第四章 網路層

除了介紹電路交換、訊息交換、分封交換與電報傳輸技術外，本章也詳解鏈路狀態和距離向量路之徑選擇技術，讓讀者奠定良好的網際網路觀念。

4-1網路層簡介

『網路層』(Network Layer) 負責處理將資料由一部工作站傳給另一部工作站之間的路徑選擇 (routing) 問題，其中包含建立、維護、以及結束兩部工作站之間的連線。連線建立時可能會有許多路徑可供選擇，但連線建立後，兩部電腦之間便依據該連線所建立的路徑來傳送訊息。

第二層鏈路層 (Data-Link Layer) 提供兩部工作站之間傳輸媒介的存取，以及資料流動管理，也表示兩部工作站還在同一區域內，並未跨越出網路之外和其他工作站連接。第三層網路層的功能是如何在複雜的網路之中(由多個網路所構成)，尋找及連線到所欲通訊的工作站，因此可稱之為**『工作站對工作站連線』(Station-to-Station Connection)**。如圖 4-1 中，工作站 A 和工作站 B 也許相距非常遠，假設工作站 A 在高雄，而工作站 B 在美國舊金山。兩部工作站必須在這複雜的網路上尋找到對方，假如工作站 A 所建立之連線為：A → node1 → node5 →node3 → node8 → node9 → 工作站 B。建立連線的方法是每個端點 (node) 都必須負責尋找下一個端點位置，也就是說，網路層的工作是由網路上所有端點共同來完成。端點也許是一個交換機、路由器、或網路閘門等網路設備。

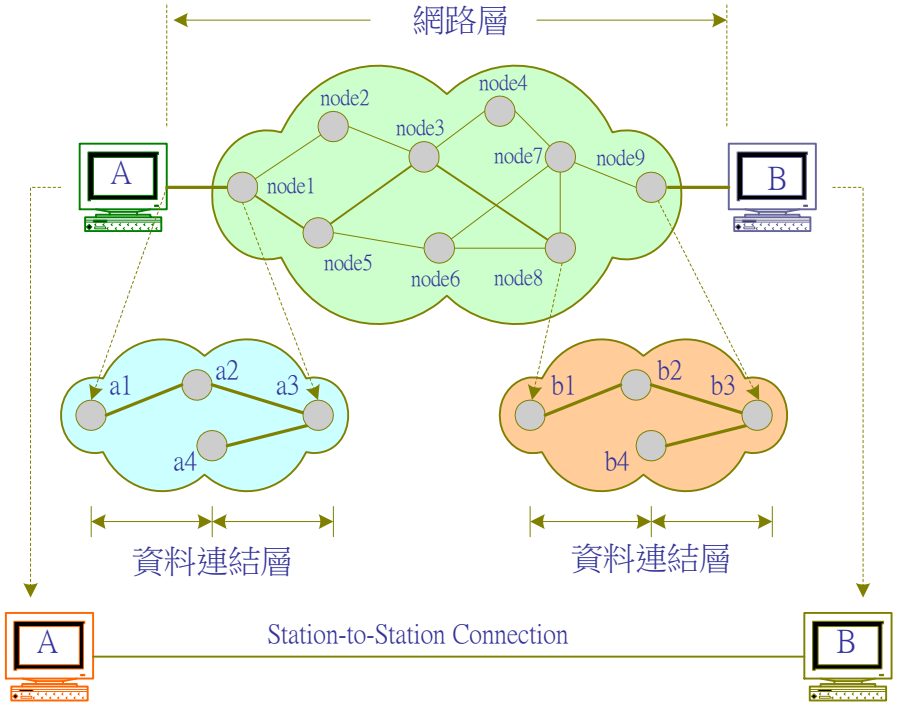


圖 4-1 網路層功能

可從另一觀點來看，在兩個網路節點（如兩個路由器）之間，也許是由許多連結端點所構成，這些接續點可能是：集線器（Hub）、橋接器（Bridge）、或交換器（Switch）等，它們負責的工作是在傳輸媒介上存取與傳送訊框，這就是鏈路層最重要的功能。如圖 4-1 之中，工作站 A → a1 → a2 → a3 → node1 之間的連線，它們也許是 Ethernet 網路、或 ATM 網路等。因此，我們可以做一個簡單的結論來描述網路層和鏈路層之間的關係：工作站 A 尋找到工作站 B 所經過的網路節點之間的路徑選擇，是由網路上所有節點的網路層所共同完成的；而每一網路節點之間，也許會經過許多傳輸媒介所構成網路之間的訊息傳輸，這就是鏈路層所提供的服務。在一條網路層連線中，也許是經由許多端點的鏈路層來共同完成，這些端點可能屬於不同的網路架構，如 Ethernet 網路、或 Token-Ring 網路等。但當我們在考慮網路層的連接技術時，必須假設有關鏈路層之間的不同連結技術都已建構完成，而不用再去理會它，這才合乎通訊協定的堆疊原理。

由以上的敘述，在網路層我們歸類三個主要議題來探討：

(A) 連線服務方式

一般網路層有下列兩種服務型態：

- **連接導向方式（Connection-oriented）：**兩部工作站傳送資料之前，先建立好連線，再依照連線路徑傳輸資料，又稱可靠性傳輸（Reliable Transmission）。

- **非連接方式(Connectionless)：**兩部工作站傳輸資料之前並未建立連線，當傳送出每一筆資料時，該筆資料再依照當時網路情況，尋找適合路徑傳送，又稱非可靠性傳輸（Unreliable Transmission）。

網路層所提供連線服務型態，是一件非常關鍵性的問題，它牽連著整個網路架構的型態。在無遠弗屆的網路之中，如要提供連接導向服務，所花費的成本勢必非常的大，但它又能提供比較穩定的傳輸服務。雖然可以利用可靠的上一層服務（如，TCP）來偵測及保障非連接服務的網路連結（如，IP），但終究是亡羊補牢的做法。因此，網路型態如採用非連接服務，一般只能應用於檔案傳輸或電子郵遞方面。也就這樣，一般廣域網路使用非連接服務，但相對的，區域網路希望提供層次較高的應用（如，負荷分擔、工廠自動化等），則採用連接導向服務。

(B) 連接技術

連接技術是整個網路層的核心工作，主要功能是**『如何在複雜的網路中尋找目的工作站之位址，並連接到它』**，基本上，有下列四種連結技術：

- **電路交換技術（Circuit Switching Technique）：**透過實體連線技術，在複雜的網路上連接連線兩端電腦，如電話網路系統。

- **信息交換技術（Message Switching Technique）：**以傳送訊息為單位，訊息由傳送端開始每經過一個路徑，再尋找下一個空閒路徑傳送，一邊尋找路徑，一邊傳送資料到達目的地。

- **分封交換技術(Packet Switching Technique):**將欲傳送的訊息分割為若干個小封包。傳送資料之前事先尋找可以到達對方的路徑，並預定路徑中所有經過的鏈路，傳送資料時，再按照所建立成功的路徑，依序傳送訊息的小封包。

- **電報傳輸技術 (Datagram Technique):**同樣的，將欲傳送的訊息分為若干個小封包，但傳送之前並未建立路徑，就將小封包依序的發送到網路上，每一個封包自行尋找可到達目的地之路徑。

(C) 工作站定址方式

『工作站定址』(Addressing)(或稱命名 Naming) 的功能是**『使每一部工作站在一個複雜網路上有一個獨一無二的名稱，才能使其他工作站找到它』**。連接技術不同，所延伸的定址方式也不相同，而且也會影響網路的成長性。基本上，定址方式是件非常複雜的問題，以電話系統為例，全世界上有上億支電話，每一支電話都必須有唯一的號碼，如果沒有一個特殊編號方式，那每一支電話的號碼也許會非常的長，而且對號碼的成長性也會受到限制。因此對於較大較複雜的網路系統，大部分都是採用階級式編碼，如電話系統用國際碼、區域號碼、以及本地號碼來組織而成，例如電話號碼是：(886-7-5744438)。一般採用定址 (或命名) 方式如下：

- **區域性階層定址：**按照區域性關係區分為若干個階層，每一階層都編排一個特殊識別號碼。工作站再依照所在的位址給于一個名稱，隨著工作站移動位址，而必須重新命名，如電話號碼或 IP 位址的命名方式。一般較大型網路都採用此方式。
- **功能性階層定址：**依照網路可能提供的服務性質分成若干層次，每一階層同樣給於特殊的名稱。工作站再依照其所從事的工作性質給於一個名稱。工作站不會因移位而必須重新命名，但會因工作內容改變而必須重新命名。一般區域網路都採用這種方式，如 Microsoft 網路的**『網域 + 工作站名稱』**、或 DNS 命名方式。

4-2電路交換技術

在**『電路交換』(Circuit Switching)** 技術中，通訊兩端必須在通訊之前，完成實際電路連線。連線功能如下：連線端發送出要求連線訊號之後，該訊號在網路上尋找可能到達目的地的下一個空閒路徑，找到後就將它佔為己有，再往下一個端點繼續尋找下一個路徑，一個端點接一個端點的連線，一直到達目的地為止。如果連線成功便依照連接路徑傳回連線成功訊號；否則在某一個端點上找不到空閒路徑時，便即時回應連線失敗。當兩端連線建立後，就按照連線路徑傳送資料，除非雙方任何一端要求中斷連線，否則不論是否在傳送資料，將一直佔有著整條連線，由此可知，電路交換是以連接導向的可靠性傳輸。

