第四章

純 ATM 型架構之 MAC 層協定

純 ATM 型架構的主要目標是將 ATM 骨幹網路擴展到無線領域及提升系統效能。與混和型架構相同，系統效能的提升方式仍將重點放在減少隱藏用戶對傳送者的影響，以減少不必要的通道浪費。此外，因為 ATM 網路著重在維持各用戶的服務品質在一定程度之上，當系統由有線網路擴展到無線領域，這個要求仍是最終目標，因此在純 ATM 型架構下，如何保證各用戶的 QoS 要求是另一個重點。雖然在 ATM 網路內碼格是基本單位長度，但如果每次獲得通道使用權後只傳送一個碼格加上 MAC 層及實體層所需之標頭會造成更多的 overhead，因此在 MAC 層下，仍需將多個碼格合成一個 MPDU，也就是將一連串的碼格合成一個“碼格串”（cell train）並加上標頭所形成的 MAC 層傳輸單位，見圖 3.1，以提升系統效能。

4.1 環境架構

在純 ATM 型環境下，網路架構類似混和型環境仍有 BSS、ESS 及 DS 的觀念，但此時使用者全為 ATM 用戶，而 DS 則由 ATM 網路構成，如圖 4.1。值得一提的是由 ATM 網路形成的 DS 只限於提供 ESS 內部資料傳送的服務，若要連結

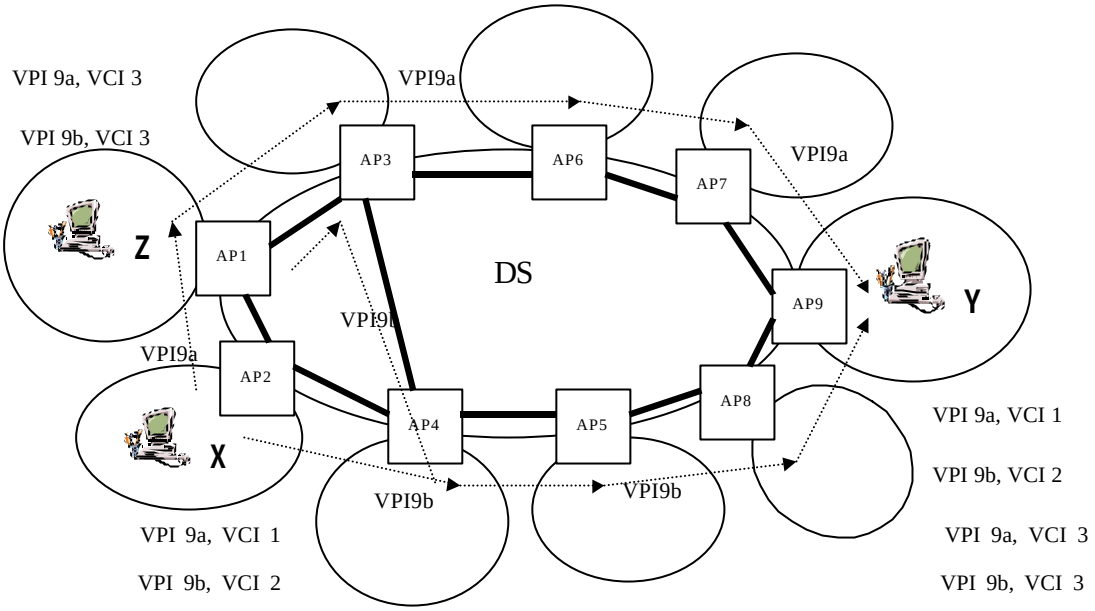


圖 4.1 **無線** ATM VP/VC **概念**

到外部網路仍須經由入口埠接轉。DS **內部框架傳送途徑的資訊會放在每個碼格標頭的** VPI/VCI **欄位內，但不同於一般** ATM **網路的** VPI/VCI **值會在各路由器（**router**）之間不斷替換，**DS **內部的** VPI **是指向某一特定的** AP**。多個** VPI **可能同時分配給一個** AP**，就如同以** AP **為樹根，每一個** VPI **為分散的樹枝各走不同的路徑，但源頭皆相同。另一方面，**VCI **則是用來區分在相同** VPI **路徑上傳送的不同連線。如圖** 4.1**，使用者** Z **連線到使用者** Y**，使用者** Z **可選擇從** VPI9a **或** VPI9b **將資料送達** AP9**，而使用者** X **到使用者** Y **的連線也可由** VPI9a **或** VPI9b **到達** AP9**。由圖可知** VPI9a **及** VPI9b **皆是指向** AP9 **的** VPI**，但路徑不同。為了區分** AP9 **所收到的** ATM **碼格是屬於** Z-Y **連線或** X-Y **連線，**AP9 **分配（**VPI9a**，**VCI3**）及（**VPI9b**，**VCI3**）給使用者** Z**，（**VPIa**，**VCI1**）及（**VPI9b**，**VCI2**）給使用者** X**。如此當** AP9 **收到標頭內的** VPI/VCI **為（**VPI9b**，**VCI3**）的** ATM **碼格時即可斷定此碼格是使用者** Z **傳送的**〔25〕**。**

使用者利用信號通道（signaling channel**），也就是每個** VPI **的** VCI 5**，傳送**〝setup〞 **框架要求建立連線。一旦連線成功，**AP **將分配給此使用者一個連線識**

別碼用以代替 ATM 碼格之 VPI/VCI 欄位，而在 DS 內部，AP 會決定好傳送路徑，此時碼格在 DS 內部傳送時 VPI/VCI 值不會改變。當 AP 接收到無線用戶傳送的碼格時需先重寫碼格標頭：加入 GFC 及 HEC 並將連線識別碼還原為 VPI/VCI。在 ATM 碼格還原成原本 53 個位元組的格式後，將碼格送入 DS 並根據 VPI/VCI 決定傳送途徑。當碼格來到 AP 端準備送往無線用戶端時，AP 再將碼格標頭縮短為 2 個位元組，如第三章所述，傳送到接收端。

在協定層級方面，由於所有的設備皆為 ATM 介面，因此沒有轉換的需要。圖 4.2 是使用者 Z 傳送資料到使用者 Y 的情況下之協定層級〔16〕，由圖 4.2 可以推演出其他情況的層級型態。

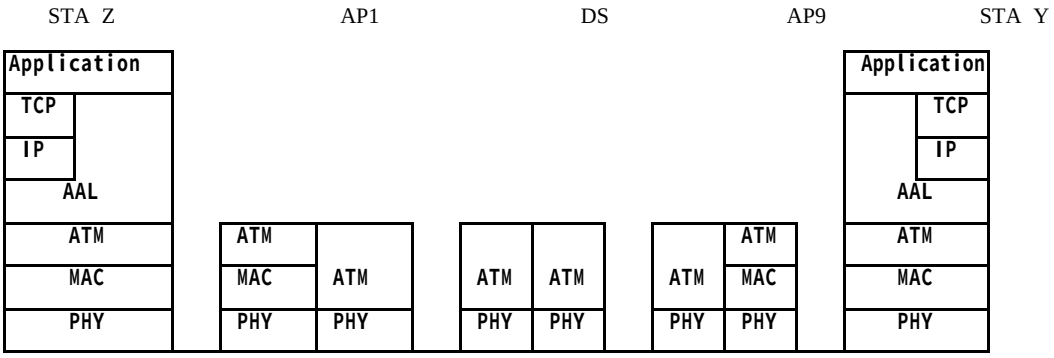


圖 4.2　ATM 用戶 Z — ATM 用戶 Y 之協定層級

4.2 協定架構

在純 ATM 環境下，協定的重點在於維持服務的品質，因此除了在 DCF 區間仍採用與混和型環境相同的協定外，PCF 週期則由排程器規劃傳送順序及數量，其超級框架的架構如圖 4.3。PCF 區間仍利用輪呼方式傳送資料，適用於上傳或

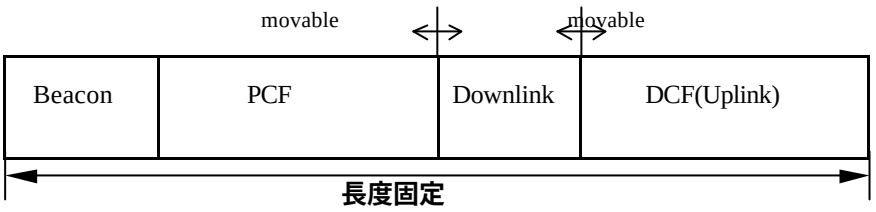


圖 4.3　純 ATM 型環境之超級框架架構

下傳之 CBR 及 rt-VBR 的碼格。下載區（Downlink）則是供 AP 下傳 nrt-VBR、ABR 及 UBR 之碼格。在收到 AP 的下傳資料後，接收者只能回覆 ACK 框架給 AP，不可兼送資料。另外，需要一個信號通道讓使用者傳送各種管理框架，如〝setup〞、結合要求框架等訊息（information）給 AP。DCF 週期則是專供上傳的使用者傳送 ABR 及 UBR 碼格，利用與混和型架構相同的協定由 AP 控制 CSMA/CA 的運作。至於上傳之 nrt-VBR 碼格傳送區域將在 4.3 節介紹。每個超級框架至少保留一個最大長度的 MPDU 及 RTS/CTS 交換和接收 ACK 所需時間給 DCF 週期，再扣除 PCF 區間所花費的時間，剩餘時間就留給 AP 下傳框架之用。如果下傳時間小於保留的時間，則在 AP 傳送出 CF-end 框架後提前開始 DCF 週期。

相同於混和型架構，超級框架的長度仍維持固定，以確保及時性交通型態的傳輸品質。由於每個超級框架皆需保留一段時間供 DCF 週期使用以保障非及時性交通型態的權力，因此此段時間的長短將會影響及時性交通型態的傳送時間。在此，為了讓及時性交通型態得到較好的服務，選擇加長 CFP 最大區間的時間，即縮短 MPDU 最大長度值（在 IEEE 802.11，最大長度的 MPDU 為 2346 個位元組。）的方式來達到目的。但是一旦縮短 MPDU 最大長度， SIFS 等的影響就會浮現，因此每次競爭通道成功後傳送的 MSDU 資料量應該多長才可讓系統效能維持在一定水準之上則是壓縮 MPDU 長度時應該考量的問題。圖 4.4 以最大 MPDU 長度為 684 個位元組（即 MPDU 負載由 13 個碼格組成）為例，其他參數與 3.3.1 相同，並利用相同的分析方法探討 MSDU 資料量與系統貫通率的相關性。從圖中可以看

