第二章

相關研究

2.1 Bandwidth Allocation with Fairness

在[3]中，提出二層式的 Bandwidth Allocation 的概念，如圖 2.1 所示：

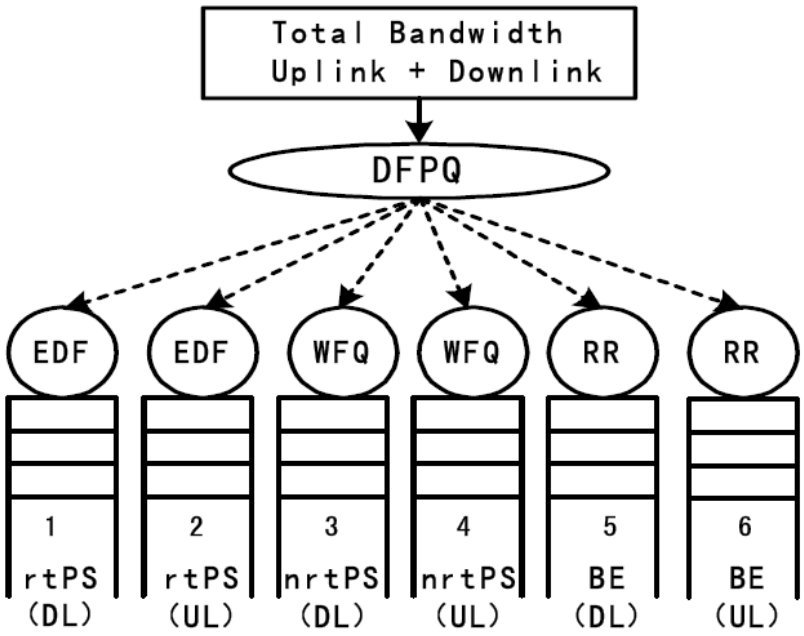

圖 2.1 Hierarchical structure of bandwidth allocation [3]

在第一層 DFPQ scheduling 前需先對連線種類及方向做優先權分類，其區分規則爲 rtPS > nrtPS > BE，Downlink > Uplink。因而得出表 2.1 的順序，數字愈小代表其優先權愈高。

1	2	3	4	5	6
DL-rtPS	UL-rtPS	DL-nrtPS	UL-nrtPS	DL-BE	UL-BE

表 2.1 Priority of each service class [3]

DFPQ 的運作方式如圖 2.2 所示，其主要關鍵爲 Quantum size，利用負預算(deficit)的概念來停止 service class 運作，當某 service class 其 Quantum size 使用至負數時停止其運作，切換到下一個 service class 繼續運作，其 Quantum size 表示式如下所示

$$Quantum\ [i] = \sum_{j=0}^{Ji} r_{\max}(i, j) \qquad (b)$$

其中$r_{max}(i,j)$爲(the Maximum Sustained traffic rate of the *j*th connection in the *i*th class of service flow)，由上式(b)可知，每一種不同優先權的service class，它所擁有的Quantum size是由相同優先權中各別連線的Maximum Sustained Traffic Rate加總起來的值。因此表 2.1 中六個不同優先權的service class會有六個不同的Quantum Size。詳細的運作過程以圖 2.2 爲例，其中L_{total}表示系統總頻寬，L_a表示目前系統可用頻寬。假設系統目前只有二種連線rtPS與BE，其中rtPS_Queue中的所有連線其Maximum Sustained Traffic Rate加總起來的值爲1000，因此Quantum[rtPS]=1000，同時假設Quantum[BE]=500。在每一回合中先從高優先的連線開始，在這個例子則先從rtPS開始，從圖 2.2 中的 1-1 回合我們知道，rtPS連線的流量在queue中是以burst形式存在，所以一次必須以一個burst爲單位送出，因此 1000 的quota可以滿足 600+300+400=1300，此時Quantum[rtPS]變成-300，結束 1-1 回合。在滿足的過程中必須隨時注意L_a是否小於 0，因爲L_a代表系統所剩的Bandwidth。在此例中L_{total}=2500，當 1-1 回合過後$L_a=L_{total}$-1300=2500-1300=1200，當rtPS的Quantum Size用完之後則換下一個Service Class。在此例爲BE，因此同 1-1 回合的作法，在 1-2 回合中由於Quantum[BE]=500，因此只能滿足 400+300=700，此時Quantum[BE]變成-200，1-2 回合結束。此時L_a=1200-700=500，由於L_a還有剩餘，因此進行下一回合。下一回合開始所有Queue裡面還有資料的Service Class再加一次Quantum[i]，所以此時Quantum[rtPS]=1000-300=700，而Quantum[BE]=500-200=300，再繼續 2-1、2-2 回合，以此類推，直到L_a用完或所有的queue都沒有任何資料時就停止。