如圖 4-2 所示，各個端點（Node）一般都採用數位交換機，每一個端點的邏輯鏈路被佔有時便不可以再分配給其他連線要求，一直到該連線結束才可以分配給其他連線。如工作站 B 和工作站 E 連線後，就在交換機 SW1、SW2 及 SW4 固定佔有一條虛擬鏈路。

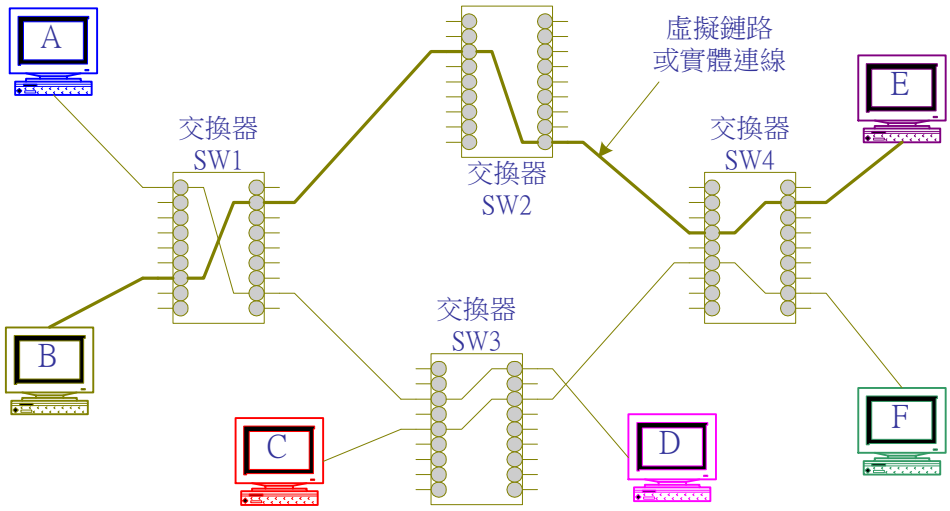


圖 4-2 電路交換技術

4-2-1 電路交換的優點

電路交換有下列優點：

(1) **可靠性高**：通訊之前便建立連線再通訊，兩端傳送資料不易遺失。

(2) **即時性高**：連線後該線路已被佔有且獨享該線路，不會和他人分享，可以作即時資傳送料，適合語音和視訊傳輸。

(3) **連續性高**：佔有固定線路，且依該線路傳送，因此傳送到對方之封包次序不會錯亂。

(4) **錯誤率低**：依固定線路傳送，錯誤率較低。

4-2-2 電路交換的缺點

電路交換有下列缺點：

(1) **線路佔有時間過長**：建立連線後不一定連續傳送資料，線路佔有時間太長，成本費用太高。

(2) 線路使用率低：建立連線後，又佔有該連線。如連線距離太遠，在傳送資料時，大部分連線區段皆空閒，所以線路使用率降低。

(3) 廣播困難：電路交換是點對點間的連線，對於廣播或多點傳送不易實現。

4-3信息交換技術

『訊息交換』(Message Switching)技術是一種非連接方式（Connectionless）。連線運作如下：傳送端欲傳送資料之前並未建立連線，而是先找到任意一個可以到達目的地的端點，便將整個訊息全部傳送給它，這個端點再依照訊息的目的地，尋找下一個端點的路徑，整個訊息就依照所經過的端點自行尋找下一個路徑傳送，而到達目的地。當訊息所經過後，路徑即時被釋放出來，因此接收端回應訊息時也必須如同資料訊息一樣，一個端點一個端點尋找空閒路徑。所以資料訊息的傳送和回應，非常有可能走不同的路徑。

網路層將上一層（傳輸層）通訊軟體所傳來的訊息，經包裝後便直接傳送給下一層通訊軟體，並未分割該訊息，稱之為**『訊息封包』(Message Packet)**交換。當訊息封包到達每一端點後，再尋找出下一個端點的路徑時，必須將整個封包儲存於端點上。當尋找出下一個端點的空閒路徑後，再將訊息封包往前送，這個功能稱之為**『儲存再前送』(Store-and-Forward)**。又在兩個端點之間執行儲存再往前送的功能時，可針對訊息封包做錯誤偵出，一般都採用檢查集（Check-Sum）方法。

任何兩個端點之間檢查出錯誤發生，可要求傳送端點重新傳送。因此，當訊息到達目的地時，應該沒有錯誤（但不保證沒有）。

如圖 4-3 所示，工作站 A 傳送資料到工作站 D，其經過 node1、node2、node5、node9 端點。當訊息到達任一端點（如，node2），該端點便將該訊息存放於本身儲存空間內，再判斷是否傳送給本端點所屬的工作站，或是其它端點之工作站。如果是目的位址在其它端點上，必須負責再尋找出下一個路徑（如，node5），依此類推，將訊息轉送到工作站 D。當訊息每經過一端點後，即時釋放該連線。

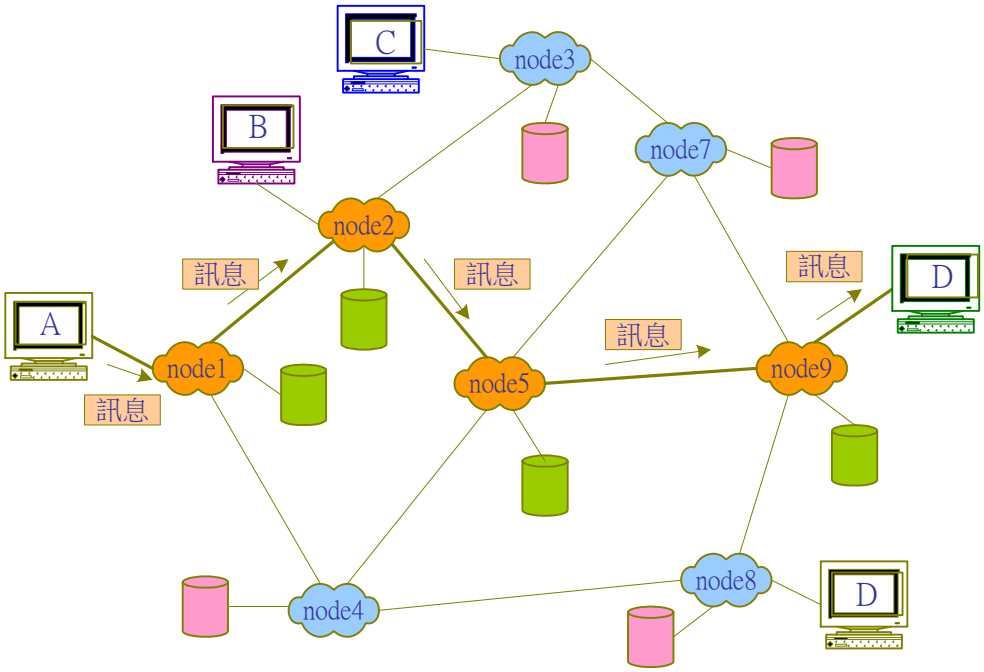


圖 4-3 訊息交換技術

4-3-1 訊息交換的優點

訊息交換技術有下列優點：

(1) **線路使用率高**：訊息傳送中才會佔有線路，線路的使用率高。

(2) **錯誤檢查完整**：經過每一個端點都有做錯誤檢查，當到達目的地時訊息應該沒有錯誤發生。

(3) **適合近距離傳送**：快速訊息傳送。適合比較近距離、而且品質較好的小網路環境使用。

4-3-2 訊息交換的缺點

(1) **即時性較低**：當開始傳送前不能預估在什麼時間內可傳送到達對方，對於連續訊息的傳送，每一個訊息封包的延遲時間的差異非常的大。

(2) **重新傳送機率高**：每次整個訊息（message）傳送，如果訊息太大（如 10Mbps），如在訊息中任何位元(bit)發生錯誤，則整個訊息必須重新重送，對整體來講效率太低。而且當訊息太大的時候，錯誤檢出率也相對較低。

(3) 傳送效率低：傳送前並未確定是否可達到目的地，如果網路距離太遠或對方電腦不存在，則傳送到半途失效的機率高。

(4) 大量儲存空間：對每一個端點來講，必須儲存前一端點傳送過來的整個訊息，如果網路非常忙碌，則停留在端點上的訊息的數目將會增加，因此每一端點必須預留大量的儲存空間。

4-4分封交換技術

『分封交換』(Packet Switching)技術是一種連接導向服務，通訊雙方事先必須建立連線。運作方式如下：首先工作站發出要求連線訊號，該訊號攜帶著目的位址和來源位址，由原始端點開始尋找可以到達目的地的下一個端點路徑，如找到就將連線訊號傳送給下一個端點，並在端點上註冊該訊號經過的鏈路。如此，一個端點傳給下一個端點，並在每一個端點紀錄要求連線訊息的鏈路，一直到達目的地。目的工作站回應是否同意連線訊息時，就依照要求連線所註冊的路徑傳回給要求連線端，雙方傳送訊息就依照所註冊的連線傳送。

要求連線時所經過的端點，都有註冊連線鏈路，但這只是登錄並未實際佔有，以後在傳送資料時，每一個端點依照事先登錄的路徑傳送，因此稱之為**『虛擬電路』(Virtual Circuit)**。訊息在虛擬電路上傳送時，並不保證下一個路徑是空閒的，也就是說，所登錄的路徑也有可能被其他連線佔用。所以訊息到達某一端點並不保證可以即時往下一個端點傳送，訊息也必須儲存在端點上等待下一個路徑是否空閒，因此也具備有**『儲存再往前送』(Store-and-Forward)**的功能。