出，當 MSDU 由 4 個 MPDU 構成時，系統效能就可在一定程度之上。

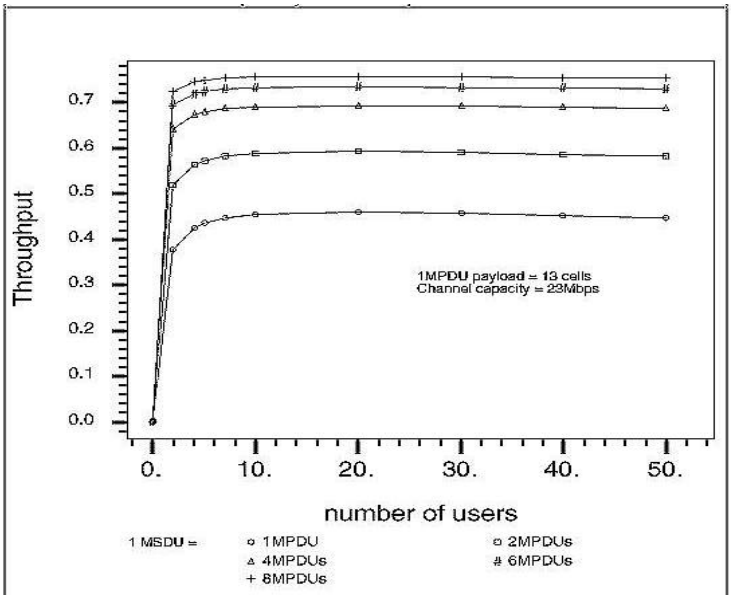


圖 4.4　純 ATM 架構中 MSDU 長度之比較

4.3 純 ATM 型 PCF

純 ATM 型 PCF 仍是利用輪呼方法控制各個連線的傳送順序及傳輸量的大小，而傳送順序及傳輸量之規劃必須靠排程器(Scheduler)來安排。因此 AP 必須包含兩個機制：排程器及執行器。排程器每隔一段時間送出一個已規劃完成的超級框架給執行器，執行器即利用這些資料依序呼叫各個連線，並告知每次傳送最多可送出多少碼格。在此排成器的基本觀念是盡可能使 MPDU 的傳輸時間越靠近期限（deadline）越好（在此的期限是指由用戶端到 AP 端的期限），以下我們將採用 Nikos Passas 等人所提出的 MASCARA 概念架構我們的排程器〔26〕。

排程器要規劃各個連線的碼格位置就必須先知道各連線所產生的碼格個數，因此 AP 必須去預估每個上行連線的碼格產生量：在 AP 端相對於每一個上行連線

會有一個暫存器（buffer），若是 CBR 連線，AP 會以 CBR 的碼格產生速率（cell rate）產生碼格在其專屬的暫存器內。若是 VBR 連線，由於 VBR 連線在每次送出一個 MPDU 時會將從上一個 MPDU 結束到此 MPDU 結束間到達的碼格數 piggyback 到 AP 端，因此每當 AP 收到 piggyback 的值，就會在其專屬的暫存器內產生相同數目的碼格，且這些碼格的期限皆為上個 MPDU 結束的時間加上從用戶端到 AP 端的 CDT(cell delay tolerance)，以後簡稱 CDT。而 VBR 連線所屬暫存器之大小為此 VBR 的最大叢發尺寸（max. burst size），因此當碼格的量超過暫存器的大小時，需將過量碼格的 CLR 設為 1，當有足夠的空間時這些 CLR＝1 的碼格仍可送出，若空間不夠將會先被丟掉。除此之外，若某 VBR 連線送出一 MPDU 時，piggyback 的值為 0 且其在 AP 專屬暫存器內已沒有碼格存在，這時之後到達的碼格就無法利用 piggyback 讓 AP 知道它們的存在。因此在這種情況下當有碼格出現時，此 VBR 連線必須再等待（1/2）CDT 後在信號通道送出一個管理框架告知 AP 這段時間內來了多少碼格，此時 AP 即產生相同數目的碼格在 AP 端相關的暫存器內，其期限為（第一個碼格到達時間＋CDT）。當由這些碼格組成之 MPDU 送出後，piggyback 的值為在使用端暫存器內的碼格數，而其期限為（管理框架到達時間（arrival time）＋CDT）。在知道碼格產生的量後必須考慮排程的順序：首先給予不同等級的連線不同的優先權（priority）：CBR 的優先權最大，其次為及時性 VBR，最後為非及時性 VBR。接下來考量安排順序：

（1）當有不同等級的碼格必須安排時，先安排優先權大的。

（2）當有同等級的碼格必須安排時，先安排暫存器內有最多碼格的。

也就是說排程器一般時間是在規劃 VBR 的碼格，先將及時性 VBR 所有暫存器內有碼格的連線按碼格的多少由大排到小，如此一來排程器的規劃順序就由有最多碼格的連線開始，依序往碼格數少的方向前進，一次規劃一個碼格，結束後開始規劃非及時性 VBR 的碼格，同樣由碼格數多的到碼格數少的方向規劃。當完成之後再由及時性 VBR 開始，一直循環。但若在開始要規劃某 VBR 連線的碼格前發現有某個 CBR 連線的暫存器有碼格產生，則需先規劃此 CBR 連線的碼格，再回頭規

劃 VBR 的碼格。

在介紹排程器工作原理之前有四個觀念必須提出：

（1）從排程器的觀點來看，PCF 是由一連串的時槽（slot）組成，時槽的大小為傳送一個碼格所需之時間。排程器會將同一連線的碼格排在一起，形成一個 MPDU。由於使用者每次傳送一個 MPDU 之前需先由 AP 呼叫，在過了 SIFS 的時間後才能傳送 MPDU，再過 SIFS 後由 AP 回覆 ACK，因此排程器在規劃一個 MPDU 時必須將這些時間考慮進去，如圖 4.5。因為排程器在安排一個 MPDU 時是一個碼格一個碼格的安排，所以在安排 MPDU 負載中的第一個碼格時必須將每一個上行交通（traffic）所需的額外時間：1 個 MPDU 標頭，2 個 SIFS，1 個呼叫框架及 1 個 ACK 框再加上實體層標頭，或每個下行交通所需的額外時間：1 個 MPDU 標頭，2 個 SIFS 及 1 個 ACK 框再加上實體層標頭，安排進去，這是因為如果一開始就沒有空時槽可安排給這些一定會使用到的時間，那麼之後的 MPDU 也就沒有傳送的可能，因為它無法完成 handshaking 的動作。為了方便起見這些多出來的時間將以整數個時槽表示，假設其值為 ϕ，也就是說排程器在規劃 MPDU 負載中的第一個碼格時必需安排 $\phi + 1$ 個時槽給此連線，如此才能進行在 PCF 週期傳送一個 MPDU 所需完成的程序。

（2）因為 MAC 層架構的緣故，有些部分是不能讓排程器規劃的，如 Beacon 框架、CF-end 框架及 DCF 部分。一個超級框架開始時先假設 DCF 時間長度是固定的，為一個最大長度的 MPDU 加上 RTS／CTS handshaking 及 ACK 的時間，因此每次 PCF 週期開始時所有使用者先將 NAV 預設為一個超級框架長減去 DCF 區間長度，當 AP 呼叫完或傳送完所有規劃過的連線後會送 CF-end 框架給所有使用者，使用者在收到 CF-end 框架後即可重設 NAV 值，開始 DCF 時段，如圖 4.6。因此排程器在做規劃時須注意，若有碼格的期限落在 DCF 時段必須假設期限是在 DCF 開始之前；若有碼格的期限落在 Beacon 框

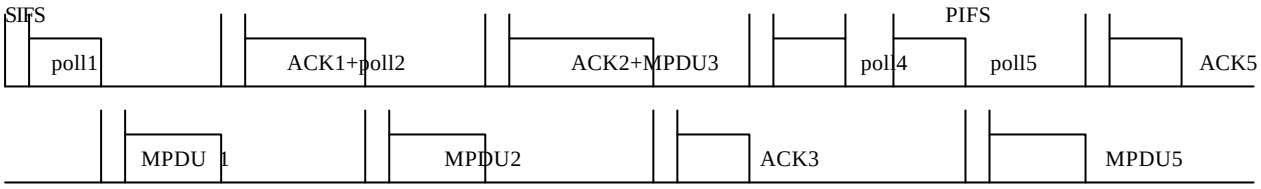


圖 4.5　PCF 構造圖

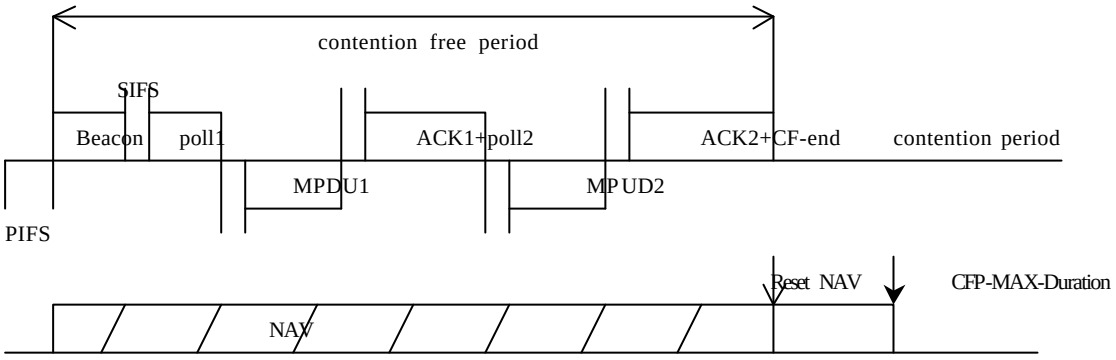


圖 4.6　PCF 與 NAV 之關係

架上則必須假設期限是在前一個超級框架 DCF 週期開始之前；若 MPDU 的期限在超級框架的第 i 個 slot (i≦φ)，則需假設此 MPDU 的期限在前一個超級框架 DCF 區間開始之前，如圖 4.7。