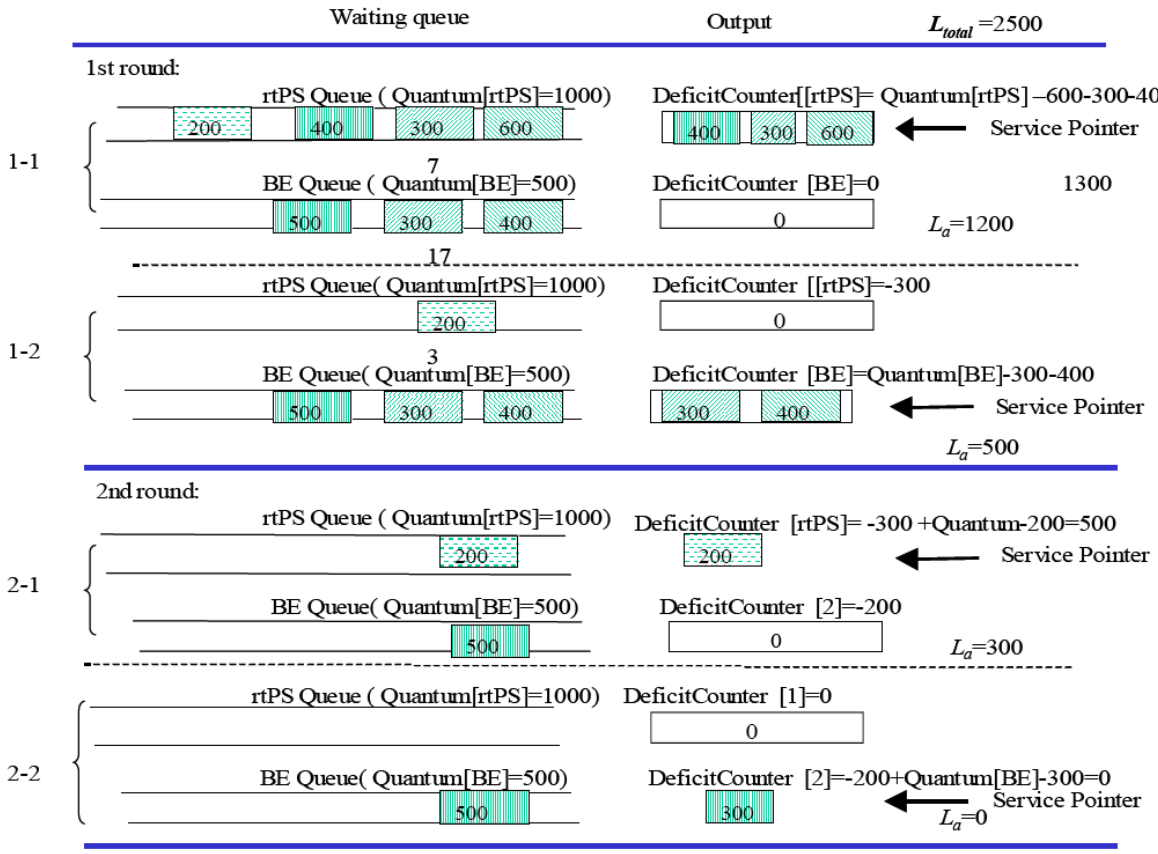

圖 2.2 An example of DFPQ scheduling [3]

在第二層的 scheduling 中，最主要是針對各別 service class 的 queue 做設計，因此會發現對於不同用途的連線，會有不同的 queue management 方法，我們會發現大致使用的方法如下：

- RTPS 使用 EDF (Early Deadline First)
- NRTPS 使用 WFQ (Weight Fair Queue)
- BE 使用 RR (Round Robin)