我們可以發現，虛擬電路具有儲存再往前送功能，因此在每一個端點上必須有大量的緩衝器儲存訊息。為了提高傳輸效率及減低端點的儲存空間，我們將欲傳送的訊息分割成若干個較小的封包，每個封包上都給於順序編號，稱之為**『分封包裝』(Packet Encapsulation)**。傳送訊息時，再依照封包序號發送到網路上，也許每一個封包上網路上會有不同延遲的時間，但因為每一個封包都是依照事先建立的路徑傳送資料，所以到達目的地時也會依照原來的封包順序排列（In-order）。也可以發現，每一個封包經過端點到端點之間都是儲存再前送，在端點之間就可以完成錯誤控制，因此當封包到達目得地時應該沒有錯誤。又在每一端點之間針對小封包做錯誤檢出，錯誤檢出率也較高；並且發生錯誤時，只要針對故障的封包要求重新傳送，也可以提高整體傳輸效率。

總而言之，訊息傳送前將其分封為若干個封包，再依照事先建立的虛擬路徑傳送，因此稱之為『**分封交換**』(**Packet Switching**)技術。在每一個端點上都必須具有分封交換功能，因此網路上的端點稱為『**分封交換機**』(**Packet Switch**)。

如圖 4-4 所示，工作站 A 希望將訊息傳送給工作站 D。其動作步驟如下：(1) 首先工作站 A 將訊息分割若干個封包並依序編號，每個封包固定大小(或某一長度範圍內) (2) 工作站 A 發出要求連線訊號進入 node1，node1 依照其目的位址尋找出下一個端點(node2)，並將 node1 到 node2 之間的虛擬路徑登錄下來。再由 node2 尋找下一個端點路徑。依此類推，所建立並保留的路徑為 node1 → node2 → node5 → node6 → 工作站 B。(3) 工作站 B 收到要求連線訊號，判斷是否同意連線，並將回應訊號依照保留路徑，傳送給工作站 B (node6 → node5 → node2 → node1 →工作站 A)。(5) 工作站 A 收到回應訊號再依照路徑將訊息封包依序(編號 1、2、3、4、)發送到網路上。(6) 工作站 B 將按順序到達的封包組合回原來訊息。(7) 工作站 A 在訊息發送完後，送出要求斷線的訊號進入 node1，node1 也發送要求斷訊訊號給 node2 並釋放該連線，依此類推，工作站 D 接收到要求斷線訊號並釋放所有虛擬路徑。工作站 D 再將依序收到的封包組合回原來的訊息。一般網路上對於分割封包和組合封包的工作都是由上一層(傳輸層)的通訊軟體來完成，網路層只負責分封交換的工作。

圖 4-4 分封交換技術

4-4-1 分封交換的優點

(1) **可靠性高：**結合電路交換和訊息交換技術的優點，提供一個可靠性的傳輸。而且分封成小封包，每一封包的延遲時間也相對較短。

(2) **即時性較高：**建立虛擬路徑後再傳送，對於封包在網路上的延遲時間比較容易預估，如使用在即時性較高的應用上，也較能夠事先用其他方法來克服時間延遲的問題。

(3) **線路使用率高：**虛擬路徑建立後並未佔有，因此網路的使用率高。

(4) **適用範圍：**可靠性要求較高的中小型網路。

4-4-2 分封交換的缺點

- **大網路費用高：**分封交換網路上每一端點都必須設有分封交換機，在較大網路上設備費用過於龐大（如網際網路）。
- **小訊息效率低：**在大網路上，只傳送小訊息就必須事先建立連線，效率低、浪費資源（如 E-mail）。

4-4-3 各種交換技術的比較

圖 4-5 為上述三種交換技術的比較時序圖。在圖中斜線表示經過每一個端點的延遲時間。電路交換的延遲主要是發生建立連線時，但連線成功後便佔有該線路，爾後再傳遞訊息的延遲時間便較短，因此較適用於即時性的傳輸訊息，如語音或視訊傳送。分封交換技術中雖然建立了連線但並未佔有，因此不論回應訊息或傳遞封包，在每一端點之間都有不一定的延遲時間；雖然即時性方面沒有電路交換技術好，但是透過虛擬路徑整體延遲時間還是可以預估到，而且到達的封包也按照順序，還是可以應用在傳送經過壓縮的語音或視訊。如以傳輸速率來講，訊息交換傳輸速率最快，但傳送前並未建立連線，而且是整個訊息一起傳送也一起延遲，因此並不適合即時性的訊息傳送，主要應用於資料傳輸。

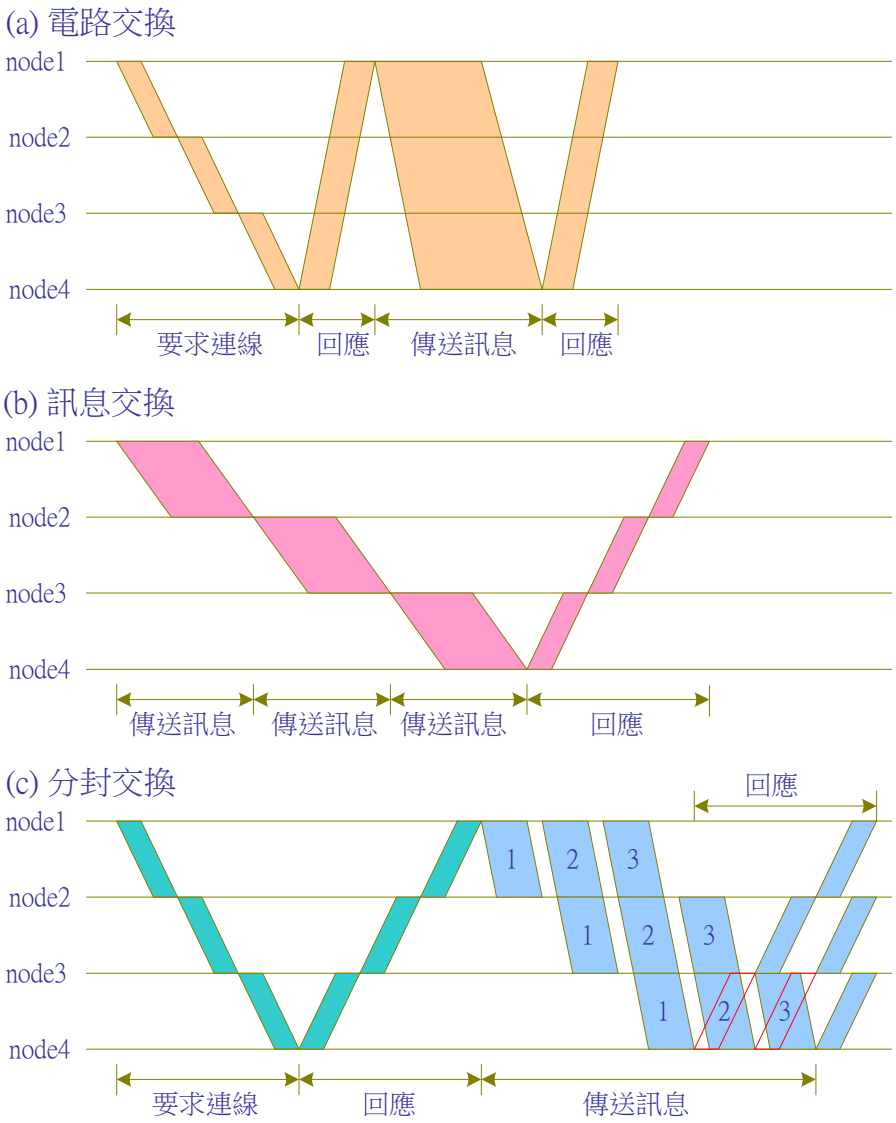


圖 4-5 各種交換技術之比較

在連結交換技術下，各端點的交換機並不同於一般交換器（如，Ethernet Switch 或 IP Switch）。一般交換器只針對進入的封包（或訊框）依照其目的位址轉送到另一個埠口上（Port），並不具有路徑選擇的功能，只限制於交換器本身的運作。然而交換機不但具有路徑選擇功能，而且它的連線是屬於交換機之間共同來達成，目前 ISDN 或 ATM 交換機都採用這種技術模式運作。

4-5電報傳輸技術

『電報傳輸』（Datagrams）是目前網際網路（Internet）上所使用的連結技術，其為非連接方式。運作方式如下：傳送端傳送信息之前，並未事先建立連線，首先將訊息分裝成若干個小封包，稱之為**『分封』（Packet Encapsulation）**；每一個封包上都要註明來源地址、目的

地址和相關的控制訊息；傳送端將封包依序發送到網路上，每一個封包到達某一個端點，該端點再依照封包上的目的地址尋找下一個端點（Next-hop）的路徑來傳送。因此各個封包都是獨立路徑（Isolated Path），意即，每一封包都分別依據當時網路壅塞情形，獨立尋找傳送路經，同一傳送訊息內的各個封包可能走不同路徑到達目的地。各個封包到達目的地的路徑也不盡完全相同，因此，傳送延遲時間也不會相同，封包到達時的次序也不會和傳送時相同，因而會產生順序錯亂（out-of-order）的問題。一般通訊協定關於重整封包順序的工作是由上一層（第四層）通訊軟體來負責。

在電報傳輸的網路裡，端點的工作是將進來的封包依照封包上的目的地址尋找下一個路徑傳送，而不管爾後如何（也許下一個路徑以後網路就不通了），因此封包在傳送時並不保證可以到達。又每一個封包進入端點希望馬上被傳送出去，儘量減少停留在端點上的時間以提高網路效益，因此，一般端點並不做錯誤檢查，當端點讀取到封包的目的位置便即時傳送出去，關於錯誤檢查的工作也是由上一層的通訊軟體負責。在電報傳輸網路上的端點也許是路由器（Router）或網路閘門（Gateway）等。