(3) 由於排程器是根據一個 MPDU 的期限來安排傳輸的順序，因此排程器一次可能同時規劃多個超級框架，但每當執行器中的超級框架快結束時，排程器就必須把排程順序中最前面的超級框架提出，並將位置適當調整後（之後會介紹）送給執行器執行，見圖 4.8。

(4) 由於上行的交通量是 AP 經由前述的方法預估而來，若遇到某些連線如語音只在發話狀態下以一定的速率傳送，會高估碼格的數量，此時 AP 可由用戶

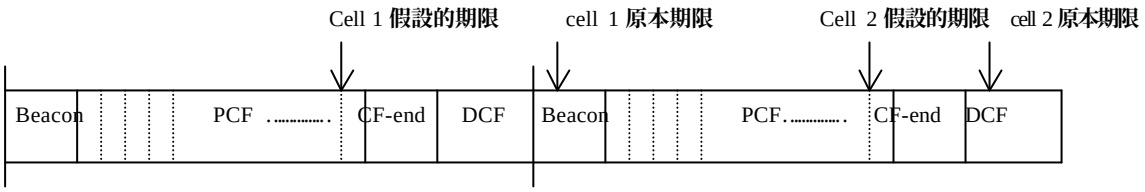
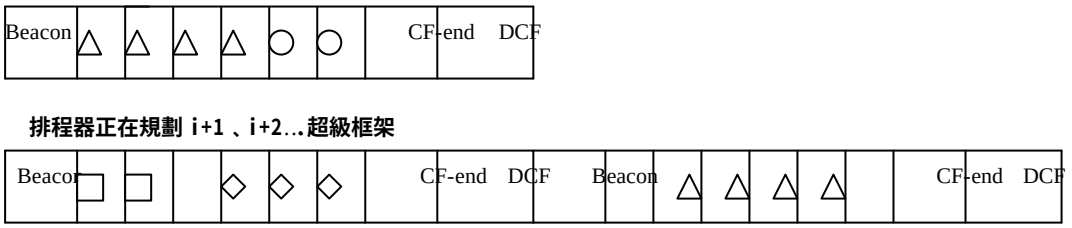


圖 4.7　超級框架與期限值之關係

(1) 執行器從排程器取得第 i 個超級框架的規劃，並開始執行。



(2) 當第 i 個超級框架快結束時，排程器取出第 i＋1 個超級框架給執行器並繼續規劃 i＋2 、i＋3・・・・超級框架

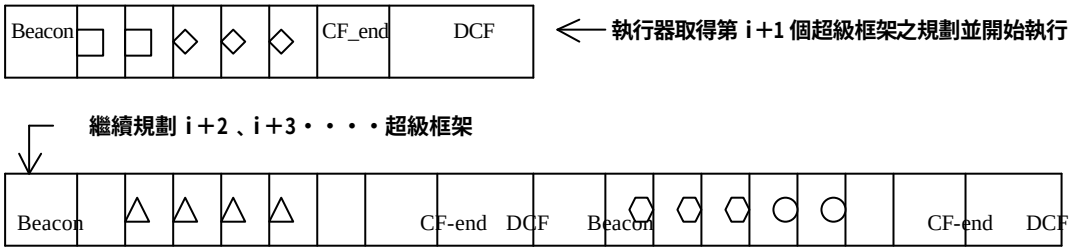


圖 4.8　排程器與執行器的關係

端送出的 MPDU 標頭中 Duration/ID 欄位得知真正的交通量，若比排程器規劃的少則可提前呼叫其他使用者以免浪費頻寬。另外，如果被呼叫到的使用者沒有碼格要送出則必須送出 NF 框架告知 AP，若有碼格要傳但在傳出碼格前發現此碼格已超過期限則可先丟掉此碼格，以免浪費頻寬在沒必要的傳送上。另一方面，為了減少錯誤重傳的次數，MPDU 的長度必須有一上限，若新來的碼格加入原本的 MPDU 會讓 MPDU 長度超過上限就必須將此碼格當成是另一個 MPDU 負載的第一個碼格重新做規劃。

以下將介紹排程器的工作原理：

（Ⅰ）首先介紹文獻〔26〕常用之符號：

1、$C_n(i)$為連線 C_n 的第 i 個 ATM 碼格。

2、$a(C_n(i))$為 $C_n(i)$的到達時間。

3、$CDT(C_n(i))$為從 MS 端到 AP 端可延遲的最長時間。

4、$d(C_n(i))=a(C_n(i))+CDT(C_n(i))$，即為 $C_n(i)$在不超過期限最晚該傳送出去的時間。

5、$D_n=\min\{d(C_n(i)): C_n(i)$是多個碼格所組成之 MDPU 中的一個碼格$\}$，因此此 MPDU 負載中第一個碼格的期限即為此 MPDU 的期限。

6、$F_n = \begin{cases} 0, & \text{若在此超級框架沒有時槽分配給} C_n \\ \text{分配給} C_n \text{的第一個時槽}, & \text{若有時槽分配給} C_n \end{cases}$

7、$L_n = \begin{cases} 0, & \text{若在此超級框架沒有時槽分配給} C_n \\ \text{分配給} C_n \text{的最後一個時槽}, & \text{若有時槽分配給} C_n \end{cases}$

8、$o(x) = \begin{cases} 0, & \text{若時槽} x \text{不被任何連線使用} \\ n & , F_n \le n \le L_n \end{cases}$

9、$D_n^- = \begin{cases} D_n, & \text{若} o(D_n) = 0 \\ F_{o(Dn)} - 1, & \text{若} o(D_n) \neq 0 \end{cases}$

10、$M_f(i)$為超級框架 i 最前面的（第一個）MPDU。

11、$M_l(i)$為超級框架 i 最後一個 MPDU。

12、$Bn(i)$為第 i 個超級框架從 DCF 始點往前算起第 n 個空的時槽，也就是 PCF 倒數第 n 個空的時槽。

（Ⅱ）排程器的工作原理：

這裡所提出的方法是沿用 MASCARA 排程器〔26〕的觀念而來，分成兩個步驟。

第一個步驟：安排 C_n 的碼格。

Case 1：考慮 C_n 在此 MPDU 的第一個碼格（MPDU 負載的第一個碼格）的安排，將

根據它的期限作安排。假設 C_n 的 overhead 佔 ϕ_1 個時槽，則在規劃 MPDU 負載中的第一個碼格時總共需安排 $\phi_1 + 1$ 個時槽。設 C_n 第一個碼格的期限在第 i 個超級框架。

（1）先安排第一個時槽：若時槽 D_n^- 是空的，排程器就將此時槽分配給 C_n，若不是空的則考慮以下幾個案例：

Case 1.a：若在此超級框架〔1，D_n^-〕時槽之間至少有一個空的時槽，想空出 D_n^- 的位置而不分割其他連線的碼格串，則將所有在 D_n^- 前最後一個空的時槽到 D_n^- 間的碼格往前移一個時槽，空出 D_n^- 給 C_n 使用。

若無法滿足 Case1.a 則檢查超級框架(i-1) ,超級框架(i-2)中空時槽的個數及 $M_f(i)$,$M_l(i-1)$,$M_f(i-1)$,$M_l(i-2)$ 各屬哪個連線，並考慮以下案例：

Case 1.b：若 $M_f(i)$,$M_l(i-1)$ 屬同一個連線且超級框架(i-1)空時槽的個數大於或等於 1，則將 B_1（i-1）至 CF-end 框架之前的碼格往前移一個時槽，並將 $M_f(i)$ 負載中的第一個碼格放於 $M_l(i-1)$ 負載中（如果將 $M_f(i)$ 負載中的碼格往前移後負載中已沒有碼格，則丟掉 $M_f(i)$。）。此時超級框架 i 會空出一個時槽，將超級框架 I 在 D_n^- 之前（含 D_n^-）的碼格往前移一個時槽即可空出 D_n^- 給 C_n，如圖 4.9。

Case 1.c：若 $M_f(i)$,$M_l(i-1)$ 屬同一個連線，而超級框架(i-1)空時槽的個數為 0，則考慮以下案例：

Case 1.c.1: 若 $M_f(i-1)$,$M_l(i-2)$ 屬同一個連線且超級框架(i-2)空時槽的個數大於或等於 1，則將 B_1（i-2）至 CF-end 框架之前的碼格往前移一個時槽，並將 $M_f(i-1)$ 負載中的第一個碼格放於 $M_l(i-2)$ 負載中（如果將 $M_f(i-1)$ 負載中的碼格往前移後 $M_f(i-1)$ 負載的長度為 0，即只剩 overhand，則丟掉 $M_f(i-1)$。）。此時超級框架(i-1)會空出一個時槽，其餘步驟見 Case1.b。

Case 1.c.2：若 $M_f(i-1)$,$M_l(i-2)$ 不是同一個連線，且超級框架(i-2)

