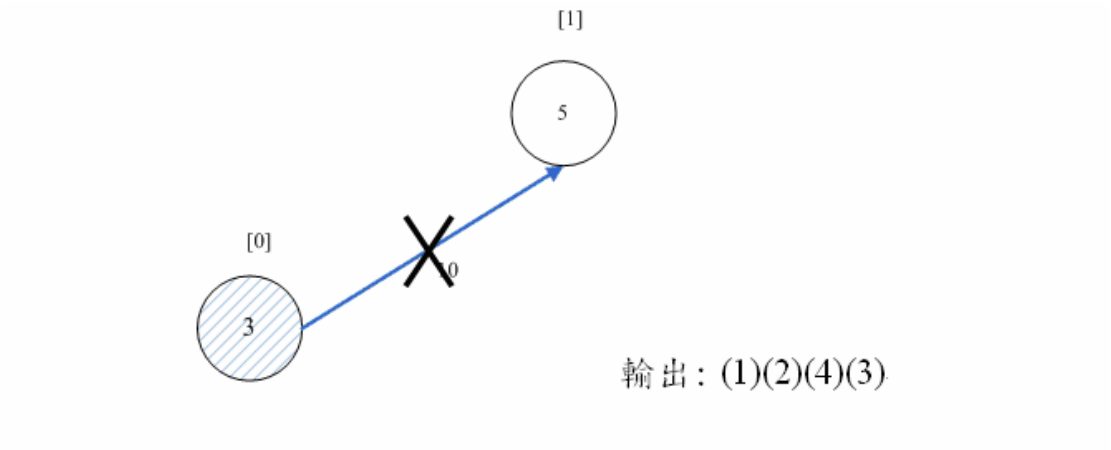


圖 21、範例二應用階層值於最大入射權重值之貪婪演算法，刪除節點 3

步驟六:最後把 In_degree 值為零的節點 5 也放入廣播序列中，便得到一廣播序列為:(1)(2)(4)(3)(5)。

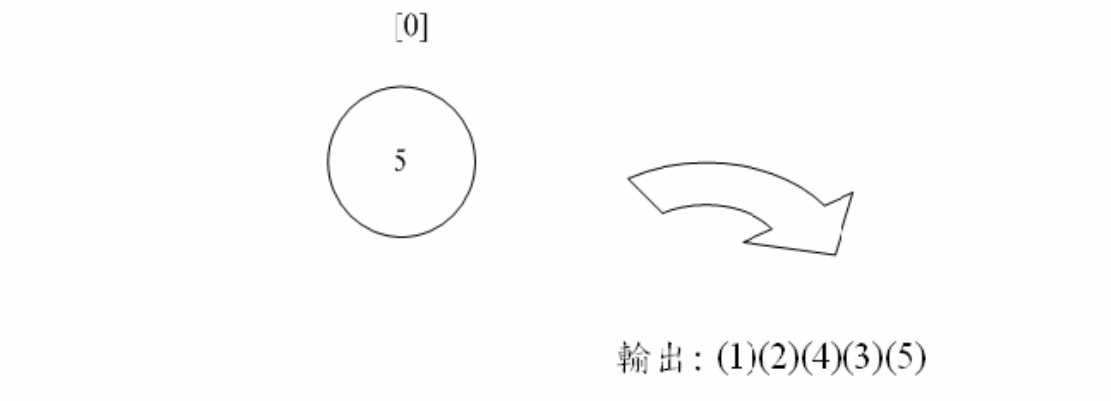


圖 22、範例二應用階層值於最大入射權重值之貪婪演算法，刪除節點 5

經由此廣播序列套用公式 3，便可以得到此序列之平均讀取時間為 1.61971831(2*8+17+2*11+2*25+10/71=115 /71)。

4.3 連續性雙向貪婪演算法

用於正向排序時主要在解決運用貪婪性零入分支度排序演算法時，有兩個或兩個以上節點的 In_degree 值同時為零，且入射總權重值皆相同時，就有可能會出現一個以上的解，所得到的廣播序列就不一定是最佳路徑。相同的在反向排序時也有此現象。因此在運用於反向排序後所得到的序列可以和正向排序後所得的廣播序列相互支持，用於 In_degree 值和入射總權重值同時為零時。

本研究提出另一個方式，建立 preference list 來解決存在多個 In_degree 為零且入射總權重相同的解時，無法找尋到最佳的廣播序列的問題。首先將圖 10 有向無循環圖形範例二反轉其中所有節點之有向邊，並重算其入射總權重值和階層值，其結果如下圖 22。利用應用階層值加入貪婪演算法將反向後的圖形依 4.2 節所敘述應用階層值於最大入射權重值之貪婪演算法之步驟排序後，可得到序列為: [5][4][3][2][1]，並將此序列存入 preference list。則 preference list 為: [5][4][3][2][1]。經由此廣播序列套用公式 3，便可以得到此序列之平均讀取時間為