EDF：

將最先到期的封包排在最前面，對於有時間限制的 rtPS 連線，這是必要的，因爲如此可保證最快到期的封包可以最優先送出，以避免送出過期的(無效的)封包。

WFQ：

用在 nrtPS，雖然 nrtPS 連線不像 rtPS 有嚴格的延遲時間限制，但有最小流量的要求，因

此使用 WFQ 可反應出各連線不同最小流量要求的差別。

RR：

用於 BE 連線，因爲 BE 沒有任何服務品質要求，因此使用 RR 的方式即可。

最後定出 Fairness 公式

$$FAIR_{rb} = \left| \frac{Th_{rtps}}{S_{rtps}} - \frac{Th_{be}}{S_{be}} \right|$$

- Th_{rtps}：爲rtPS的throughput
- Th_{be}：爲BE的throughput
- S_{rtps}：爲rtPS的total traffic source
- S_{be}：爲BE的total traffic source

得出圖 2.3 的結果

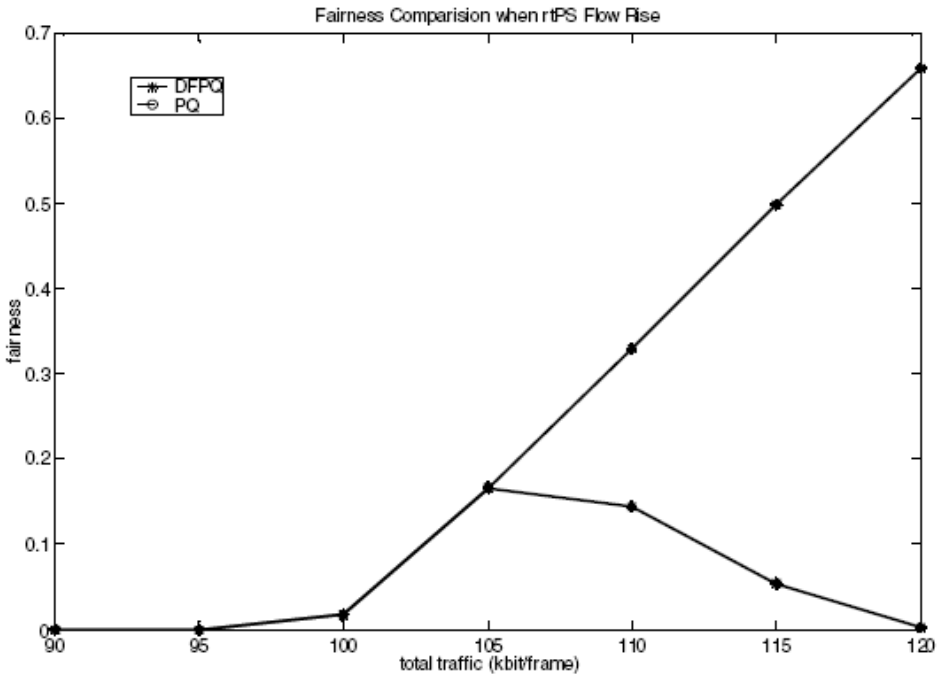

圖 2.3 Fairness comparison [3]

Chen[3]的作法有一個缺點，使用 Quantum 的方式，有可能在追求公平性上導致 rtPS 在

QoS 品質產生不滿足的情況。

Wang[4]提出以 DFPQ 爲核心技術並加以改進，再與 802.16-2005 使用 WFQ/PQ 的方法以模擬數據做比較，其主要概念如圖 2.4 所示

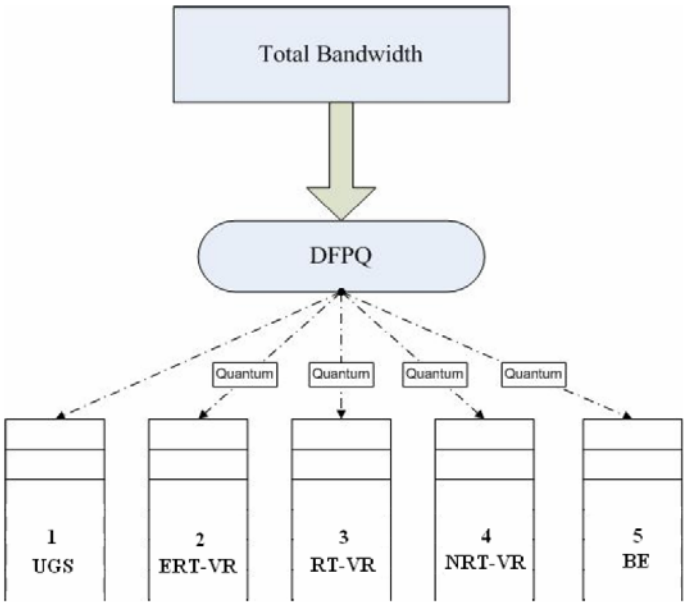

圖 2.4 DFPQ 使用於 802.16-2005 模型 [4]