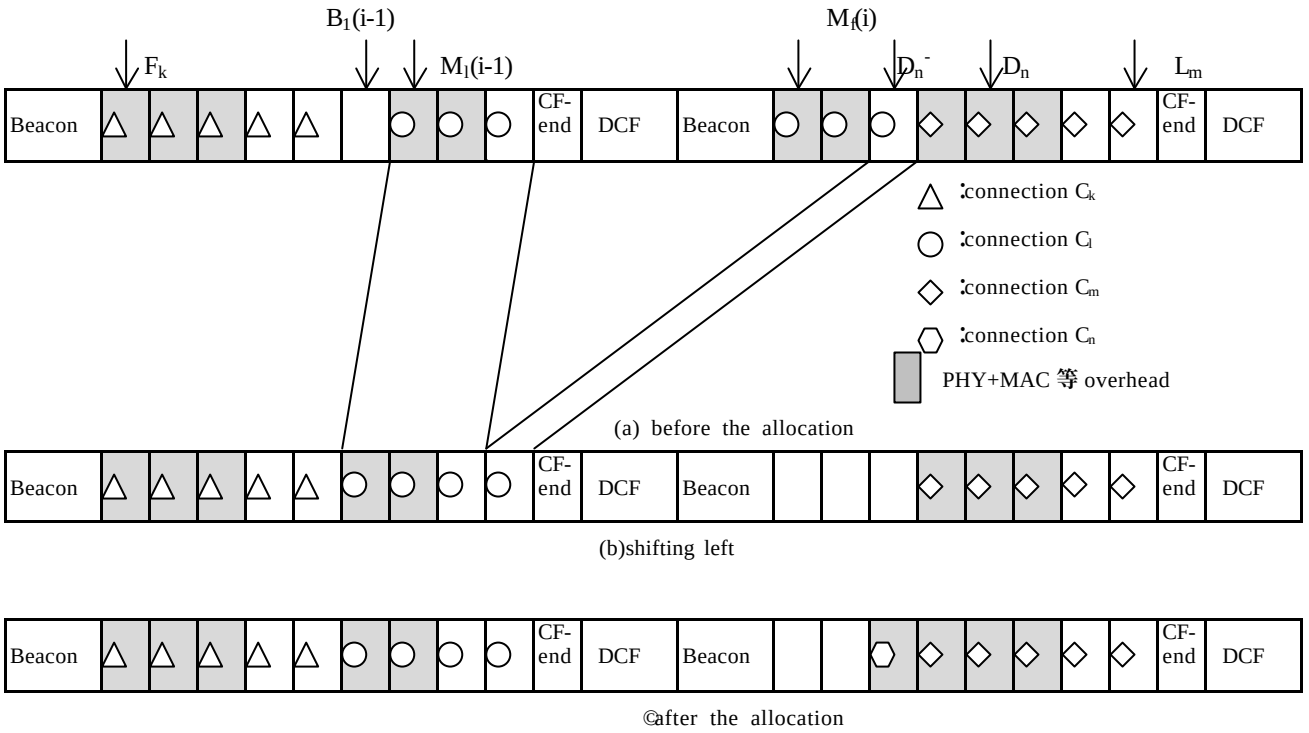


圖 4.9　$M_l(i-1)$ 、$M_f(i)$屬同一連線時的規劃

空時槽的個數大於或等於 $M_f(i-1)$的 overhead 長度加 1 即ϕ_3+1，則將〔$B_{\phi 3+1}(i-2)$,$B_{\phi 3}(i-2)$〕間的碼格往前移一個時槽，〔$B_{\phi 3}(i-2)$,$B_{\phi 3-1}(i-2)$〕間的碼格往前移 2 個時槽，依序前移後超級框架(i-2)在 CF-end 框架之前將會空出ϕ_3+1個時槽。將前ϕ_3個時槽安置與 $M_f(i-1)$相同的 overhead，最後一格空位放置 $M_f(i-1)$負載內的第一個碼格。(如果將 $M_f(i-1)$負載中的碼格往前移後負載的長度為 0，即只剩 overhand，則丟掉 $M_f(i-1)$。）此時超級框架(i-1)會空出一個時槽，其餘步驟參考 Case1.b。

Case 1.d：若 $M_f(i)$,$M_l(i-1)$不是同一個連線，且超級框架(i-1)空時槽的個數大於或等於 $M_f(i)$的 overhead 長度加 1 即ϕ_2+1，則將〔$B_{\phi 2+1}(i-1)$,$B_{\phi 2}(i-1)$〕間的碼格往前移一個時槽，〔$B_{\phi 2}(i-1)$,$B_{\phi 2-1}(i-1)$〕間的碼格往前移 2 個時槽，依序前移後超級框架(i-1)在 CF-end

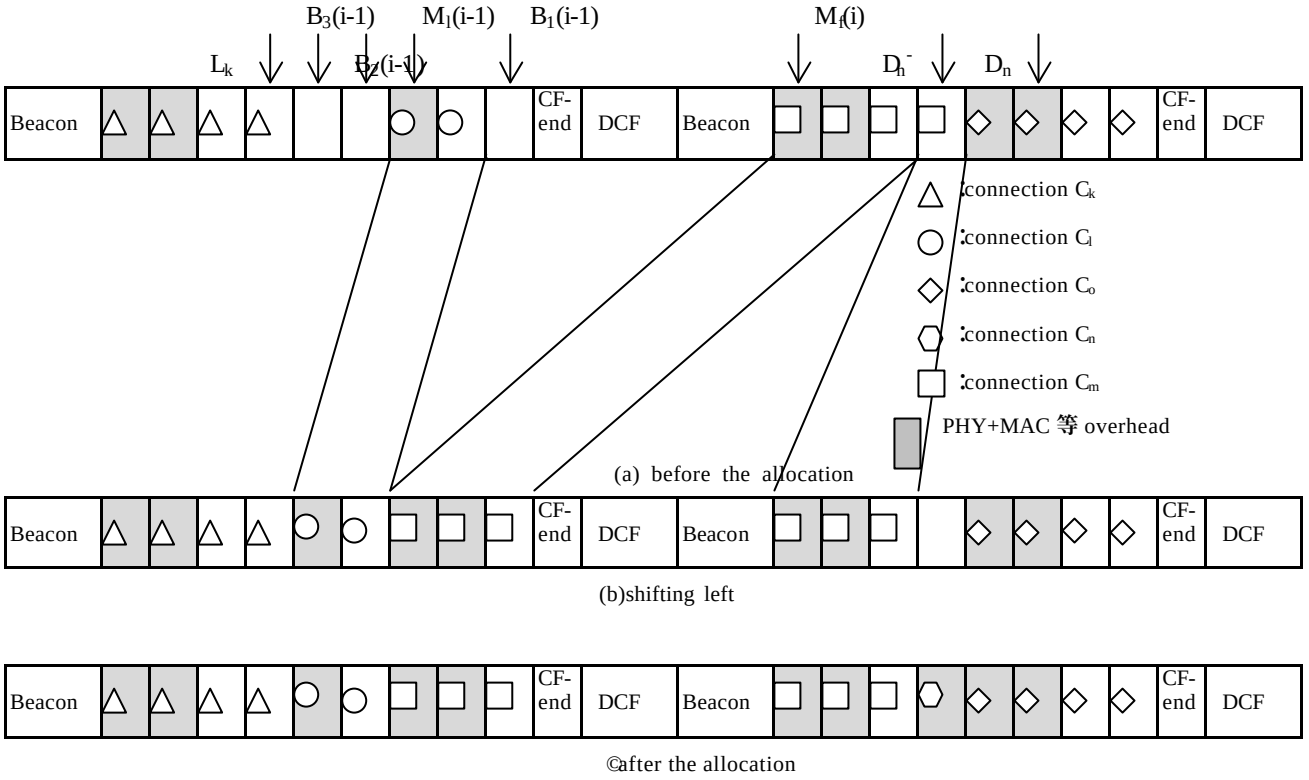


圖 4.10　$M_l(i-1)$ 、$M_f(i)$屬不同連線時的規劃

框架之前將會空出ϕ_2＋1個時槽。將前ϕ_2個時槽安置與 $M_f(i)$相同的 overhead，最後一格空位放置 $M_f(i)$負載的第一個碼格。（如果將 $M_f(i)$負載中的碼格往前移後負載的長度為 0，即只剩 overhand，則丟掉 $M_f(i)$。）此時超級框架 i 會空出一個時槽，將超級框架 i 在 D_n^-之前（含 D_n^-）的碼格往前移一個時槽即可空出 D_n^-給 Cn，如圖 4.10。

Case 1.e：**若 $M_f(i)$,$M_l(i-1)$不是同一個連線，且超級框架(i-1)空時槽的個數 $n<\phi_2+1$ （$n\geqq 0$），則**

（i）先加入一個空時槽：先調整位置使超級框架（i-1）PCF 的最後 n 個時槽空出來。

Case 1.e.1: **若 $M_f(i-1)$,$M_l(i-2)$屬同一個連線且超級框架(i-2)空時槽的個數大於或等於 1，則將 B_1（i-2）至 CF-end 框架之前的碼格**

往前移一個時槽，並將 $M_f(i-1)$負載中的第一個碼格放於 $M_l(i-2)$負載中。（如果將 $M_f(i-1)$負載中的碼格往前移後負載的長度為 0，即只剩 overhand，則丟掉 $M_f(i-1)$。）此時超級框架(i-1)會空出一個時槽，將超級框架(i-1) B_n（i-1）之前（不含 B_n（i-1））碼格往前移一個時槽使空時槽集中在超級框架（i-1）的最後。

Case 1.e.2：若 $M_f(i-1)$,$M_l(i-2)$不是同一個連線，且超級框架(i-2)空時槽的個數大於或等於 $M_f(i-1)$的 overhead 長度加 1 即ϕ_3+1，則將〔$B_{\phi 3+1}(i-2)$,$B_{\phi 3}$ (i-2)〕間的碼格往前移一個時槽，〔$B_{\phi 3}$ (i-2),$B_{\phi 3-1}$ (i-2)〕間的碼格往前移 2 個時槽，依序前移後超級框架(i-2)在 CF-end 框架之前將會空出ϕ_3+1個時槽。將前ϕ_3個時槽安置與 $M_f(i-1)$相同的 overhead，最後一格空位放置 $M_f(i-1)$負載的第一個碼格。（如果將 $M_f(i-1)$ 負載中的碼格往前移後負載的長度為 0，即只剩 overhand，則丟掉 $M_f(i-1)$。）此時超級框架(i-1)會空出一個時槽，依序移位使空時槽集中在超級框架(i-1)的最後。