1.6056338(25+11+2*10+2*17+3*8/71=114 /71)。

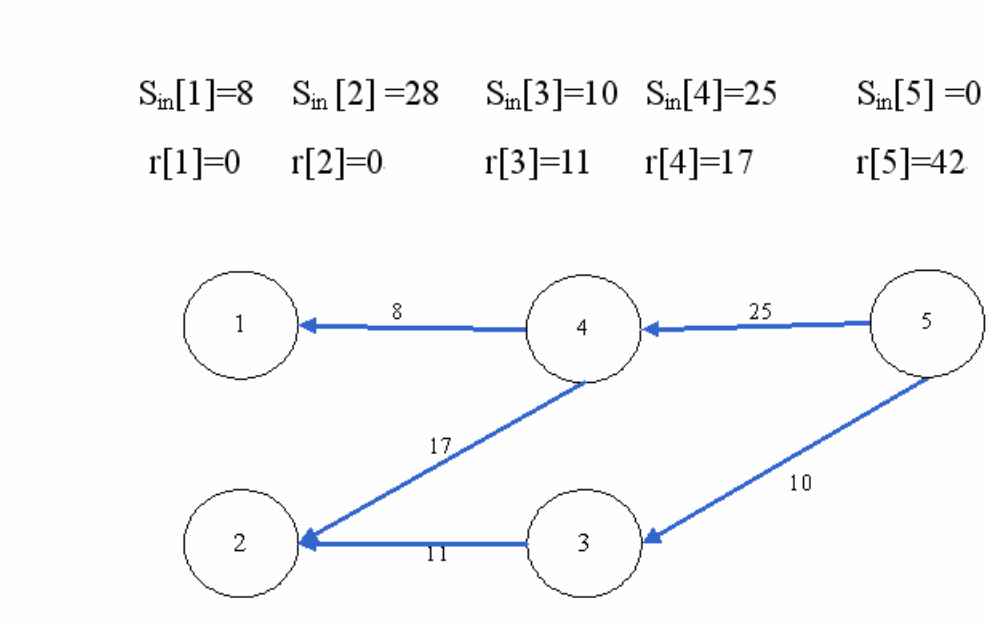


圖 23、反轉範例二有向無循環圖形(DAG)

其 preference list 可拿來再利用此演算法正向運作時參考。當有兩個或兩個以上節點的 In_degree 值同時為零，且入射總權重值也相同時，在選擇資料項加入廣播序列當下就可參考這 preference list。以 preference list 中節點順序為正向排列之基礎，則在 preference list 順序較後面者為優先排列。

以下圖 24 為 *BiSort* 雙向排序演算法。

Algorithm *Bidirectional-Greedy-Zero-In-Degree-Sort*

Input: A graph G= (V, E) and a vertex v= {0, 1, 2, 3… n-1}

Output: f^{opt}

1. f^{opt} ← random valid topological order list
2. G′← Reverse edge(G)
3. while true
4. pre$f$ ← f^{opt}
5. f = Sort(G, pre$f$)
6. If f = f^{opt} or W(f)>W(f^{opt}) then
7. return f^{opt}
8. f^{opt} ← f
9. pre$f$ ← reverse order(f^{opt})
10. f = Sort(G′, pre$f$)
11. f ← reverse order(f)
12. If f = f^{opt} or W(f)>W(f^{opt}) then
13. return f^{opt}
14. f^{opt} ← f

圖 24、*BiSort* 雙向排序演算法

主要不同於 4.1.3 應用階層值加入貪婪演算法步驟如下圖 25，其步驟二更改如下:

步驟二:，比較節點 1 和節點 2 入射總權重值，此時入射總權重同時為零，然而在 preference list 順序節點 1 排在節點 2 後面，則挑選節點 1，並放入廣播序列中。

其他步驟依此類推，直到所有點節皆排入廣播序列為止，其結果為:[1][2][4][3][5]。與反向廣播序列結果做比較，其兩次排序的作法相同，因架構上不同會得到不相同之廣播序列，然而這結果可以相互支持。分別計算出資料項之間的平均長度，求較短者為實驗結果樣本。

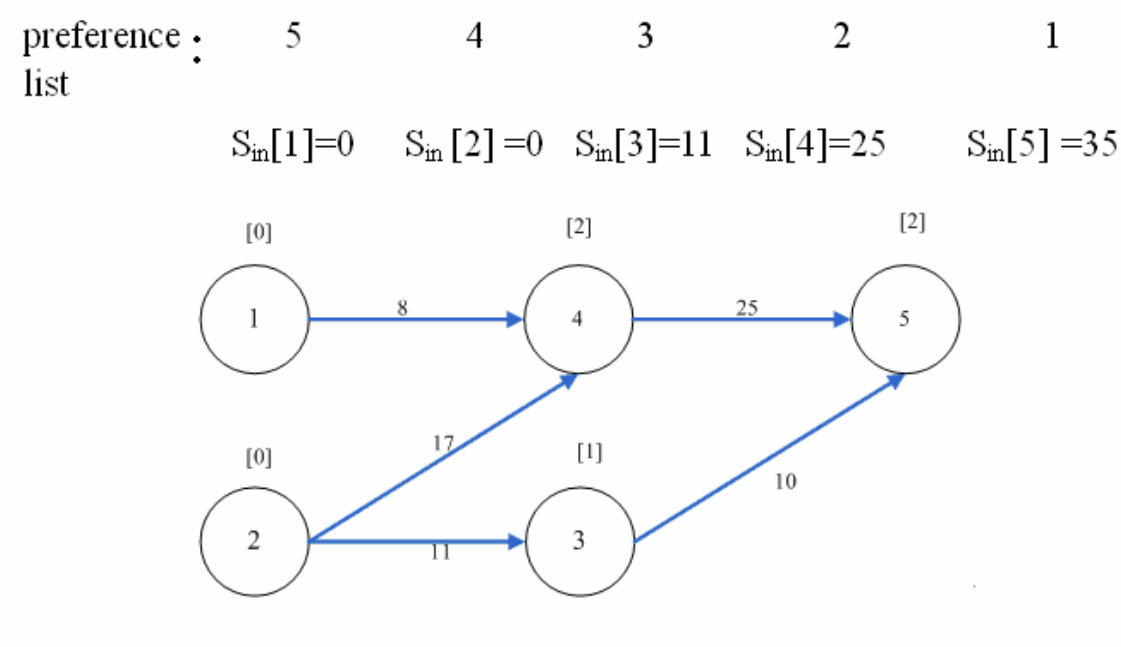


圖 25、範例二應用連續性雙向貪婪演算法

經由此廣播序列套用公式(3)，便可以得到此序列之平均讀取時間為

1.61971831(2*8+17+2*11+2*25+10/71=115 /71)。重複連續性雙向演算法尋找較佳解，當平均讀取時間較前次序列短者就存入並取代。因此本研究以較短的實驗數據為實驗結果樣本。由研究據數可得知圖 10 範例二有向無循環圖形其最佳路徑為:[1][2][3][4][5]。