如圖 4-6 所示，工作站 A 欲將訊息傳送到網路之前，首先將其分割為若干個封包。在每一個封包上都註明清楚來源及目的位址，再依序傳送到網路上，每一個封包自行尋找路徑到達目的地（工作站 D）。各個封包到達目的的時間將可能不同於發送時的順序，也不保證封包是否可到達目的地。

但對整個網路而言，傳輸必須是可靠的，否則便失去架設網路的目的。既然電報傳輸是不可靠的傳輸模式，所以有關於可靠性的維護工作就必須仰賴他的上一層（第四層）的通訊軟體來完成。正因為如此，一般來說如果第三層採用電報傳輸（如 Internet Protocol，IP），那麼第四層就必須採用可靠性傳輸（如 Transmission Control Protocol，TCP），這樣子對整個網路來講，傳輸還是可靠的。除非是作為一般廣播或控制訊息之用，第四層才會採用不可靠的傳輸（如 User Datagram Protocol，UDP）。

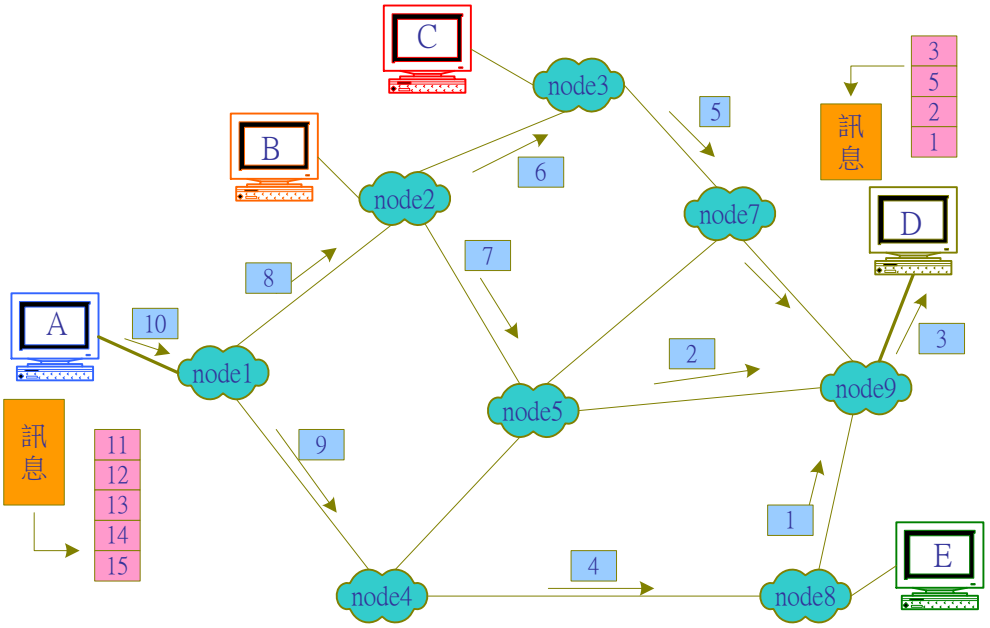


圖 4-6 電報傳輸技術

4-5-1 電報傳輸的優點

(1) **線路使用率高：**封包傳送時才佔有線路，可提高網路的使用率，讓網路空出更多的頻寬讓其他連結共同使用，以降低通訊費用。

(2) **連結效率高：**在大網路的系統內，各地區的網路環境很難掌握，如採用交換連接方式，連結效率較低，並且通訊連線費用也較高。如採用電報傳輸則傳輸效益較高，尤其小訊息（不用分割封包）方面更為顯著。早期網際網路訴求大地區範圍中短訊息的傳送（如 E-mail），電報傳輸技術最為適合，也造就 TCP/IP 網路的廣泛使用。

(3) **網路架設容易：**各個端點（Node）無需昂貴的設備，只要一個便宜的路由器（或數據機、ADSL）連接，便可以成為網路中的成員。也因如此，網際網路擴充非常的快速。早期電話網路經歷百年所能連結的架構，網際網路不到十年就能達成。

4-5-2 電報傳輸的缺點

(1) **可靠性低：**電報傳輸是一種不可靠的連結方式，至於可靠性問題必須由上一層（第四層）通訊軟體來處理。

(2) 封包風暴（Packet Storm）：為了要讓封包能迅速傳送，每一端點接到封包後迅速丟出不會去考慮最佳路徑，因此封包有可能在網路迴轉不停，也有可能同一封包被多個端點重複傳送，造成封包風暴。

4-6靜態路徑選擇技術

網路層最主要的功能就是如何在網路叢林中找到一條路徑可以到達目的地，這就是**『路徑選擇』（Routing）**技術。不論何種連接技術（交換技術或電報傳輸）都必須用到路徑選擇技術，尤其在電報傳輸技術上更是重要。路徑選擇功能主要是由網路上所有**『路由器』（Router）**來共同達成，當然一般主機電腦也具有路由器的功能。路由器（或主機）的工作是當有一個封包由某一個埠進來時，依照路由器內之**『路由表』（Routing Table）**查出應該往哪一個埠送出，轉送到其他的路由器，再由下一個路由器決定路徑傳送。所以路由選擇又稱為**『下一跳躍路徑選擇』（Next-hop Routing）**。

圖 4-7 (a) 為一般網路架構圖，我們將其轉換為路徑拓樸圖，如圖 4-7 (b) 所示。在每一個路由器上都有建立路由表，對於每一個目的地都有標明下一個路由器位置（下一站）。如工作站 A 欲傳送封包給工作站 F，當它的封包進入路由器 1，路由器 1 由它的路由表中查詢到應往下一個路由器 2 傳送。當這個封包進入路由器 2 後，由路由器 2 轉送到路由器 4。再由路由器 4 將封包傳送給工作站 F。

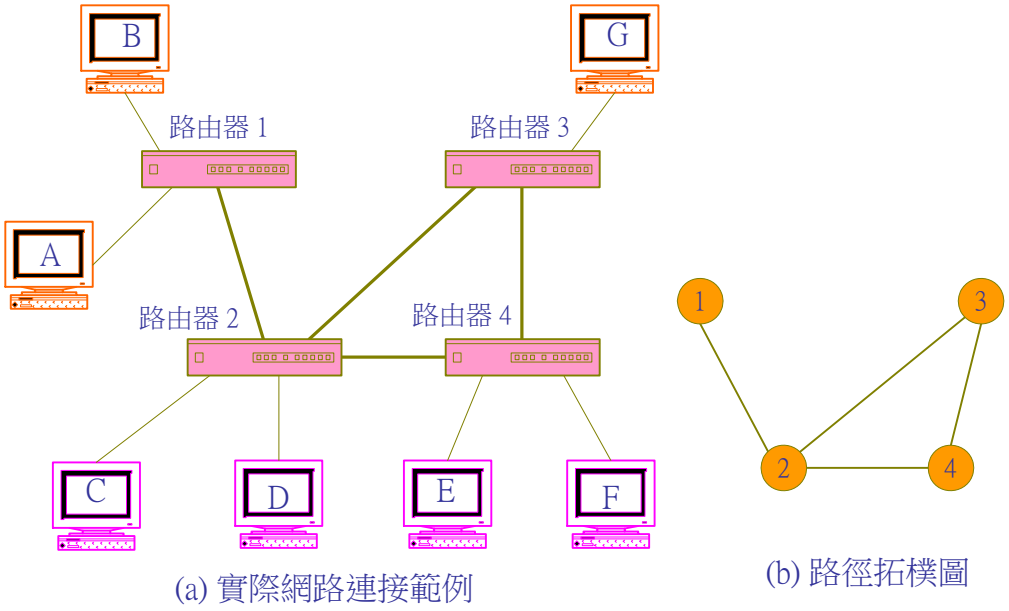


圖 4-7 路徑選擇範例

路由器 1		路由器 2		路由器 3		路由器 4	
目的地	下一站	目的地	下一站	目的地	下一站	目的地	下一站
1		1	1	1	2	1	2
2	2	2		2	2	2	2
3	2	3	3	3		3	3
4	2	4	4	4	4	4	

圖 4-8 路由表範例

對於路由表的建立主要有兩大類：

(1) 靜態路徑選擇（Static Routing）：路由表是靜態的，不會隨時改變，一般有下列兩種：

- 固定路徑選擇（Fixed Routing）
- 熱馬鈴薯方法（Hot-potato）

(2) 動態路徑選擇（Dynamic Routing）：路由表可能經由路由器之間隨時交換訊息，計算出新的路由表，目前較常用的路徑選擇技術有下列兩種：

- 鏈路狀態路徑選擇（Link-State Routing, LS Routing）
- 距離向量路徑選擇（Distance Vector Routing, DV routing）

4-6-1 固定路徑選擇

『固定路徑選擇』(Fixed Routing)是利用人工建立之路由表，建立後除非再用人工修改，否則路由表將永遠不會變更。系統管理者利用固定路由表來規劃網路架構。亦是，網路中主要的路徑分配是利用固定路由表來完成。如果想要更改網路型態，除了變更實體連線外，主要還必須設定固定路由表來決定網路上實際的分配。如果僅更改網路實體架構，而沒有重新設定固定路由表的話，網路將會發生嚴重的錯誤，也可能會因此而癱瘓。

4-6-2 熱馬鈴薯方法

『熱馬鈴薯方法』(Hot-potato) 又稱為洪氾法（Flooding），也是一種靜態演算法。其功能是：當封包由路由器的某一個埠口進入後，該路由器便複製多份，再往其它埠口發送，而