（ii）重複（i）直到ϕ_2+1-n個時槽皆安排完成。

（iii）若可以完成（ii），超級框架(i-1)將會空出ϕ_2+1個時槽。

其餘步驟與 Case 1.d 同。

(2) 根據定義可知，我們將在此超級框架所構成之時槽中已分配給連線 C_n的最後一個時槽稱為 L_n，因此在安排第 2 個時槽時將已安排成功的第一個時槽稱為 L_n，其排程方法與（1）同但須將（1）中的 D_n改成 L_n。

(3) 重複（2）直到ϕ_1+1個時槽皆安排完成。如果在重複（2）時有任何一次安排無法得到想要的空時槽個數則見 Case 3.b。

(4) 若在安排第一個時槽時就無法得到需求的空時槽個數則見 Case 3.a，若執行 Case 3.a 後仍無法完成 initial case 的規劃則不安排此 MPDU 給 C_n。

Case 2:如果要規劃的不是 C_n 此 MPDU 負載中的第一個碼格而是之後的碼格，將把此次要規劃的的碼格放在 L_n 處。考慮以下幾個案例：

Case 2.a：若在此超級框架〔1，L_n〕之間至少有一個空的時槽，想空出 L_n 的位置而不分割其他連線的碼格串，則將所有在 L_n 前最後一個空的時槽到 L_n 之間的碼格往前移一個時槽，空出 L_n 給 C_n 使用。

若無法滿足 Case2.a 則檢查超級框架(i-1) ,超級框架(i-2)空時槽的個數及 $M_f(i)$,$M_l(i-1)$,$M_f(i-1)$,$M_l(i-2)$各屬哪個連線，並考慮以下案例：

Case 2.b：若 $M_f(i)$,$M_l(i-1)$屬同一個連線且超級框架(i-1)空時槽的個數大於等於 1，則將 B_1（i-1）至 CF-end 框架之前的碼格往前移一個時槽，並將 $M_f(i)$ 負載中的第一個碼格放於 $M_l(i-1)$ 負載中。(如果將 $M_f(i)$ 負載中的碼格往前移後負載的長度為 0, 即只剩 overhand, 則丟掉$M_f(i)$。) 此時超級框架 i 會空出一個時槽，將超級框架 I 在 L_n 之前（含 L_n）的碼格往前移一個時槽即可空出 L_n 給 C_n。

Case 2.c：若 $M_f(i)$,$M_l(i-1)$屬同一個連線，而超級框架(i-1)空時槽的個數為 0，則考慮以下案例：

Case 2.c.1: 若 $M_f(i-1)$,$M_l(i-2)$屬同一個連線且超級框架(i-2)空時槽的個數大於或等於 1，則將 B_1（i-2）至 CF-end 框架之前的碼格往前移一個時槽，並將 $M_f(i-1)$負載中的第一個碼格放於 $M_l(i-2)$ 負載中（如果將 $M_f(i-1)$ 負載中的碼格往前移後 $M_f(i-1)$負載的長度為 0，即只剩 overhand，則丟掉 $M_f(i-1)$。)。此時超級框架(i-1)會空出一個時槽，其餘步驟見 Case2.b。

Case 2.c.2：若 $M_f(i-1)$,$M_l(i-2)$不是同一個連線，且超級框架(i-2)空時槽的個數大於或等於 $M_f(i-1)$的 overhead 長度加 1 即 ϕ_3+1，則將〔$B_{\phi 3+1}(i-2)$,$B_{\phi 3}$ (i-2)〕間的碼格往前移一個時槽，〔$B_{\phi 3}$ (i-2),$B_{\phi 3-1}$ (i-2)〕間的碼格往前移 2 個時槽，依序前移後超級

框架(i-2)在 CF-end 框架之前將會空出ϕ_3＋１個時槽。將前ϕ_3個時槽安置與M_f(i-1)相同的 overhead，最後一格空位放置M_f(i-1)負載內的第一個碼格。（如果將M_f(i-1)負載中的碼格往前移後負載的長度為 0，即只剩 overhand，則丟掉 M_f(i-1)。）此時超級框架(i-1)會空出一個時槽，其餘步驟參考 Case2.b。

Case 2.d：若M_f(i) ,M_l(i-1)不是同一個連線，且超級框架(i-1)空時槽的個數大於或等於M_f(i)的 overhead 長度加即ϕ_2＋1。則將〔$B_{\phi 2+1}$(i-1),$B_{\phi 2}$ (i-1)〕間的碼格往前移一個時槽，〔$B_{\phi 2}$ (i-1),$B_{\phi 2-1}$ (i-1)〕間的碼格往前移 2 個時槽，依序前移後超級框架(i-1)在 CF-end 框架之前將會空出ϕ_2＋１個時槽。將前ϕ_2個時槽安置與 M_f(i)相同的 overhead，最後一格空位放置M_f(i) 負載的第一個碼格。（如果將M_f(i)負載中的碼格往前移後負載的長度為 0，即只剩 overhand，則丟掉 M_f(i)。）此時超級框架 i 會空出一個時槽，將超級框架 I 在 L_n之前（含 L_n）的碼格往前移一個時槽即可空出 L_n給 Cn。

Case 2.e：若M_f(i) ,M_l(i-1)不是同一個連線，且超級框架(i-1)空時槽的個數 n≦ϕ_2＋１（n≧0），

（i）先加入一個空時槽：先調整位置使超級框架（i-1）PCF 的最後 n 個時槽空出來。

Case 2.e.1: 若M_f(i-1) ,M_l(i-2)屬同一個連線且超級框架(i-2)空時槽的個數大於或等於 1，則將 B_1（i-2）至 CF-end 框架之前的碼格往前移一個時槽，並將M_f(i-1) 負載中的第一個碼格放於M_l(i-2)負載中。（如果將 M_f(i-1) 負載中的碼格往前移後負載的長度為 0，即只剩 overhand，則丟掉 M_f(i-1)。）此時超級框架(i-1)會空出一個時槽，將超級框架(i-1) 在 B_n(i-1)之前（不含 B_n(i-1)）碼格往前移一個時槽使空時槽集中在超級框架（i-1）的最後。

Case 2.e.2：若M_f(i-1) ,M_l(i-2)不是同一個連線，且超級框架(i-2)空

時槽的個數大於或等於 $M_f(i-1)$的 overhead 長度加 1 即ϕ_3+1，則將〔$B_{\phi 3+1}(i-2), B_{\phi 3}(i-2)$〕間的碼格往前移一個時槽，〔$B_{\phi 3}(i-2), B_{\phi 3-1}(i-2)$〕間的碼格往前移 2 個時槽，依序前移後超級框架(i-2)在 CF-end 框架之前將會空出ϕ_3+1個時槽。將前ϕ_3個時槽安置與$M_f(i-1)$相同的 overhead，最後一格空位放置$M_f(i-1)$負載的第一個碼格。（如果將 $M_f(i-1)$ 負載中的碼格往前移後負載的長度為 0，即只剩 overhand，則丟掉 $M_f(i-1)$。）此時超級框架(i-1)會空出一個時槽，依序移位使空時槽集中在超級框架(i-1)的最後。

（ii）重複（i）直到ϕ_2+1-n個時槽皆安排完成。

（iii）若可以完成（ii），超級框架(i-1)將會空出ϕ_2+1個時槽。其餘步驟見 Case 2.d。

若在安排此碼格時無法得到所需之空時槽個數則：

(1)如果新的碼格之期限在超級框架 i 之後（不含超級框架 i），則將此碼格當成是 C_n另一個 MPDU 負載中的第一個碼格並根據 Case 1 做規劃。

(2)若新的碼格其期限在超級框架 i 內，則考慮 Case 3.b。若仍無法完成 Case 2 的規劃則不安排此碼格給 C_n。

Case3

Case 3.a：首先觀察超級框架 i 在 D_n^-之前（含 D_n^-）排程過的 MPDU 中是否有碼格其期限在超級框架 i 之後（不含超級框架 i），可以取出當成是 C_m 的另一個 MPDU 負載中的第一個碼格。取出一個這樣的碼格，其餘碼格往左移動一個時槽，使得 C_n可以獲得 D_n^-的位置。倘若無法取得，則先考慮本身的 CLR 是否為 1，若是，則將此碼格丟掉；若不是則觀察 D_n^-之前是否有碼格其 CLR 為 1，若可取出 1 個 CLR 為 1 的碼格，則將分配給 C_n的第一個時槽放在 D_n^-，若無法取出夠量的 CLR 為 1 之碼格，