4.4 有向性叢集演算法

本研究運用啟發式簡單直覺法(simple intuitive heuristic)假設每個節點為一個叢集(cluster)，連結兩鄰近、相似節點形成新叢集並符合有向性合併(DAG)，其分別為來源叢集和目的叢集。這兩節點在序列中，來源叢集必須是在目的叢集的前面。

在圖形中，每個節點都可成為來源叢集或目的叢集[9,15]，其主要條件為，從目的叢集到來源叢集不能存在任何的連結，並且也不能有任何路徑從目的叢集或者經由其他叢集到達來源叢集。否則在合併後將違反非循環式圖形的規則，所形成之拓撲排序將為無效。在沒有危及非循環式圖形的規則下任選兩個節點合併，首先要使用 DFS 或零入分支度排序來檢驗合併，是否符合非循環式圖形的規則。如果是符合者，就確定這兩節點適合於合併。

合併的過程中，假設每個節點都是單一叢集、邊為單一群組(group)

的邊，除了必須是要適於合併有向非循環式者並且要符合根據叢集被合併在一起的有向邊權重之大小。在所有適於合併之叢集中的群組有向權重邊要是最大者,合併方式為連結雙方的連結節點表[6]。

令來源叢集 Γ 到目的叢集 Δ 之有向邊權重聚集公式如下:

$$W_c(\Gamma,\Delta) = \frac{1}{|\Gamma||\Delta|}\sum_{u\in\Gamma,v\in\Delta} W_{u,v} \qquad \text{公式(3)}$$

藉由新合併產生的叢集修改來源叢集和移動目的叢集，並將鄰接來源叢集和目的叢集的邊也一併刪除。主要目的是要降低叢集的數量[8]。利用拓撲排序來檢驗是否適於合併時，假設有一叢集 x 是符合於合併要件其將此叢集 x 權重值放入 heap 中，並依序所有群組的有向邊權重值排序成為一個 heap，因此不適於合併之叢集會放在 heap 的最後，接著根據 heap 次序來做合併。主要處理步驟如下:

步驟一:根據 heap 中的第一個群組合併來源叢集/目的叢集

步驟二:從 heap 中移除第一個群組

步驟三:更新 heap

如圖 10 有向無循環圖形範例二，應用在有向性叢集演算法其步驟如下。任選所有叢集裡的二個叢集合併，一共會產生 10 個叢集並運用上述

公式 3，計算出每個叢集的有向邊權重值。在全部的合併叢集中，當來源叢集和目的叢集間不存在任何連結即叢集之有向邊無權重值，則在圖 28 中省略，並依序放入 heap 中。在圖中 26 裡，布林值為真(t)為適於合併之叢集反之布林值為假(f)為不適於合併之叢集，因而放在 heap 中的最後面。