如同[3]所示之 DFPQ，[4]的做法多了一個考量。首先確知因爲每類訊務皆有一定的公平性可分配到系統頻寬，不至於在佇列中發生飢餓的情況；但也因爲此公平性，會造成高優先權訊務的延遲時間會因此升高，因此將做法調整成下列五個步驟：

(a)若 ertPS buffer 中有資料等待傳送，即傳送 ertPS 資料；反之執行步驟 2。

(b)更新具資料欲傳送的 flow list 及每類訊務之參數(Quantum)。

(c)依每類訊務可使用的 Quantum，依序由高優先權往低優先權從佇列中提出資料傳送。

(d)當 BE 佇列中有資料等待傳送，判斷於 RT-VR 中最應優先傳送資料其最大延遲時間是否可容忍插入 BE 訊務傳送 (不同於原本 DFPQ 設計的改良處)。

(e)當佇列中資料爲空或 Quantum 使用完畢或此 frame 已安排完畢，則重新執行此流程安排下一個 TDD frame 的排程順序。

Chan**[5]**提出一個不同的架構，其主要架構如圖 2.5 所示

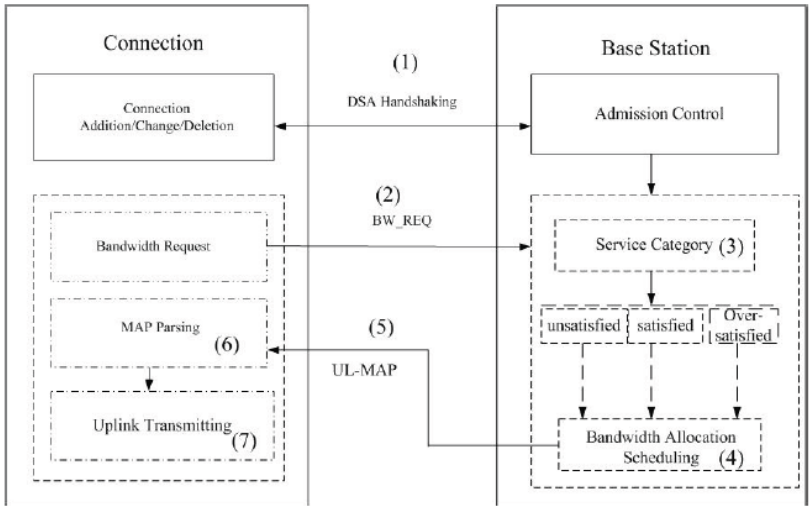

圖 2.5 Proposed structure [5]

當所有連線的 BR 進來後，會先將之分成三類，分別爲 unsatisfied、satisfied、over-satisfied。其最主要演算法如下，其中 PF 爲 Priority Function 用來計算各連線的 priority function 值。

$$PF_i = \begin{cases} \dfrac{R^i_{allocated}}{R^i_{min}}, & \text{if } R^i_{allocated} < R^i_{min} \\ \dfrac{R^i_{allocated} - R^i_{min}}{R^i_{max} - R^i_{min}}, & \text{if } R^i_{min} \le R^i_{allocated} \le R^i_{max} \\ \dfrac{R^i_{allocated} - R^i_{max}}{QoS\_Fac_i \times R^i_{allocated}}, & \text{if } R^i_{allocated} > R^i_{max} \end{cases}$$

$$QoS\_Fac_i = \begin{cases} 3, & \text{if } i \in rtPS \\ 2, & \text{if } i \in nrtPS \\ 1, & \text{if } i \in BE \end{cases}$$