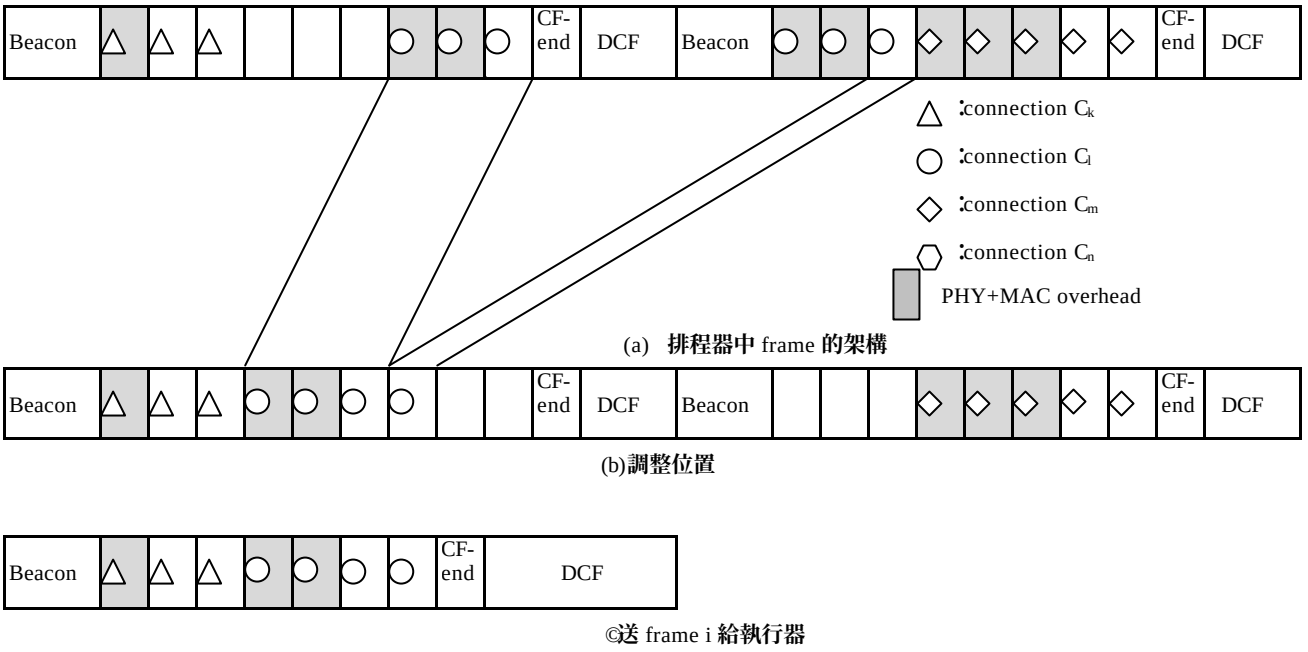


圖 4.11　超級框架 i 的第二步驟過程圖

則不規劃此 MPDU 給 C_n。

Case 3.b：首先觀察超級框架 i 在 F_n 之前（不含 F_n）排程過的 MPDU 中否有碼格其期限在超級框架 i 之後（不含超級框架 i），可以取出當成是 C_m 的另一個 MPDU 負載中的第一個碼格。取出一個這樣的碼格，其餘碼格往左移動一個時槽，使得 C_n 可以獲得 L_n。倘若無法取得，則先考慮本身的 CLR 是否為 1，若是，則將此碼格丟掉；若不是，則觀察 F_n 之前是否有碼格其 CLR 為 1，若可取出 1 個 CLR 為 1 的碼格，則將分配給 C_n 的第一個時槽放在 L_n，若無法取出夠量的 CLR 為 1 之碼格，則不規劃 C_n 此碼格。

第二個步驟：完成下一個超級框架的規劃。

當排程器將規劃中最前面的超級框架提出時所必須做的調整動作。設將提出的超級框架為超級框架 i，先檢查 $M_f(i+1)$,$M_l(i)$是否為同一組連線及 $M_f(i+1)$ 負

表 4.1　純 ATM PCF 週期分析所使用之參數值

Parameter	Value
Channel capacity	23Mbps
Slot duration	17.24μs
SIFS	20μs
DIFS	60μs
superframe length	2ms
CFPMaxDuration (Pure ATM)	1.569ms
CFPMaxDuration (IEEE 802.11)	0.93ms
MPDU payload (IEEE 802.11 PCF)	10cells
Mean arrival rate	500kbps
rt-VBR wireless hop CDT	17.2ms
nrt-VBR wireless hop CDT	517ms
overheadψ	4 slots

載中有多少碼格。接著將超級框架 i 中的碼格往前移補滿空位並觀察剩餘的空時槽個數。若 $M_f(i+1)$,$M_l(i)$為同一組連線且空時槽的個數比 $M_f(i+1)$負載中的碼格多，則將 $M_f(i+1)$負載中的碼格往前移至超級框架 i 空時槽處，此時可將 $M_f(i+1)$的 overhead 丟掉。最後將 CF-end 框架往前移，放在這些碼格之後，而此超級框架剩餘的部分即為 DCF，見圖 4.11。

為了瞭解此系統的效能，以下利用 BONeS 模擬系統架構。模擬所使用的系統參數如表 4.1〔9〕,〔26〕，在此假設所有的使用者皆傳送 VBR 的交通型態。因為此協定是以期限為排程的依據因此時間延遲是被保證的，而 CDVT（cell delay variation tolerance）可在 AP 端或之後的網路端做調節，如〔27〕所提之方法，故我們將重點放在碼格漏失率（loss rate）上。首先考慮的是此系統與 IEEE 802.11 PCF 週期之效能比較。假設兩系統的使用者皆為及時性 VBR 用戶，而 IEEE 802.11 呼叫使用者的順序是以用戶的序號為依據，且每次用戶送出的 MPDU 負載為 10 個碼格。圖 4.12 是比較兩系統的漏失率，在此漏失的定義是送達 AP 端的碼格超過其期限即為漏失。由圖 4.12 可發現，IEEE 802.11 隨著用戶人數增加其碼格漏失率比純 ATM 系統的漏失率有較明顯的上升趨勢，這是因為 IEEE 802.11 的系統無法讓較緊急的用戶先傳送的緣故。另外，圖 4.13 是兩系統的貫通率比較，在此貫通率指的是在 CFPMaxDuration 中成功傳送資料框的時間與

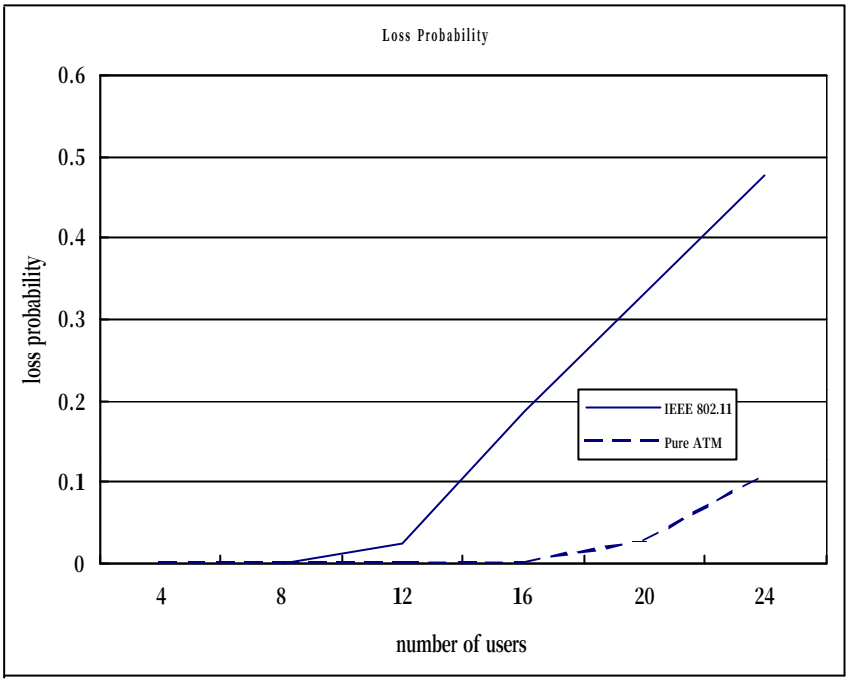


圖 4.12 IEEE 802.11 與純 ATM 系統間漏失率之比較

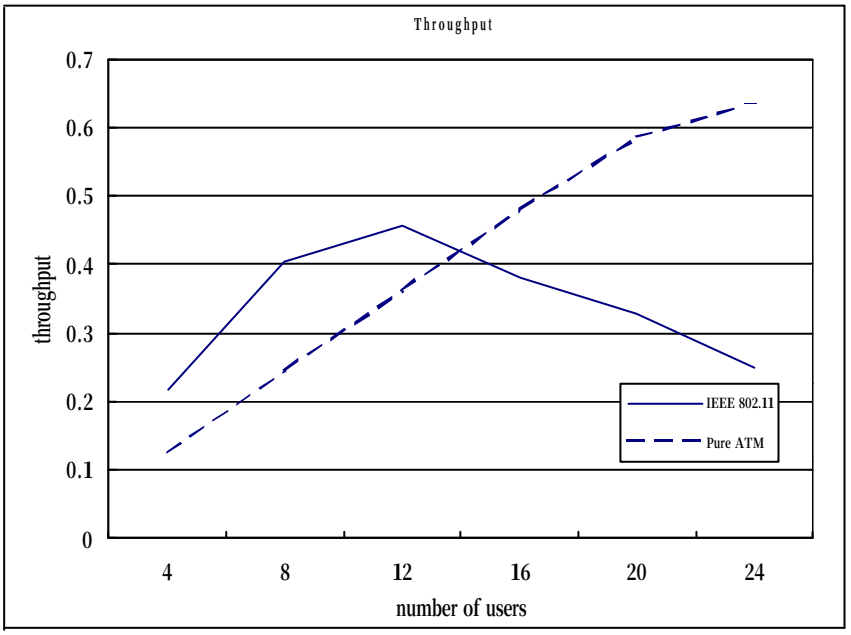


圖 4.13 IEEE 802.11 與純 ATM 系統間貫通率之比較

CFPMaxDuration 的時間之比值。在用戶人數少時，兩系統的使用者應該皆可成功傳送資料框，此時若 CFPMaxDuration 的值較小則貫通率會較高，因為大部分的時間皆在傳送資料，因此 IEEE 802.11 在用戶人數少時貫通率較高。但隨著人