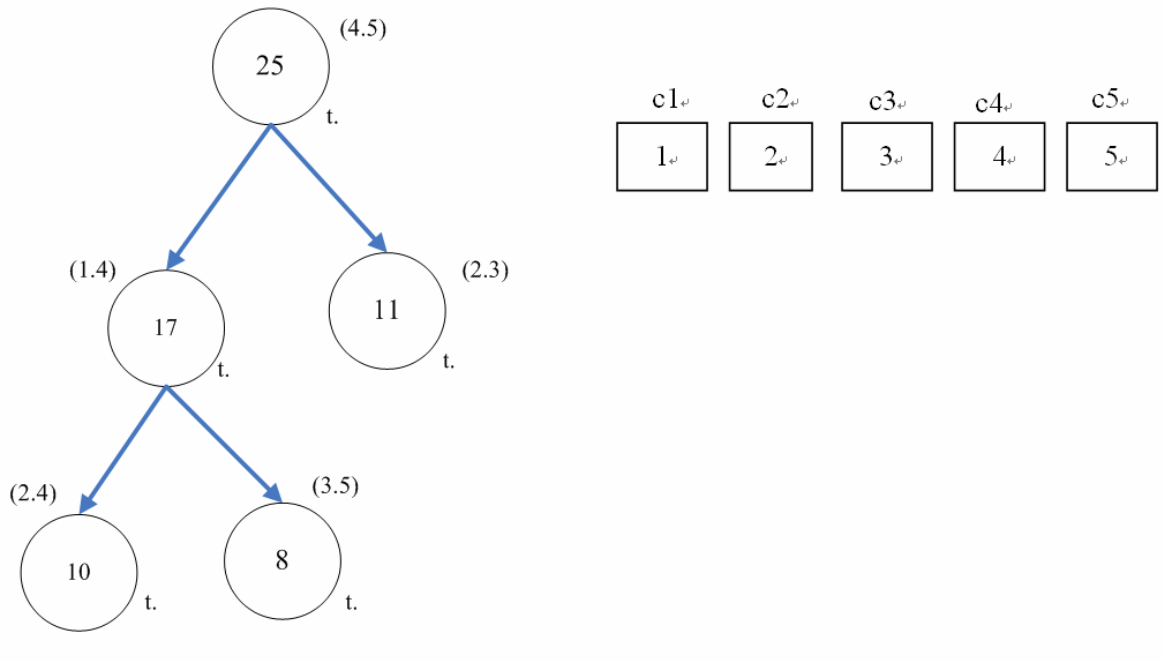


圖 26、範例二應用有向性叢集演算法

步驟一：從 heap 中選出第一個群組(4,5)，合併叢集 4 和叢集 5 成為新叢集 4，刪除舊的叢集 5。

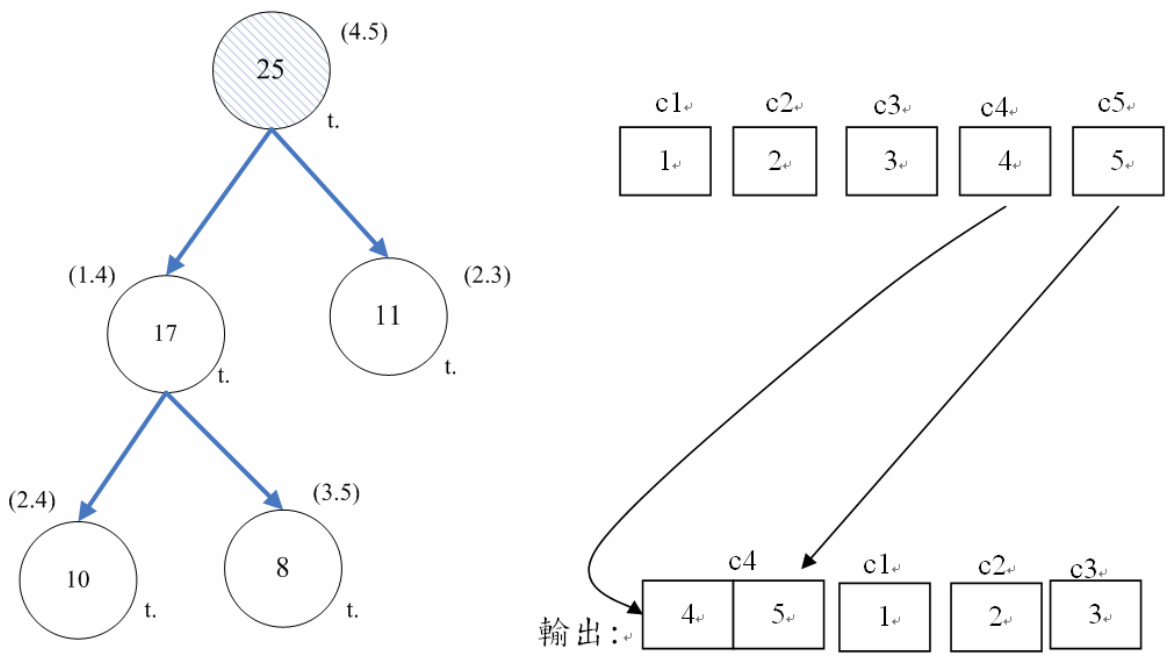


圖 27、範例二應用有向性叢集演算法，合併叢集 4、5

步驟二: 重新更新 heap 後，並重計算叢集之有向邊權重值。在從新 heap 中，選出 heap 裡的第一個群組(2,3)，合併叢集 2 和叢集 3 成為新的叢集 2，刪除舊的叢集 3。

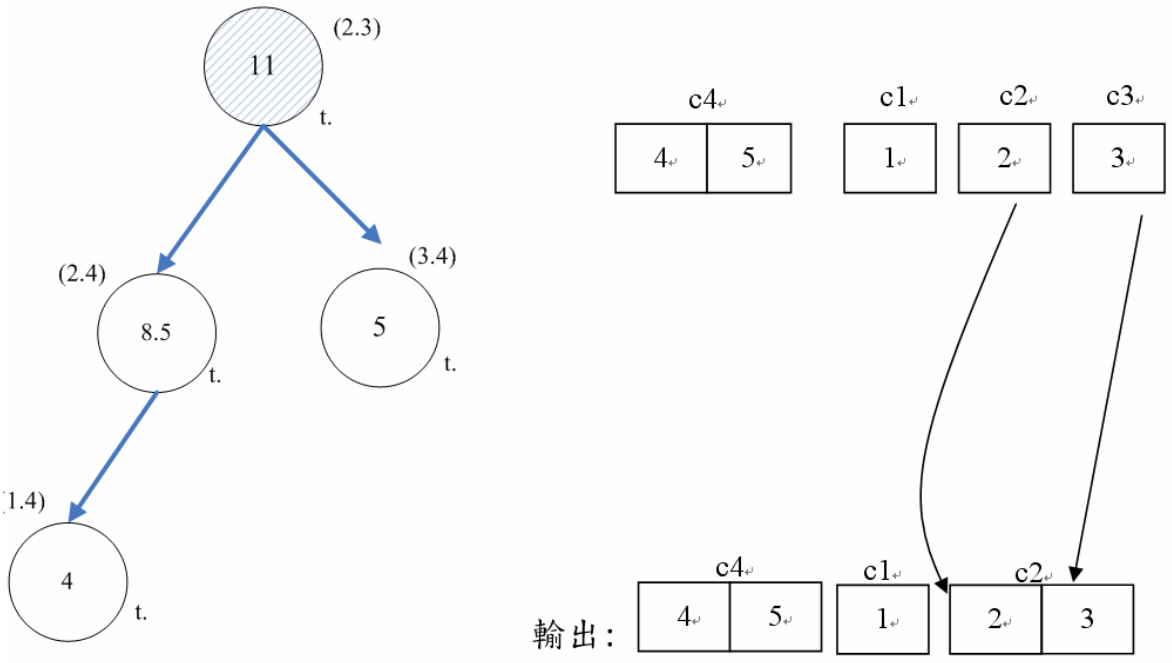


圖 28、範例二應用有向性叢集演算法，合併叢集 2、3

步驟三: 重新更新 heap 後，並重計算叢集之有向邊權重值。在從新 heap 中，選出在 heap 裡第一個群組(2,4)，合併叢集 2 和叢集 4 成為新的叢集 2，刪除舊的叢集 4。