不管封包的目的位址，即往外丟 (好像手拿到熱馬鈴薯，燙到手馬上往外丟)。在理想狀態之下，至少會有一個封包到達目的地。為了避免封包在網路上永無止境的傳遞，在封包內裝設一個跳躍計數器。封包每經過一個路由器，就將計數器的值減一，如果路由器發現某一封包的記數器的值為 0，便將該封包拋棄而不再發送。一般我們都會預估網路最大的範圍 (路由器的數量)，而取一半路徑的數量作為計數器的基準值。如圖 4-9 所示，封包由路由器 A 進入欲傳送到路由器 C。首先該封包被路由器 A 複製兩份，分別發送到路由器 B 和 E，再由它們繼續往前發送，最後至少會有一個複製封包到達路由器 C。

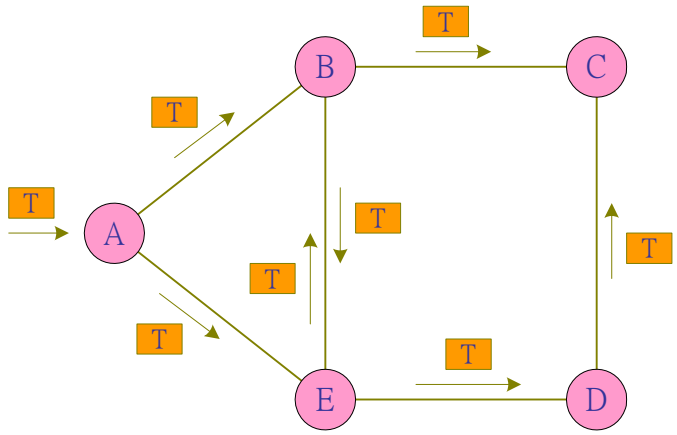


圖 4-9 熱馬鈴薯法

雖然我們用跳躍記數器來限制封包的壽命，但只要發出一個封包便會在網路上產生無數的封包，也隨著路由器的數量而增加，這種現象稱之為『**封包風暴**』(**Packet Storm**)。一種修正方法是：每一封包上編有特殊序號，路由器紀錄所經過的封包序號，如果某一封包已被複製轉送過，當它又從另一埠口進入時，便將其拋棄而不再轉送。這樣的話，就可以減少許多重複轉送的機會。但要維護紀錄序列表也是件頭痛的問題，因為當每一封包進入時，都必須搜尋或登錄紀錄表，而且也很難預估同一封包下一次何時會再重複進入。因此，必須再加入登錄時間，登錄時間經過某段時間後，再將該紀錄刪除。

另一種修正版本可能較適合，稱之為『**選擇性洪氾法**』(**Selective Flooding**)，其功能是：封包進入後，除了搜尋紀錄表外，並非往所有路徑複製轉送，而只發送到比較有可能到達目標位址的路徑上。每一個路由器的埠口可能前往的目標位址，可由人工事先輸入(固定路由)。除非有特殊情況，否則封包風暴問題已大大改善。熱馬鈴薯方法的特點是針對廣播訊息較多的網路上使用，例如，分散式資料庫系統必須隨時廣播更新資料庫訊息。因此，一般使用在一些特殊網路，或者是小型區域網路上。

由於網路大環境會隨時改變，使用靜態路由選擇就顯得非常不方便，因其必須了解網路情況，隨時修改路由表。動態路由選擇會依照網路情況隨時修正路由表，當網路上有任何更動，動態路由選擇器會隨時更新資料，而計算出下一個路徑應該是哪一個路由器的效率最高。雖然動態路由選擇能隨時提供最佳路徑，但他為了維持這個功能，必須隨時在網路上收集最新資訊，也是會浪費不少頻寬。所以一般我們在私有網路上儘量使用靜態路由選擇，以加快網路效率，而且系統管理者也較容易掌握自己網路狀況。對於連結到網路外的公眾網路儘量使用動態路由選擇，以適應因隨時改變而很難掌控的網路。目前所有路由器都具有靜態和動態路由選擇功能，一般系統管理者也可以在主要幹線建立靜態路由選擇，其他就利用動態路徑選擇隨時依照網路情況建立路由表，這樣網路變異性最高，也是最常用的方法。

大型網路中連接許多**『異質性網路』(Heterogeneous Network)**，動態路由選擇必須由各個路由器之間互相交換訊息，各家廠商所生產的路由器也不盡相同，因此在路由器之中必須有共通的**『路徑協定』(Routing Protocol)**來完成。各種路徑協定都有自己的動態路徑選擇方法，以下我們介紹兩種較常用的動態路徑選擇技術，至於路徑協定將會在第十三章以實例說明。

4-7鏈路狀態路徑選擇法

『鏈路狀態路徑選擇』(Link-State Routing，LS Routing)是屬於動態演算法。路由器必須隨時依照最新消息計算出路由路徑，也隨時更新路由表。它是一種**『半集中式』(Quasi-centralized)**的路徑選擇演算法 (Routing algorithm)。首先每一個路由器必須定期測量它和鄰近路由器之間的費用，這費用可能和佇列延遲、頻寬等因素有關，不同通訊協定都有各自的定義。這費用又稱為**『鏈路費用』(Link Cost)**。當每一路由器測出相鄰之間的費用後，定期廣播給其他**『所有』**路由器，該廣播的訊息又稱為**『鏈路狀態』(Link State)**訊息。同樣的，任何一部路由器也會接收到其他路由器廣播的鏈路狀態，再依照這些訊息計算出到達其他路由器的最短路徑，並建構路由表。路由表上註明欲往哪一個路由器的下一個路由器位置 (如圖 4-8 所示)。因此在 LS Routing 演算法下每一個路由器建立路由表的步驟如下：

(1) 利用 Hello 封包查詢相鄰路由器

(2) 計算鏈路費用

(3) 建立鏈路狀態封包並廣播給所有路由器

(4) 計算出最短路徑及更新路由表

以下分別說明各步驟的處理程序：

4-7-1 利用 Hello 封包查詢相鄰路由器

當路由器啟動後，立即發送 Hello 封包（Hello Request）給所有相鄰路由器（或定期發送）。當其它路由器收到 Hello 封包後，必須立即回覆該 Hello 封包（Hello Response），並告知路由器名稱。如圖 4-10 所示，路由器 A 發送 Hello 查詢封包（H_Re）給相鄰路由器（B、C、D），相鄰路由器也回應 Hello 封包（H_Rp）給它。路由器因為這樣而知道有哪些路由器和它相鄰。

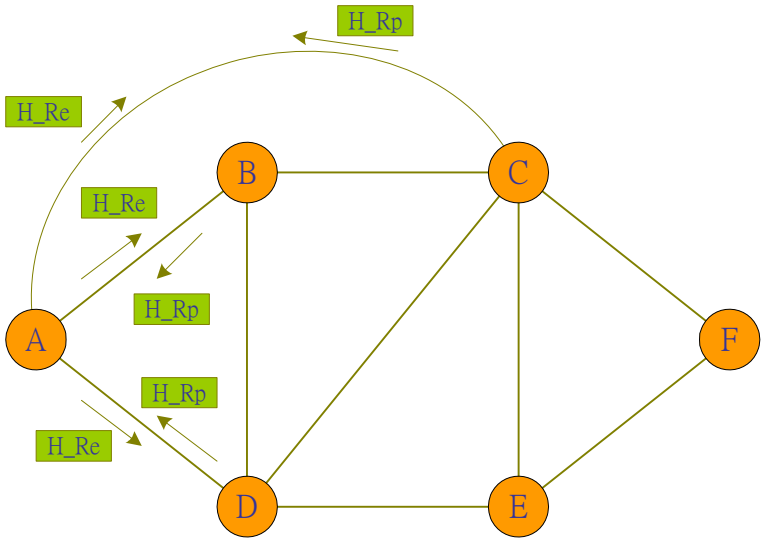


圖 4-10 發送 Hello 封包查詢及回應

4-7-2 計算鏈路費用

發送 Hello 之路由器，由相鄰路由器的回應 Hello 封包，而得知它們之間的路徑費用。再由這些訊息計算出到達相鄰路由器之間的鏈路費用。依照各種通訊協定的需求，針對路徑費用的定義也不盡相同，可能採用傳輸延遲、佇列延遲、或頻寬容量等等因素，但最容易取得的是傳輸延遲。當路由器發送查詢 Hello 封包(H_Re)後，到它收到某一路徑回應(H_Rp)的時間，計算之間差異，再取一半的值（除以 2），這就是該路徑的傳輸延遲時間，一般都以微秒（ms）為單位。在圖 4-11 之中，我們假設所有路由器都經過廣播 Hello 封包後，計算