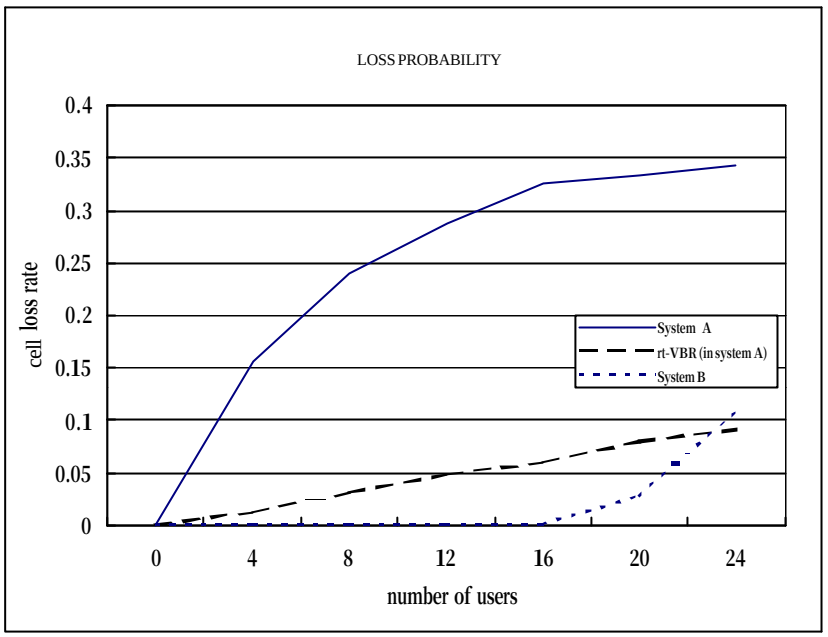


圖 4.14 系統 A 與系統 B 碼格漏失率之比較

數上升，IEEE 802.11 系統會因不管此資料框是否已超過期限仍舊傳送，造成其他有用的資料框被迫延遲傳送時間，使得當此資料框可傳送時又超過其期限的機率大增，因此漏失情況嚴重。而每次 PCF 週期雖然皆有資料框傳送但卻因傳送的大都為無效的資料框，因此貫通率不斷下降。而純 ATM 系統卻能保證傳送出的碼格皆為有用的碼格並提供良好的多工增益及有效率的資源分配，因此能緩和漏失的情形，且貫通率也能在一定程度之上，故我們可以證明此系統的確可以改善系統效能。接著，我們將探討非及時性 VBR 對及時性 VBR 的影響：當系統中同時有及時性 VBR 和非及時性 VBR 存在時（各佔一半），稱此系統為系統 A，此時系統的碼格漏失率會遠大於系統中皆為及時性 VBR（稱此系統為系統 B）的情況，如圖 4.14。如果只考慮系統 A 中的及時性 VBR 碼格漏失率與系統 B 比較時，還是有較大的漏失率，這是因為當執行器在執行超級框架 i 時，會陸續收到用戶傳送之 piggyback，根據這些資訊，排程器繼續安排碼格的放置位置。如果此時收到非及時性 VBR 的 piggyback，排程器會將這些碼格放置在其連線的期限之前，假設是在超級框架（i＋n），n 的值可能為 1 、200。當排程器經過一段時間後準備將及時性碼格安排到超級框架（ i＋n）時會發現時槽已被非及時性 VBR 佔據，

導致及時性 VBR 因沒有足夠的時槽可容納它們而漏失，此時這種暫時性的優先權喪失情形將影響及時性交通的效能。此外，由圖 4.14 可看出，雖然系統 A 及時性交通的漏失率上升，但整個系統的漏失率更大，這表示非及時性 VBR 的漏失情形比及時性交通的漏失更大。這是因為此系統的超級框架長度固定，所以在將碼格往前移時會因 $M_t(i-1)$ 、$M_f(i)$連線的相同與否而有不同的空時槽需求量，如果前一個超級框架的空時槽數不足會再往前一個超級框架尋找空時槽，此時要求的空時槽個數可能增加。舉例來說，如果連線 C_n 在 L_n 之前沒有空時槽可用，而 $M_t(i-1)$ 、$M_f(i)$ 屬於不同連線，此時如果超級框架（i-1）有（ψ_2+1）個空時槽則經過移位後可讓 C_n 的碼格放在 L_n 處。但如果超級框架（i-1）沒有空時槽可用，而 $M_t(i-2)$ 、$M_f(I-1)$屬於不同連線，則超級框架（i-2）至少要有（$\psi_2+\psi_3+1$）個空時槽才可讓 C_n 的碼格放在 L_n 處。而後每影響一個超級框架，所需的時槽個數就可能不斷增加。因此為了減少系統複雜度，在此架構下設定碼格是否能放置在 PCF 週期內須視此從碼格預定放置的時槽（設位於超級框架 i）之前到超級框架（i-2）之間是否有夠量的空時槽可供使用，若無法得到所需之空時槽個數則將此碼格丟掉。由於非及時性 VBR 的期限很大，而 MPDU 是在靠近期限時才送出，所以兩 MPDU 間的時間差可能也很大造成每次 piggyback 的量會很可觀，但是因為每個連線受限只能使用三個超級框架內的空時槽，所以非及時性 VBR 的漏失率將會很嚴重。因此，為了解決以上問題，我們必須考慮如何讓（1）非及時性 VBR 每次 piggyback 的碼格數減少，（2）非及時性的 VBR 交通減少對及時性交通的影響。

關於第一個問題，解決碼格量過多的方法就是將這些碼格分段送出，如此兩 MPDU 的間距縮短，則每次 piggyback 的量就會相對減少。或者另一個較簡單的方式是讓這些連線的期限縮短，如此可以不改協定架構而縮小兩 MPDU 的間距。圖 4.15 比較原期限值與縮短的期限值在系統碼格漏失率上的差異，其中 d2 代表非及時性 VBR 的期限。由圖可知，當期限縮短時系統的漏失率下降，表示

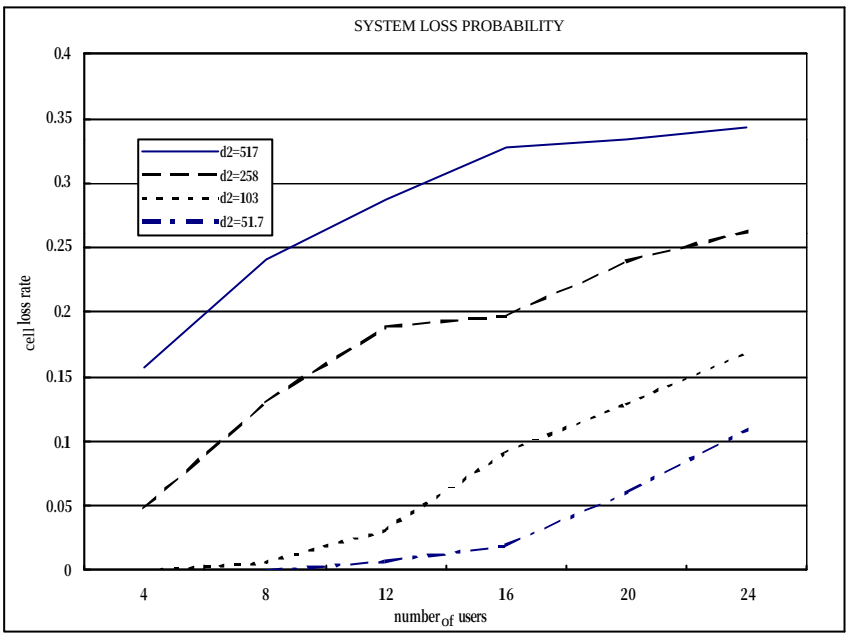


圖 4.15　系統碼格漏失率在各種期限下的比較

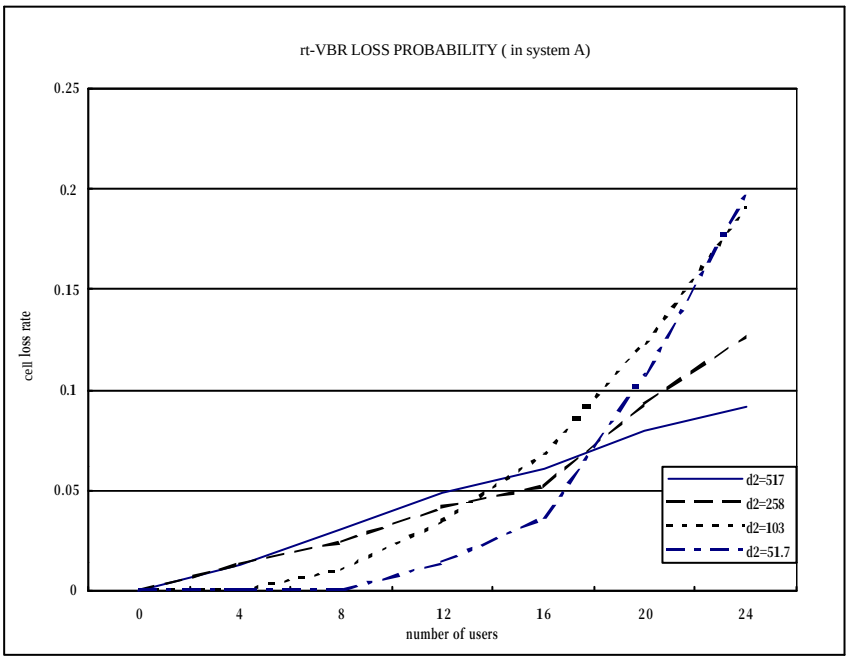


圖 4.16　及時性 VBR 在不同非及時性 VBR 期限下的碼格漏失率

非及時性 VBR 的碼格漏失量減少。然而，另一個需考量的問題是及時性的交通是否會受其影響，因此圖 4.16 比較這些非及時性 VBR 將期限縮短成不同值時，

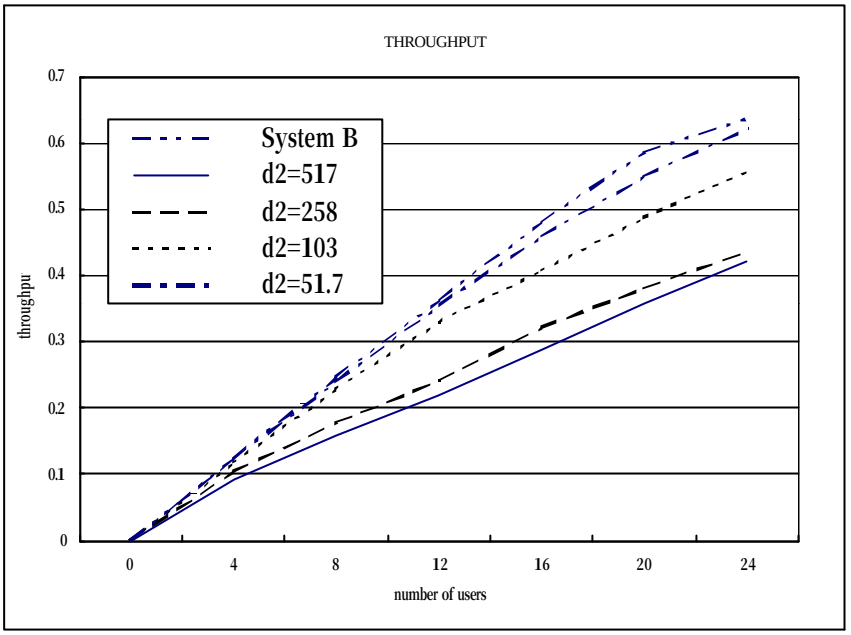


圖 4.17 在各種期限下的系統貫通率

對及時性交通碼格漏失率的影響。雖然將非及時性 VBR 期限縮短會影響到及時性交通的效能，但事實上影響的程度並不及非及時性 VBR 交通改善的程度，因此可以設定一臨界值（例如 100ms），當非及時性 VBR 的期限大於此臨界值時就將其期限縮短到此臨界值。圖 4.17 為系統貫通率在不同期限下的情形，可以看出當非及時性 VBR 交通期限壓縮後系統貫通率上升，這是因為碼格漏失機率減少的緣故。