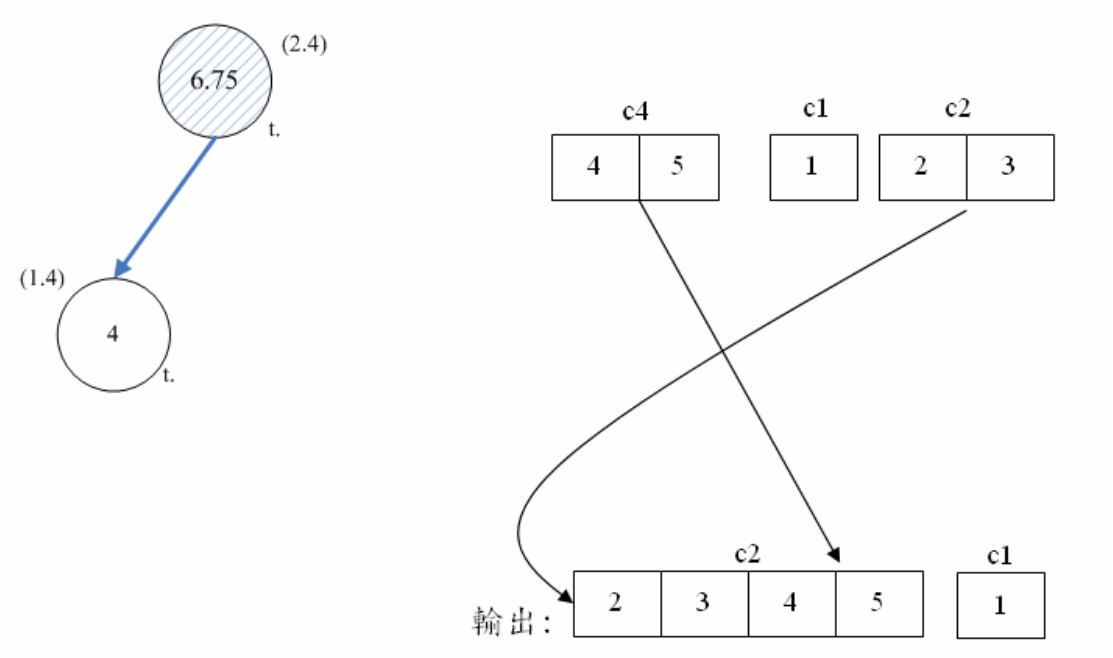


圖 29、範例二應用有向性叢集演算法，合併叢集 4、2

步驟四: 重新更新 heap 後，並重計算叢集之有向邊權重值。從新 heap 中，選出 heap 裡第一個群組(1,2)，合併叢集 1 和叢集 2 成為新的叢集 1，刪除舊的叢集 2。

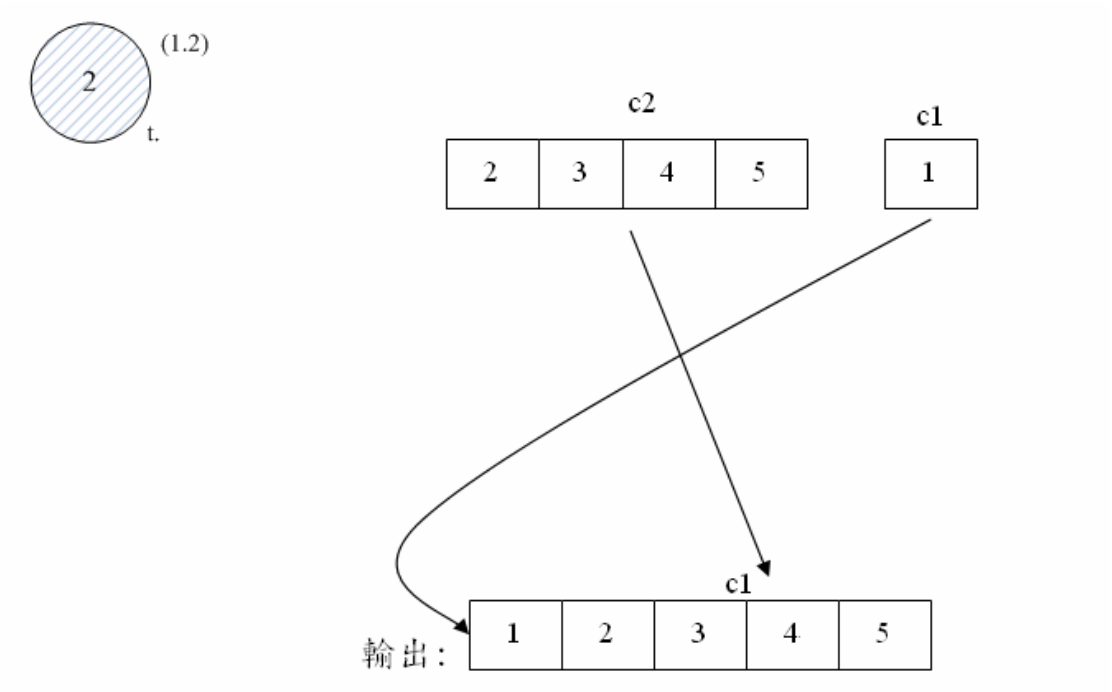


圖 30、範例二應用有向性叢集演算法，合併叢集 2、1

應用有向性叢集演算法之後，所有的叢集會整合成為一個單一叢集，因此產生一條連結節點的序列。經由此廣播序列套用公式(3)，便可以得到此序列之平均讀取間為 1.6056338 (3*8+11+17*2+2*10+25/71= 114 / 71)。

第五章 實驗結果與討論

藉由執行實驗來進行模擬，來分析評估本研究所運用各種貪婪策略演算法，解決於廣播排程式效能之問題，以確保聯網的服務品質。經由序列運算資料項間的平均長度，比較各種貪婪策略之演算法其優劣。

5.1 模擬實驗環境

本研究所基於 Microsoft Windows XP 的作業系統平台上，利用具有物件導向(Object-Oriented)功能的爪哇程式語言(Java Language)來開發模擬程式以及無線廣播環境。

本文採用 Zipf[26]分配模擬用戶端對資料項請求的頻率，在 Zipf 分配裡，其值(θ)介於 0 和 ∞之間，當 θ 值等於 0 時資料項的頻率為平均分布。隨著 θ 值的增加，資料項的請求頻率就會愈來愈傾斜。

假設用戶端所提出查詢請求都是固定大小、長度一致。當廣播伺服器接收到用戶端請求資料項後經過一段時間，廣播伺服端將查詢請求進行廣播排程資料配置的演算，產生出較佳廣播序列，降低用戶端讀取的平均等待查詢時間。

5.2 實驗資料檔與參數介紹

憑藉造檔程式產生出實驗時必要的資料項檔案 10 個(檔名: multiple00 至 multiple09)，每個資料檔案裡存在有 50 個資料項。兩資料項間的相依程式為權重值。利用這些檔案中的資料項值加上公式(4)的運算，可以得知每個資料項間的權重值。

$$W^{(i)} = \frac{i^{-s}}{\sum_{j=i}^{m} j^{-s}} \quad , \quad 1 \le i \le m \qquad \text{公式(4)}$$