出它們之間的路徑費用(也是鏈路狀態)，並假設鏈路兩端所計算費用都相同(如，A → B 和 B → A)。

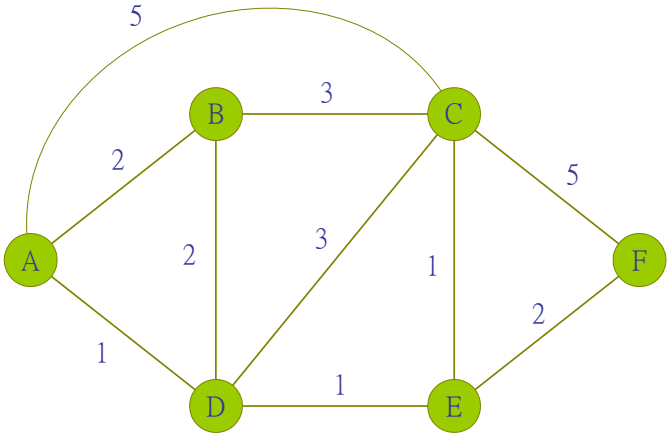


圖 4-11 鏈路狀態範例

4-7-3 建立鏈路狀態封包並廣播給所有路由器

由上一步驟，各路由器已得知相鄰路由器之間的狀態值 (如圖 4-11)，並各自建立鏈路狀態封包，如圖 4-12 所示。

圖 4-12 鏈路狀態封包範例

各路由器再將它的鏈路狀態封包廣播給網路上『所有』路由器。同樣的，每一部路由器也收到所有其他路由器的鏈路狀態封包，再由這些訊息計算出欲往某一路由器的最佳路徑，也建立了路由表。既然任何一部路由器都可由網路收到所有鏈路狀態，並算出自己的路由表，因此我們稱之為『半集中式』的路徑選擇法。

針對廣播鏈路狀態封包可能造成廣播風暴 (Broadcast Storm) 的問題，其類似熱馬鈴薯方法發送。每一個路由器收到封包後，便往其他路徑複製轉送，才可以將封包廣播到所有路由器上。因此，我們必須在封包上編有封包序號和時間戳記 (Time-stamp)。當封包進入路由器後檢查該封包是否來過，如果來過便將其拋棄不再轉送。還有時間戳記是用來更新封包序號紀錄，決定是否可以刪除。

4-7-4 計算出最短路徑及更新路由表

當每一路由器收到其他所有路由器的鏈路狀態封包，必須計算出它到達任何一部路由器的最佳路徑。在圖形理論中，有許多尋找最短路徑的演算法，較被常用的是 Dijkstra's shortest path algorithm。其演算法如下

```
## Dijkstra's shortest path algorithm
Define:
    N: set of all nodes such that shortest path from source to these nodes is known.
       N initially is empty.
    D(v): cost of known least cost path from source to node v.
    c(i, j): cost of link i to j, c(i, j)=∞ if i, j not directly connected.
    p(v): previous node (neighbor of v) along shortest path from source to v.
Algorithm:
    source node: A
    iterative: after k steps know shortest path to nearest k neighbors
    (1) Initialization:
            N = {A}
            For all nodes v
                If v adjacent to A
                        Then D(v) = c(A, v)
                 Else D(v) = ∞
    (2) Loop
            find w not in N such that D(w) is smallest
            add w into N
            update D(v) for all not in N:
                D(v) = min(D(v), D(w)+c(w, v))
        Until all nodes in N
## end of algorithm
```

我們用圖 4-11 為範例，每一路由器所收到的鏈路狀態封包如圖 4-12 所示。又以路由器 A 為例，尋找最短路徑的步驟如圖 4-13 所示。所有路由器演算法演算後所建立的路由表如圖 4-14 所示。

Step

圖 4-11 以路由器 A 為範例之演算過程

路由器 A		路由器 B		路由器 C		路由器 D		路由器 E		路由器 F	
目的地	下一站	目的地	下一站	目的地	下一站	目的地	下一站	目的地	下一站	目的地	下一站
A		A	A	A	D	A	A	A	D	A	E
B	B	B		B	B	B	B	B	D	B	E
C	D	C	C	C		C	E	C	C	C	E
D	D	D	D	D	D	D		D	D	D	E
E	D	E	D	E	E	E	E	E		E	E
F	D	F	D	F	E	F	E	F	F	F	

圖 4-12 所有路由器之路由表（以圖 4-11 為例）

4-7-5 鏈路狀態的異常狀態

鏈路狀態之路徑選擇方法雖然簡單，但會有下列異常狀態：

(1) **廣播風暴（Broadcast Storm）：**網路上每一個路由器必須隨時計算相鄰之間的鏈路費用，並廣播給網路上所有的路由器。如果每次廣播的時間間隔太長，則網路上任何變更將無法即時反映給所有路由器，會造成連結上困難；但廣播間隔時間太短容易造成廣播風暴。而且為了廣播路徑狀態給所有路由器也會佔用不少頻寬。

(2) **報喜不報憂：**對每個路由器都是主動廣播它所計算的鏈路狀態。任何路由器啟動時都會按時發送鏈路狀態。但當中某一路由器發生故障，它便失去廣播狀態之能力；其它路由器要能偵測出這一條鏈路是故障的，這可能要經過一段不短的時間，在這期間內也容易造成連結上的錯誤。

當然，上列的異常狀況有許多方法來克服，還不至於太困難。但一般來說，任何一個路由器要將它的鏈路狀況廣播給網路上所有路由器，在較大的網路上使用也太不夠經濟，何況，網路上路由器愈多，所佔用的頻寬也就愈大。

4-8距離向量路徑選擇

『距離向量路徑選擇法』(Distance Vector Routing，DV Routing) 是一種動態的分散式路徑選擇演算法。由此演算法所屬的路由器，路由表是由鄰近相鄰路由器共同建立而成，因此稱之為**『分散式』**。網路上每一個路由器都必須維護一個二維的**『向量表』(Vector Table)**，向量表格內紀錄本身路由器到每一個路由器的已知的最佳距離。路由器定期和鄰近路由器交換向量表（並不廣播給所有路由器）來建立路由表；當路由器接收到鄰近傳來的向量表時再修正本身的向量表，向量表的內容就一直修正再傳送，整個網路狀況因而能夠漸漸地傳遞到每一個路由器上。隨著時間路由器上之路由表的資料漸漸完備，也逐漸能找出最佳路徑。由於向量表只傳送給相鄰的路由器，對網路的頻寬佔用較少，所以不會造成廣播風暴。

對於向量表中各個欄位，如果使用路由器之間距離的數值，稱之為**『距離向量』(Distance Vector)**。表示兩個路由器之間相距幾個路由器，向量表又稱為**『距離表』(Distance Table)**。但目前使用之 DV routing 並不只估計相鄰之間的距離，而是將各個路由器之間的佇列延遲、頻寬和網路壅塞情況來計算相鄰路由器間的**『距離費用』(Distance Cost)**。假設路由器已知它相鄰路由器的**『距離』**，如果距離費用是封包跳躍次數，則其值剛好為 1；如果距離費用為佇列長度，則路由器僅需計算每個佇列；如果距離費用是延遲時間，則路由器可利用一個特殊的 Echo 封包要求鄰近路由器回應，再計算要求和回應之間時間的差距，取它的一半值就是延遲時間。

4-8-1 距離向量演算法的推演

我們以圖 4-13 為範例來說明路由器之間距離向量的傳遞，和路由表建立的過程。在圖中兩個端點（如，A、B）之間的標示值（6），為該兩端點之間的距離費用，其值的來由也許經過：跳躍次數、延遲時間等等因素所計算出來的。並假設距離向量傳遞路徑為：A → B → C → D → E。當距離向量進入某一路由器後，便和本身的路由表搜尋出所有可能到達的路徑，再由新建立的路由表之中尋找出最短路徑。緊接著，又將該最短路徑路由表傳遞給下一個路

由器。一直到最後，觀察路由器 E 的路由表的變化結果，便可瞭解距離向量演算法的運作情形。

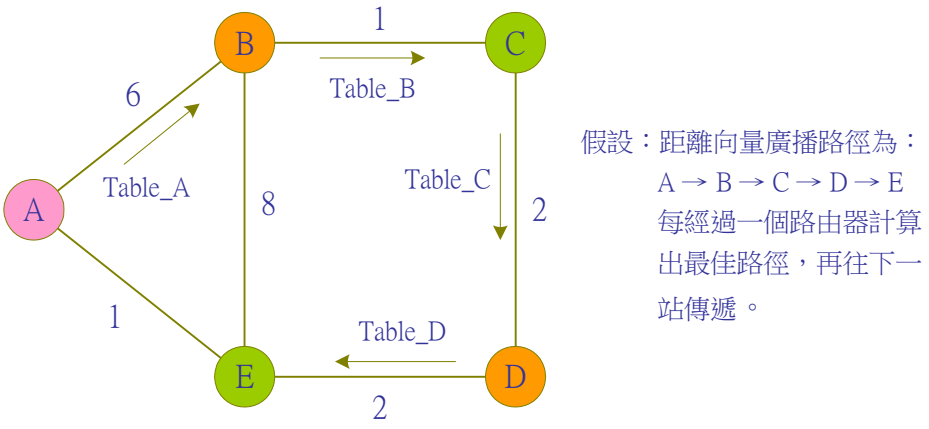


圖 4-13 距離向量演算法範例

圖 4-14 (a) ~ (e) 為各個路由器的起始距離向量表，並假設所有路由器都未和其他路由器交換訊息。針對每一個向量表，我們記錄了：從該路由器到其他路由器所經由不同相鄰路由器的費用。例如 DE(A, B) 是從 E 經由 A 到 B 的費用。其他空白部份表示沒有訊息，也可說是路徑費用無限大 (∞)。