雖然如此，我們仍希望將對及時性交通的影響降到最低，因此考慮藉由減少在 PCF 週期內傳送的非及時性 VBR 連線數目來達到這個目標。減少在 PCF 週期內非及時性連線個數的方式是將非及時性 VBR 為兩類，一類可放入 PCF 區間並在期限大於臨界值時縮短期限值；另一類則在 DCF 區間傳送。若非及時性 VBR 從來源端（source）到目的端（destination）的期限要求在 2 秒內則可放在 PCF 區間，否則需在 DCF 週期傳送。選擇 2 秒的原因是因為在 ATM 標準內，及時性交通最大可容忍的延遲時間為 2 秒（來源端到目的端）。舉例來說，若在網路內有 12 個及時性 VBR 用戶，其期限為 17.2ms，6 個非及時性 VBR 其期限為 517 ms

表 4.2 舉例說明非及時性 VBR 分類對系統的影響

Case	Loss probability	Real time loss rate	Throughput
(12,6,6,)	0.35095	0.28278	0.28278
(12,6,d2=517)	0.18289	0.05958	0.38869
(12,6,d2=103)	0.112168	0.12393	0.46344
(12,6,d2=51.7)	0.035744	0.05172	0.514

及 6 個非及時性 VBR 其期限為 1.03s，其餘參數與表 4.1 同。當這三種交通型態的用戶皆在 PCF 週期內傳送時，從表 4.2 可知系統的漏失率大約為 0.35，而及時性交通的漏失率為 0.28。若將期限為 1.03s 的 6 個非及時性 VBR 移往 DCF 週期，則系統的漏失率可下降到 0.18，而及時性 VBR 的漏失率更可下降到 0.06。由此可知，將非及時性 VBR 移往 DCF 區間的確可減少及時性交通的漏失情形。另外從表 4.2 也可看出，將非及時性 VBR 的期限縮短可使系統漏失率由 0.18 降到 0.035，大大降低了非及時性 VBR 的損失情形。由於部分的非及時性 VBR 將會在 DCF 週期與 ABR 或 UBR 交通競爭通道的使用權，為了提升非及時性 VBR 的優先權，考慮以下三種方式：(1) ABR 或 UBR 用戶在通道閒置 DIFS 後必須再等待 3 個時槽的時間才能傳送資料，而非及時性 VBR 用戶在 DIFS 後即可傳送。(2) 將 ABR 或 UBR 的最小競爭視窗增加到 64，而非及時性 VBR 的最小競爭視窗為 32。(3) 同時加入 (1)、(2) 兩種情形。

圖 4.18 比較四種不同情況下的非及時性 VBR 平均碰撞次數，這些數據是依據〔22〕的數學分析而來。系統是在固定 ABR 用戶個數為 20 的假設下，改變非及時性 VBR 的用戶個數以觀察碰撞的情形。圖中 (a,b,c) 分別表示 (非及時性 VBR 最小的競爭視窗，ABR 最小之競爭視窗，ABR 在通道閒置 DIFS 後額外的等待時間)，其中 (32,32,0) 為一般沒有優先權的情形。由圖 4.18 及圖 4.19 可以發現 (32,64,0) 的案例不論是在非及時性 VBR 或 ABR 交通下，其平均碰撞次數皆比 (32,32,0) 小，而 (32,32,3) 案例在非及時性 VBR 的交通型態下雖然仍

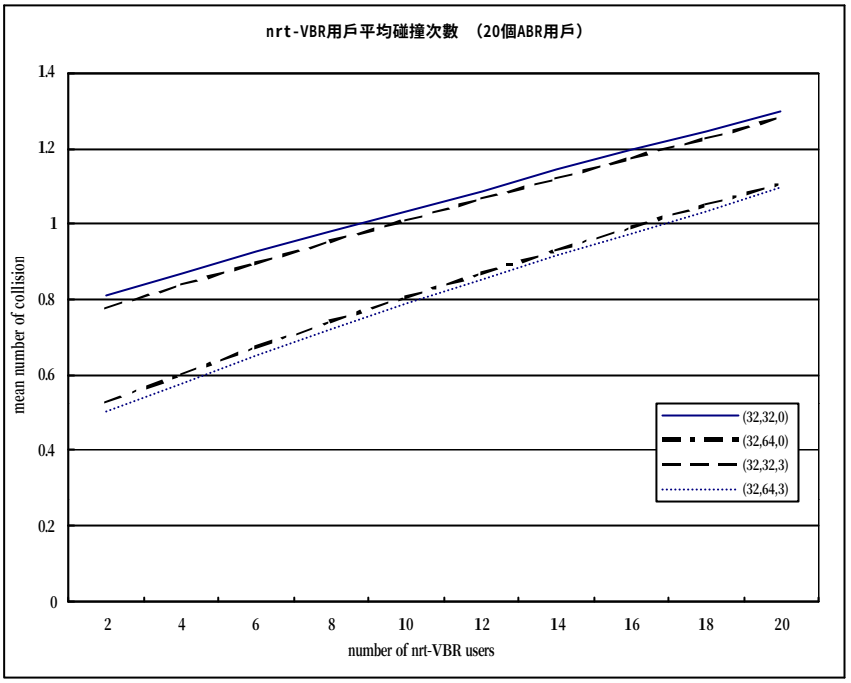


圖 4.18　**非及時性** VBR **平均碰撞次數**

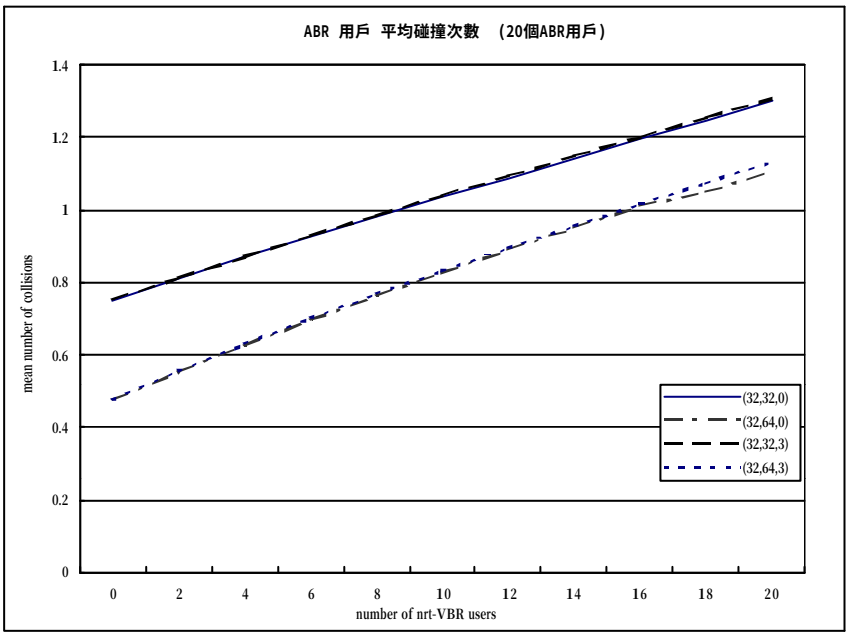


圖 4.19　ABR **平均碰撞次數**

優於（32,32,0），但程度上就比不上（32,64,0）。另外由於（32,64,3）的效能與（32,64,0）相似，再加上圖 4.20 顯示出（32,64,0）的系統貫通率與（32,32,0）相似，因此最後決定採用（32,64,0）的案例達到非及時性 VBR 優先權的要求。

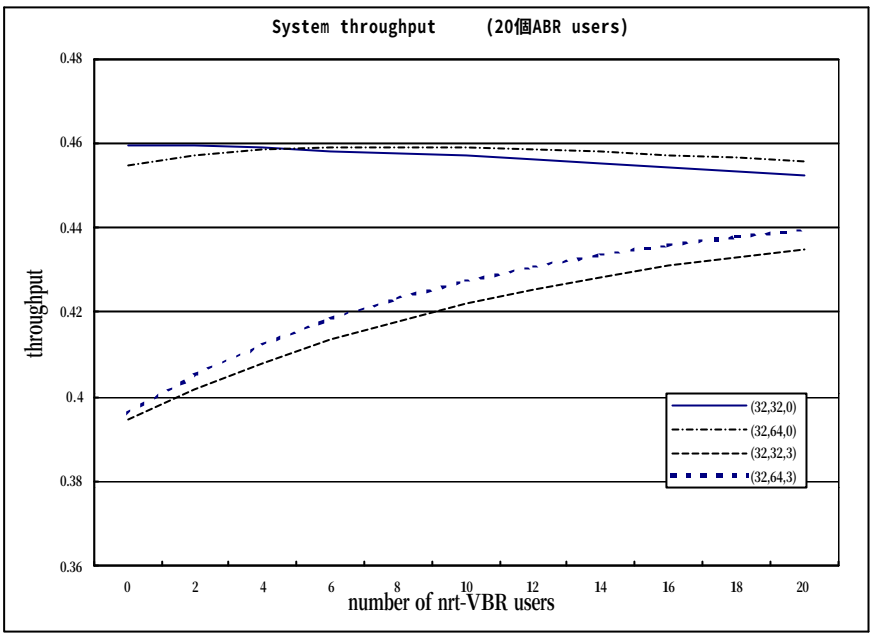


圖 4.20　四種優先權方案下的系統貫通率