S 為偏斜因素(skew factor); m 為用戶端之查詢數。其用戶端請求查詢數分為 100、150、200、250、300，每個查詢數的存取頻率各使用不同的偏斜度參數，分為均勻 0、1.0、2.0。如表 1 所示，為模擬環境參數。透過實驗，評估各個不同的貪婪拓撲排序機制之差異，經由各個策略下所產生的有向性線性排序，運算其頂點之間的平均路徑長度為基準，分別以不同的查詢數量和偏斜因素來進行模擬。其模擬參數如下:

表 1、模擬環境參數

Number of data	100
Number of data within a query	2
Number of different query	100、150、200、250、300
skew factor	0、1.0、2.0

參照表 2、3、4，rank 演算法是由學者 R. Sakellariou 與 H. Zhao[23] 提出的混合式演算法中提出階層值運算各節點的階層值，其值由小至大排序。topological 演算法則運用拓撲排序概念。group 演算法是由學者 R. Sakellariou 與 H. Zhao[23]提出的混合式演算法進行分群，在每個群集裡運用 greedy 演算法進行排序。greedy_rank 演算法則應用階層值於貪婪演算法以階層值小者為選擇條件進行排序。bidirectional 演算法為連續性雙向演算法，應用階層值加入貪婪演算法以 preference list 順序為選擇條件進行排序。cluster 演算法為有向叢集演算法。

5.3 實驗結果分析

以偏斜度為 0 在不同的查詢數量分別為 100、150、200、250、300

下，做各個演算法模擬實驗，並比較其優劣。圖 31 中顯示出隨著用戶端請求的查詢數的增加，平均查詢讀取時間也相對的提高。本研究所提出的連續性雙向演算法，可看出在查詢數為 100、150 時皆比其它演算法之平均查詢時間明顯來得短，在其餘的平均查詢時間皆微幅降低。

表 2、偏斜度 S=0 以不同查詢數量下，各演算法的平均查詢讀取時間

演算法 / 查詢數量	rank	topological	group	greedy_rank	bidirectional	cluster
100	11.00	10.92	10.16	10.29	9.54	9.80
150	12.56	12.66	12.28	12.25	11.82	12.21
200	13.24	13.24	13.08	12.94	12.81	13.13
250	14.08	14.13	14.01	14.00	13.92	14.02
300	14.60	14.59	14.55	14.50	14.46	14.55

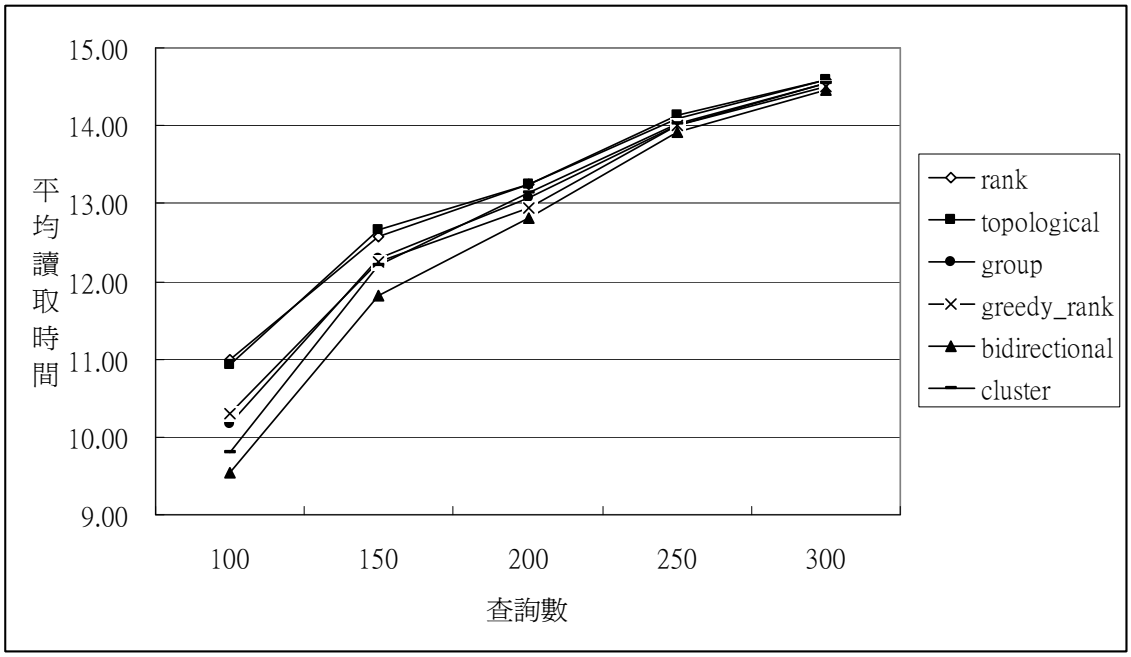


圖 31、依不同 query 在偏斜度為 0 下用戶端的平均讀取時間

圖 32 是依不同查詢數量 100、150、200、250、300 在偏斜度為 1 下之用戶端的平均查詢讀取時間。從顯示可看出本研究所提出的連續性雙向演算法，都比另外貪婪拓撲排序機制演算法的平均查詢讀取時間獲得較短的時間。

表 3、偏斜度 S=1.0 以不同查詢數量下各演算法的平均查詢讀取時間

演算法 / 查詢數量	**rank**	**topological**	**group**	**greedy_rank**	**bidirectional**	**cluster**
100	11.05	9.53	9.59	8.21	7.56	7.90
150	11.19	10.77	10.34	10.09	9.44	9.50
200	11.32	10.83	10.86	10.47	10.20	10.41
250	11.27	11.03	11.11	10.88	10.57	10.72
300	11.29	11.25	11.15	11.09	10.89	10.97