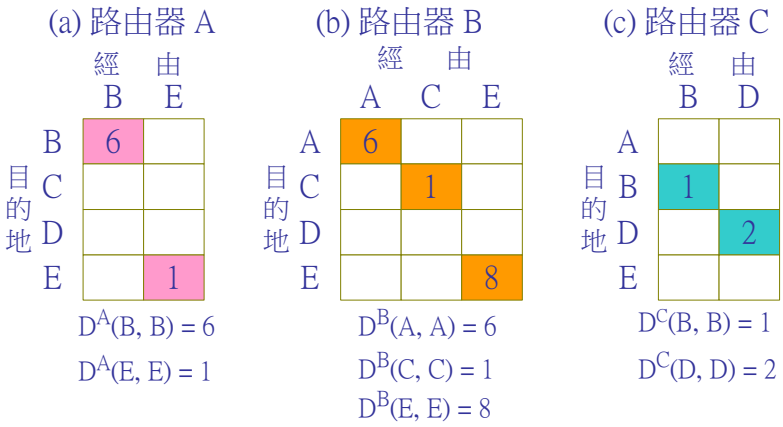


圖 4-14 (a) ~ (c) 路由器的起始路由表

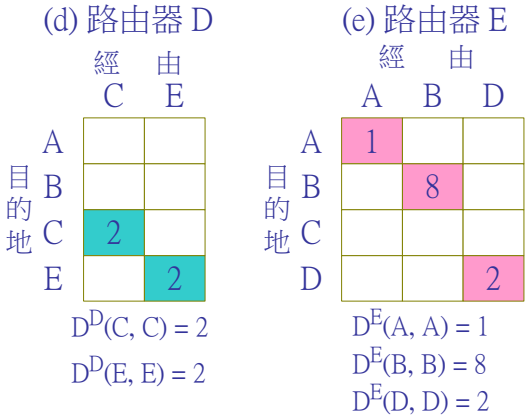


圖 4-14 (d)、(e) 路由器的起始路由表

圖 4-15 為向量表傳遞後，路由器依照進入的向量表和本身路由表，建構新的路由表，其步驟如下：

(1) 路由器（如 B）首先登錄自己和進入向量表的路由器（如 A）之間的費用（DB(A, A) = 6）。

(2) 再搜尋向量表（如圖 4-14 (a)）中可能到達的端點（其值不是 ∞），兩個路徑費用的合如小於表中內所紀錄的值，便取代它。

(3) 搜尋索有路徑後，建立新的路由表，再由新的路由向量表找出最短路徑，再將其傳遞給下一個路由器。

我們以圖 4-13 為例子，來推演距離向量演算法建構新路由表的過程，而距離傳遞方向假設只有單一路徑，其方向為： A → B → C → D → E。我們以下列步驟來觀察推演的過程：

(1) 路由器 A 沒有收到其他路由器的向量表，依照自己向鄰近路由器查詢之距離費用（如 4-14 (a)）建構出路由表。如圖 4-15 所示，最短路徑是由距離向量表中，查出每列的向量值最低者，並將其建立成路由表。如圖中，它可經由 B 到達 B 的費用是 6（$D^A(B, B)$）為該列最小值；又 $D^A(E, E) = 1$ 也是該列最小值。路由器 A 建構路由表後，再將該路由表傳遞給 B（A → B），緊接著下一步驟。

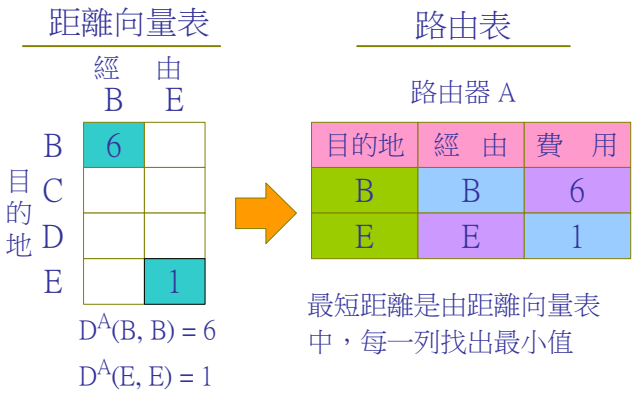


圖 4-15 路由器 A 自行建構路由表

(2) (A → B) ：路由器 A 將路由表 (圖 4-15) 傳遞給路由器 B，B 起始距離向量表如圖 4-14 (b)，再利用這兩個向量表建立出新的距離向量表。首先，B 由路由表中查出它自己經由 A 到 A 的費用 (DB(A, A) = 6) 。再由 A 的向量表中查詢可能到達的端點 (B、E) ，得知可經由 A 到達 E (DA(E, E) = 1) 。則將兩個端點路徑費用的和 (DB(A, A) + DA(E, E) = 7) 填入經由 A 到 E 的欄位內。再由表中的每一列的最小值，找到最短路徑 (如第四列) ，建構出路由表如圖 4-16 所示。並將路由表傳遞給 C (B → C) ，接下一步驟。

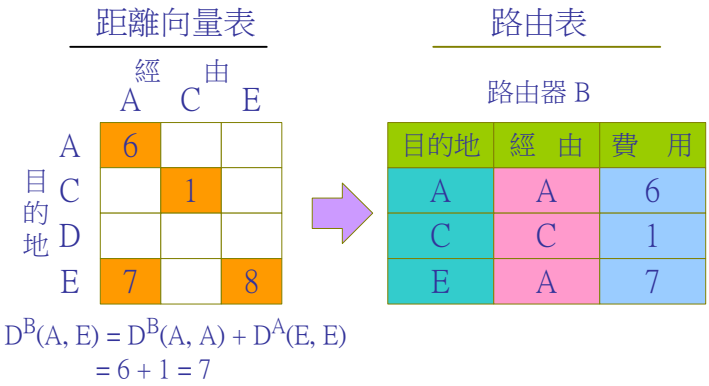


圖 4-16 路由器 B 經過 (A → B) 後建立新的路由表

(3) (B → C) ：路由器 B 將路由表 (圖 4-16) 傳遞給路由器 C，C 起始距離向量表如圖 4-14 (c) ，再利用這兩個向量表建立出新的距離向量表如圖 4-17 所示。首先 C 由本身路由表中查詢出到達路由器 B 之間的費用(DC(B, B) = 1)，在搜尋 B 的向量表可能到達的端點 (A、E) 。

- 到達 A (經由 B 到達 A) : DC(B, B) + DB(A, A) = 1 +6 = 7，原向量表中的值為無限大 (∞)，因而取代為 7。

- 到達 E（經由 B 到達 E）：DC(B, B) + DB(A, E) = 1 +7 = 8，填入表格。
- 搜尋完後所建立之向量表，再找出最短路徑，建立出新的路由表，如圖 4-17 所示。並將路由表傳遞給 D（C → D），接下一步驟。

圖 4-17 路由器 C 經過（B → C）後建立新的路由表

(4) （C → D）：路由器 C 將路由表（圖 4-17）傳遞給路由器 D。又 C 的起始距離向量表如圖 4-14 (d) ，再利用這兩個向量表建立出新的距離向量表。而 D 到 C 的費用是 2（DD(C, C)），另由向量表（圖 4-17）中查詢出可經由 C 到達之端點為 A、B、E。搜尋步驟如下：

- 到達 A：DD(C, A) = DD(C, C) + DC(B, A) = 2 + 7 = 9。
- 到達 B：DD(C, B) = DD(C, C) + DC(B, B) = 2 + 1 = 3。
- 到達 E：DD(C, E) = DD(C, C) + DC(B, E) = 2 + 8 = 10。

搜尋完後所建立之向量表及所建構出新的路由表如 4-18 所示。再將它的路由表傳遞給 E，緊接下一步驟（D → E）。

圖 4-18 路由器 D 經過（C → D）後建立新的路由表

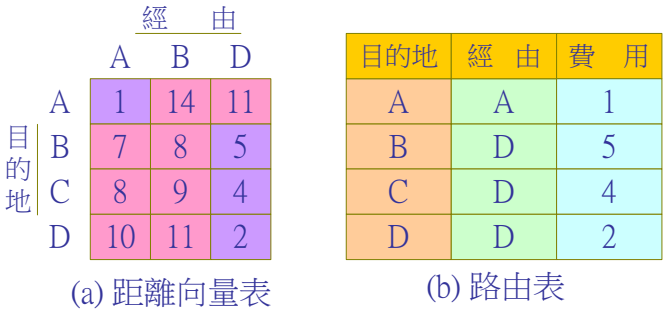
(5) （D → E）：路由器 D 將路由表（圖 4-18）傳遞給路由器 E。又 E 的起始距離向量表如圖 4-14 (e) ，再利用這兩個向量表建立出新的距離向量表。搜尋步驟如下：

- 到達 A：DE(D, A) = DE(D, D) + DD(C, A) = 2 + 9 = 11。
- 到達 B：DE(D, B) = DE(D, D) + DD(C, C) = 2 + 3 = 5。
- 到達 C：DE(D, C) = DE(D, D) + DD(C, C) = 2 + 2 = 4。

搜尋完後所建立之向量表及所建構出新的路由表如 4-19 所示。

圖 4-19 路由器 E 經過（D → E）後建立新的路由表

以上所推演的過程之中，只有單一路徑 (A → B → C → D → E)。當然，在正常情況下，路由器會隨時交換它們之間的距離向量表。假設，它們之間的路徑費用沒有改變，路由器 E 的向量表和路由表將如圖 4-20 (a)、(b) 所示。