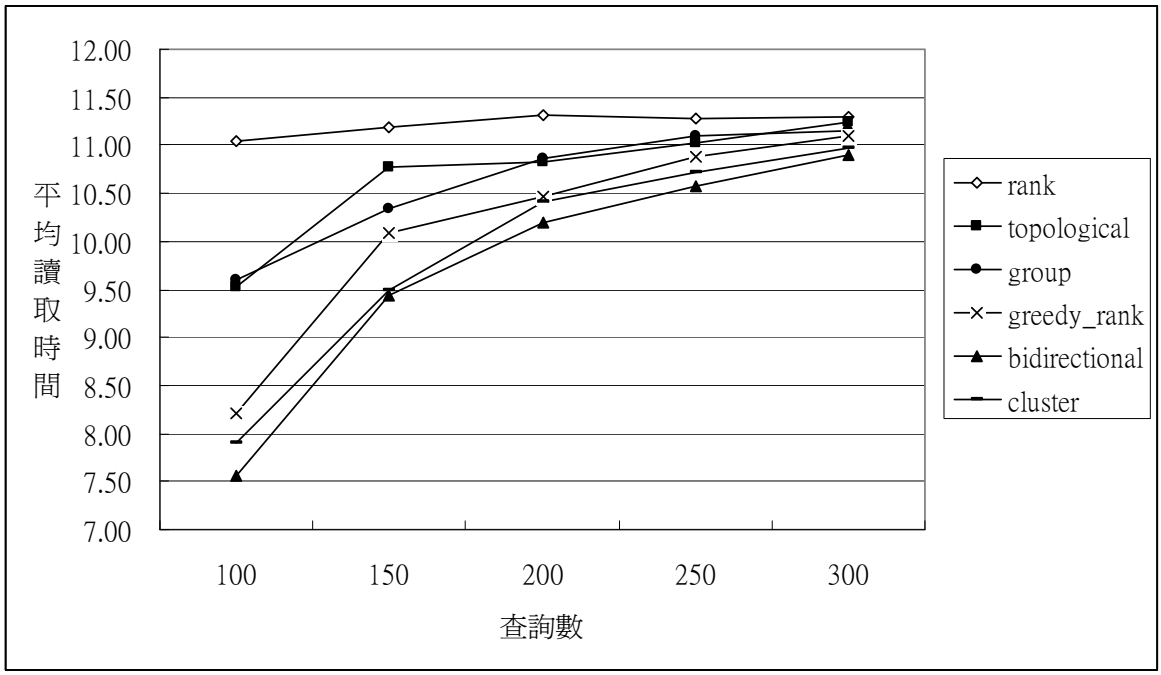


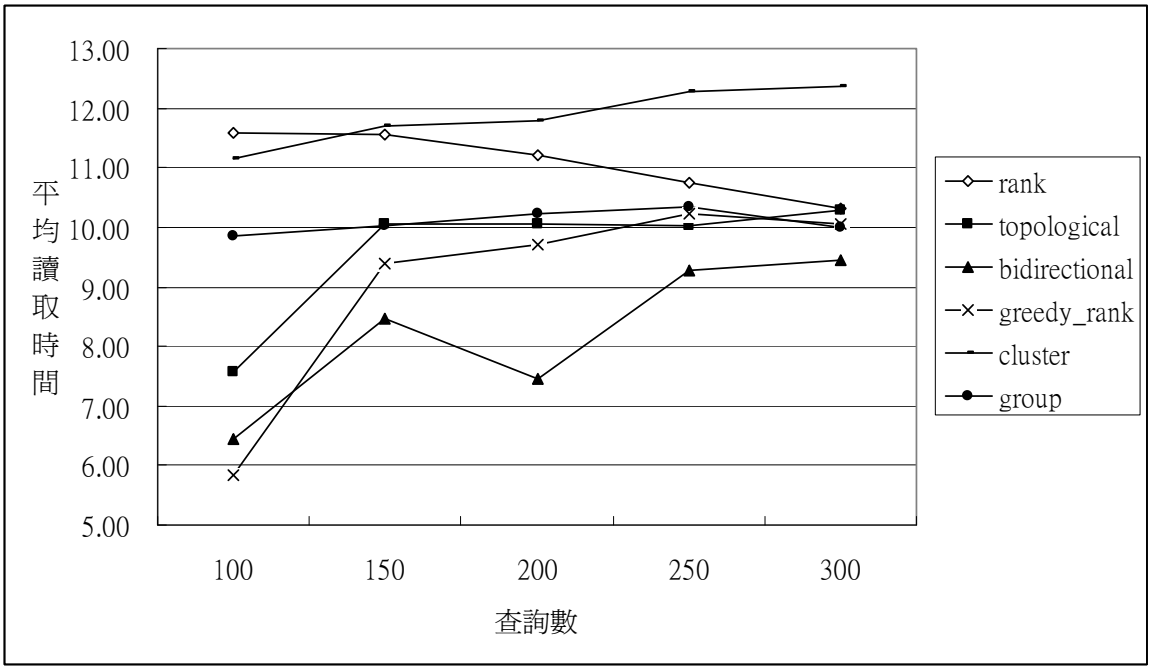
圖 32 依不同 query 在偏斜度為 1 下用戶端的平均讀取時間

依實驗圖 33 中，同上述以不同的查詢數量 100、150、200、250、300 運用在偏斜度為 2 之下的用戶端平均查詢讀取時間。圖中說明了連續性雙向演算法在查詢數 150、200、250、300，比起其他的貪婪拓撲排序機制，皆可獲得較好的平均查詢讀取時間。

表 4、偏斜度 S=2.0 以不同查詢數量下各演算法的平均查詢讀取時間

演算法 / 查詢數量	**rank**	**topological**	**group**	**greedy_rank**	**bidirectional**	**cluster**
100	11.57	7.57	9.85	5.85	6.44	11.14
150	11.57	10.04	10.03	9.38	8.47	11.71
200	11.22	10.05	10.24	9.72	7.45	11.78
250	10.75	10.02	10.36	10.24	9.26	12.29
300	10.31	10.28	9.99	10.04	9.44	12.37