經 由

目的地	A	B	D
A	1	14	11
B	7	8	5
C	8	9	4
D	10	11	2

(a) 距離向量表

目的地	經 由	費 用
A	A	1
B	D	5
C	D	4
D	D	2

(b) 路由表

圖 4-20 路由器 E 的距離向量表和路由表

但是如何來建立和更新路由表？一般我們會使用圖形理論中的分散式-非同步的最短路徑演算法。較常用的是 Bellman-Ford Algorithm。

```
## Bellman-Ford algorithm (at node X)
    1. Initialization:
       for all adjacent nodes (column) v
            D(*, v) = ∞
            D(v, v) = c(X, v)
    2. Loop
                    Execute distributed topology update procedure
       Forever
## end of Bellman-Ford algorithm
## Topology Update Algorithm
    At node X:
        1. wait (until I see a link cost change to neighbor Y, or until I receive control
           message from neighbor W)
        2. if (c(X, Y changes by δ)
             /* change in cost to my neighbor, Y    */
             change all column-Y entries in distance table by δ
             if this changes cost of least cost path to some node Z, send control
                 message to neighbors with my new minimum cost to Z.
        3. if (control message received from my neighbor W)
             /* shortest path via W to some node Z has changed */
             DX(Z, W) = c(X, W) + new distance from W to Z
             If cost of my least cost path to Z has changed send control message to
                 neighbors with my new minimum cost to Z.
** end of Topology Update Algorithm
```

4-8-2 距離向量的迴路問題

距離向量之路徑選擇法雖然較適合於大網路上使用，但也可能造成兩種異常狀態：『**迴路問題**』（**Looping**）和『**震盪情形**』（**Oscillations**）。我們先來討論迴路問題，其最主要的現象是：好消息傳的快，壞消息傳的慢。當路由器啟動時會即時計算本身的向量表，並向鄰近路由器通告，使整個網路迅速建立起來。網路使用一段時間後，各個路由器都已尋找出最佳路徑到其他路由器。但當中有任何路徑發生故障，必須再靠路由器之間的傳遞來發現路徑不通，重新計算最短路徑可能必須要一段傳遞時間。如圖 4-21 為好消息傳得快的範例，假設起始狀態之前端點 A 和 B 之間斷線，因此，各端點到 A 之間的跳躍距離為無限大（∞）。當第一次傳遞時，B 到 A 之間距離向量就被設定為 1。經過四次傳遞後，便可以將 A 路徑恢復的消息傳遞給所有端點。

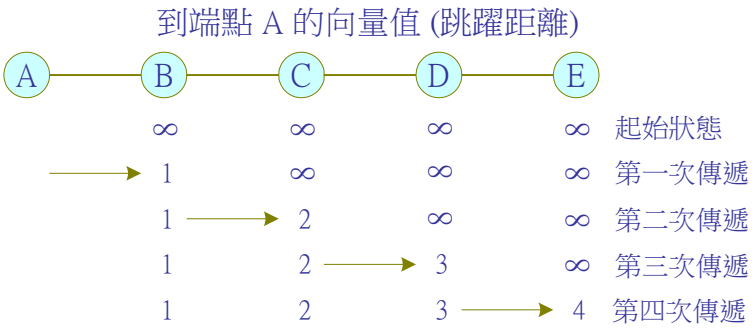


圖 4-21 迴路問題範例（好消息傳得快）

圖 4-22 為壞消息傳得慢的範例，起始狀態是所有路徑都正常，各端點上也都有紀錄前往端點 A 的距離向量（跳躍距離）。當 A 到 B 之間的路徑已斷線第一次傳遞時， B 無法傳送到 A，而將距離設為無限大，但它將這個訊息告訴 C，但 C 的紀錄裡有一路徑可到達 A 其向量值為 2，並將該值傳給 B，B 因而設定其前往 A 的向量值為 3。第二次傳遞，B 告訴 C 經由它那裡到達 A 的路徑為 3，因此，C 又將其到達 A 的向量值改為 4。第三次，C 將它到達 A 的向量值傳遞給 B 和 D，它們又將前往 A 的向量值改為 5 和 5。依此類推，如果要將 A 斷線的消息傳遞給所有端點，也許需要無止境的迴路。

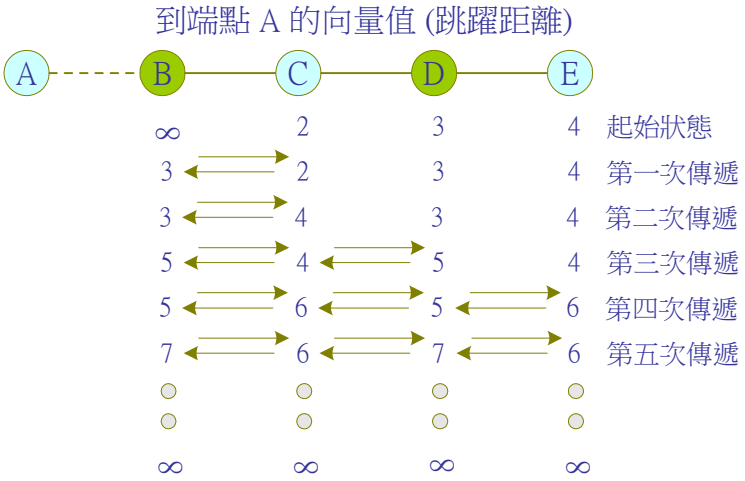


圖 4-22 迴路問題範例（壞消息傳得慢）

解決迴路問題在許多文獻中提出各種解決方法，但以下列兩種方法較為常用：

(A) 水平分離切口（Split Horizon Hack）

路由器在發送距離向量表時，都先假設某一邊路徑已有斷線的可能，而去試探它。做法如下：如果路由器往某一路徑是最佳路徑，下一次發送向量表時，發送給該端點之路徑費用設定為無限大，而另一方向以正常向量值發送。宛如，水平切開兩邊缺口。如果對方路徑還是正常，下次發送正常向量值；如果已斷線就給無限大的向量值。在圖 4-23 之中，端點 C 送前往 A 端點的向量值無限大給 B，而以正常的向量值 2 給另一邊的 D。

如果當時 A 已斷線，B 找不到最佳路徑，就將該向量值設定為無限大，並以下次傳送給 C。第二次切離時，D 將前往 A 的向量值設定為無限大並傳給 C，另一方面，它以正常向量值 3 傳給 E。端點 C 收到該向量值後，也找不到另一路徑可到達 A，便將該向量值設定為無限大。依此類推，每分離切割一次，就將 A 斷路的訊息往前發佈一個端點。也可以避免產生向量值傳遞迴路的問題。

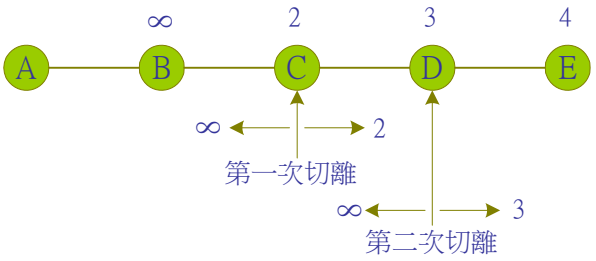


圖 4-23 水平分離切口法範例

(B) 暫停（Hold Down）

當某一路由器發現路徑故障，首先將故障路徑的費用設定為無限大，通知鄰近路由器。等待一段時間後，接到其他的向量表，再依照當時情況決定最短路徑。

4-8-3 距離向量表震盪情形

當路徑費用是以資料流量大小來評估的情況下，路由器發現某一路徑傳輸流量較低，則將該路徑的向量值設定為最低，並發送給相鄰路由器。因此，所有的傳輸訊息便往該路徑發送，這個路徑費用又變得最高。路由器又將這個訊息告訴相鄰路由器，它們又停止往這個路徑傳送，這條路徑費用又變成最低。依此類推，這條路徑費用就一直震盪（Oscillations）不停。解決方法是：(1) 不要定期的交換路徑資訊。交換訊息的時間加長，以減少震盪的現象。(2) 減低資料流量對路徑費用的比率。以減少資料流量瞬間改變對整個路由表的影響。

習題

1. 請敘述網路層（Network Layer）的功能。
2. 請問網路層所提供服務類型為何能影響整個網路的架構？針對區域網路和大型網路的應用，網路層所提供的服務類型有何關聯？
3. 何謂電路交換技術（Circuit Switching）？請敘述其特性和功能。
4. 何謂訊息交換技術（Message Switching）？請敘述其特性和功能。
5. 何謂分封交換技術（Packet Switching）？請敘述其特性和功能。
6. 何謂電報傳輸（Datagram）？請敘述其特性和功能。
7. 請比較交換技術和電報傳輸有何優異？及它們適用的環境？
8. 何謂 Next-hop Routing？請敘述其特性和功能。
9. 何謂廣播風暴（Broadcast storm）？在何種情況下會發生？
10. 依照下圖之網路拓樸型態，請繪出各路由器的路由表。

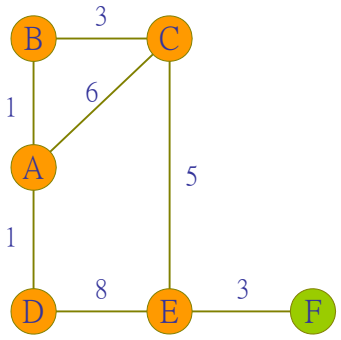


圖 4-22 網路拓樸圖範例

11. 何謂熱馬鈴薯（Hot-potato）路徑選擇法？請敘述其特性和功能。

12. 何謂靜態路徑選擇法（Static Routing）？有哪兩個主要方法。

13. 請簡述鏈路狀態路徑選擇法（Link-State Routing）的運作情形？

14. 請依照圖 4-22，使用鏈路狀態演算法推算出端點 E 的最短路徑，並建構端點 E 的路由表。

15. 請問鏈路狀態路徑選擇法可能發生哪些問題？應如何克服？

16. 請簡述距離向量路徑選擇法（Distance-Vector Routing）的運作情形？

17. 請依照圖 4-22，使用距離向量演算法推算出端點 E 的最短路徑，並建構端點 E 的路由表。

18. 請問距離向量路徑選擇法可能發生哪些問題？應如何克服？