圖 33 依不同 query 在偏斜度為 2 下用戶端的平均讀取時間



第六章 結論

以往學者專家們對於無線網路廣播資料排程的研究中，所探討的廣播資料項主要都是針對於資料項之間的獨立性。在現實的無線廣播環境下，廣播資料項之間是存在相依性的，因而本研究以其為主要之考量。使用有向無循環圖形資料存取模式及其表示順序為限制，並運用各種不同的貪婪拓撲排序的策略，考量其每個節點的階層值，經由各個策略下所產生的有向性線性排序，運算其頂點之間的平均路徑長度，比較其各個演算法效能之差異。主要目的是降低網頁讀取的平均等待查詢時間，並找出最佳的廣播資料排序。

經由模擬實驗顯示，本研究所提出之連續性雙向演算法，成功運用了階層值於有向無循環圖形，因而比起其他以貪婪拓撲排序的策略之演算法，確實能獲得較短的平均讀取等待查詢時間。

參考文獻

中文部份:

[1] 廖榮貴、許正憲、王龍發、蔡能聰等編著, “資料結構與演算法:使用 Java”, 文魁資訊, 民 94 初版, 2005.

[2] 黃國瑜、葉乃菁編“Java 資料結構”, 文魁資訊, 民 90 初版, 2001.

西文部份:

[3] D. Adolphson and T. C. Hu. “Optimal linear ordering,”SIAM Journal on Applied Mathematics, 1973.

[4] D. Aksoy and M. S. F. Leung, “Pull vs Push: A Quantitative Comparison for Data Broadcast, “Global Telecommunications Conference, 2004.

[5] A. Bar-Noy, J. Naor. And B. Schieber. “Pushing dependent data in clients-providers-servers systems.” Wireless Networks, 2003.

[6] B. Cirou and E. Jeannot. Triplet:” A clustering scheduling algorithm for heterogeneous systems.” In international Conference on Parallel Processing Workshop, 2001.

[7] Y.D. Chung and M.H. Kim. “On Scheduling Wireless Broadcast Data“. Technical Report CS-TR-98-134, KAIST, Department of Computer Science, 1998.

[8] Y. D. Chung and S. Bang. M. Kim, “An efficient broadcast data clustering

method for multipoint queries in wireless information systems," The journal of Systems and Software, 2002.

[9] C. Ding, X. He, H. Zha, M. Gu, H. Simon. "A Min-max cut algorithm for graph partitioning and data clustering."IEEE 1st Conference on Data Mining, 2001.

[10]O.E. Demir and D.Aksoy, "Energy-Efficient Broadcast-based Event Update Dissemination," Pertormance, Computing, and Communications, 2004 IEEE International Conference, 2004.

[11]Q. Fang , S. V. Vrbsky, Y. Dang, and W. Ni, "A Pull-Based Broadcast Algorithm that Considers Timing Constraints," Proceedings of the International Conference on Parallel Processing Workshops,2004.

[12]J. L. Huang, M. S. Chen, and W. C. Peng. "Broadcasting dependent data for ordered queries without replication in a multi-channel mobile environment." In Proceedings of the 19th International Conference on Data Engineering, 2003.

[13]J.J. Huang and Y. Lee," Efficient Index Caching Schemes for Data Broadcasting in Mobile Computing Environments," Proceedings of the 14th International Workshop on Database and Expert Systems Applications, 2003.

[14]Y. Huang and Y. H. Lee," An Efficient Indexing Method for Wireless Data Broadcast with Dynamic Updates," Communications, Circuits and Systems and West Sino Expositions, IEEE 2002 International Conference, 2002.

[15] A. K. Jain, M. N. Murty, P. J. Flymn.” Data Cluster: a review,” ACM computing Surveys.

[16] K. F. Jea and M. H. Chen,” A Data Broadcast Scheme Based on Prediction for The Wireless Environment,” Proceedings of the Ninth International Conference on Parallel and distributed Systems, 2002.

[17] A. B. Kahn, “Topological Sorting of Large Networks”, Communications of the ACM, 1962.

[18] T. Kaji and A. Ohuchi,” A simulated annealing algorithm with the random compound move for the sequential partitioning problem of directed acyclic graphs,” European Journal of Operational Research, 1999.

[19] J. Ma, K. Iwama, T. Takaoka and Q. P. Gu,”Efficient Parallel and Distributed Topological Sort Algorithms, ”Parallel Algorithms/Architecture Synthesis ,Proceedings, Second Aizu International Symposium, 1997.

[20] W. Ni, Q. Fang and S.V. Vrbsky, ”A Lazy Data Request Approach for On-demand Data Broadcasting,” Proceedings of the 23rd International Conference on Distributed Systems Workshop,2003.

[21] G. C. Sih and E. A. Lee. A compile-time scheduling heuristic for interconnection-constrained heterogeneous processor architecture. IEEE Transactions on Parallel and Distributed Systems, 1993.

[22] N. Saxena, K. Basu, and S. 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, 2004.

[23] R. Sakellariou and H. Zhao. "A Hybrid Heuristic for DAG Scheduling on Hetero- generous Systems". Proceedings of 13th Heterogeneous Computing Workshop, HCW2004, Santa Fe, NM, 2004.

[24] D. Tsaih, G .M. Wu, C. B. Wang, Y. T. Ho," An Efficient Broadcast Scheme for Wireless Data Schedule Under a New Data Affinity Model," Lecture Notes in computer Science, 2005.

[25] X. Wu. and V. C. S. Lee," Preemptive Maximum Stretch Optimization Scheduling for Wireless On-Demand Data Broadcast," Proceeding of the International Database Engineering and Applications Symposium, 2004.

[26] G. K. Zipf. Human Behavior and the Principle of Least Effort. Addison-Wesley, Massachusetts, 1994.

[27] J. Zhang and L. Gruenwald," Optimizing Data Placement Over Wireless Broadcast Channel for Multi-Dimensional Range Query Processing," Proceedings of the International Conference on Mobile Data Management, 2004.