首先值得注意的是 $R^i_{allocated}$ 的定義爲連線 i 前一次所分配的 allocation BW rate。

採用的流程方式同樣爲二層式的架構，第一層架構會將所有連線根據其 $R^i_{allocated}$ 與 $R^i_{\min}$ 以及 $R^i_{\max}$ 之間的關系將所有連線分配到三種不同的分類中(unsatisfied、satisfied、

over-satisfied) 。

第二層架構，則從 unsatisfied 類別開始，利用 unsatisfied 的公式計算出此類別中所有連線的 PF (Priority Function)值，擁有最小 PF 值的連線可以分配到等同其 BR 要求的資源，以此類推直到同一類別中所有連線都被滿足，再推進到下一個類別。當分配時如果資源已不足，就停止該回合。這樣的頻寬配置方式最值得注意的是由於 BE 的 $R^i_{\min}$ =0，因此在 unsatisfied 類別中不會有 BE 連線的存在，因此只要上次分配頻寬小於 $R^i_{\min}$ 的 nrtPS、rtPS 則在這次會有很大的機會獲得其 BR 相對應的 BW，再看 satisfied 類別會發現只要上次沒分配到頻寬的 BE 其計算出來的 PF 將會等於 0，因此在 satisfied 類別中，上階段沒獲得頻寬的 BE 被滿足的機會最大，平均每二次 BR 就有一次會被滿足。如此似乎很容易造成 rtPS 在 QoS 的要求上不滿足，這是很值得探討的。

表 2.2 是模擬的環境，其 throughput 如圖 2.6 所示，由於是純寫程式去模擬，因此可以發現執行的結果十分一致。

Type	$R^i_{\min}$ (*Kbps*)	$R^i_{\max}$ (*Kbps*)	Max latency (ms)	Number of flows
UGS	60	60	20	5
rtPS	600	650	50	3
rtPS	500	700	50	2
rtPS	550	800	50	2
nrtPS	450	550	100	3
nrtPS	300	500	100	2
nrtPS	200	430	100	2
BE	0	200	200	3
BE	0	180	200	2
BE	0	300	200	2

表 2.2 Simulation parameters [5]

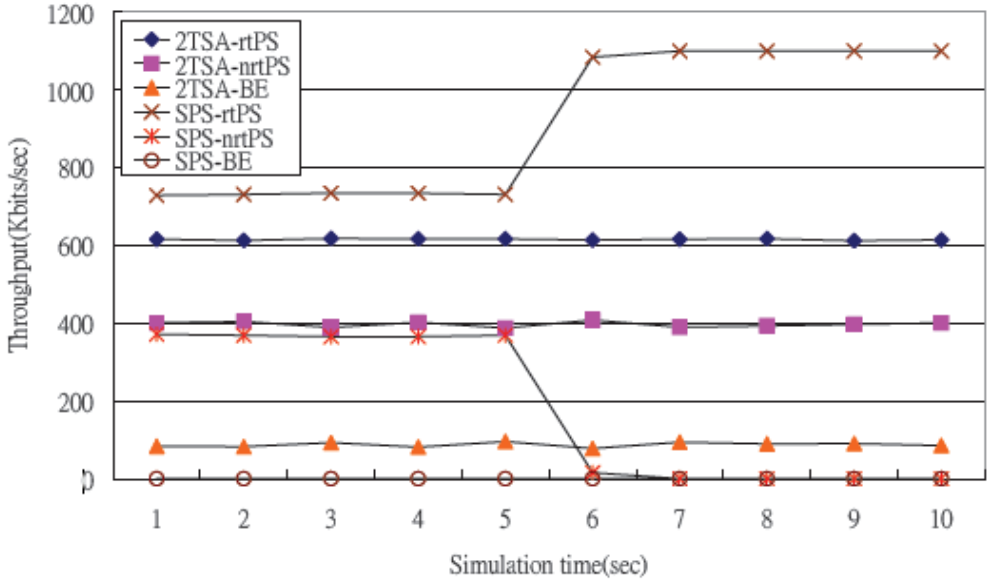

圖 2.6 Average throughput [5]

在 Fairness Degree (FD)上的定義如下式所示

$$FD = \frac{\left[\sum_{i=1}^{n} SD(i)\right]^2}{n\sum_{i=1}^{n}\left[SD(i)\right]^2}$$

在此，n 代表連線數，而 SD (Share Degree)則定義成

$$SD(i) = \frac{R^i_{allocated} - R^i_{\min}}{R^i_{\max} - R^i_{\min}}$$

FD 計算出來的值代表各連線 SD 之間的變異數，愈小的變異數會得出較大的 FD 值，FD 會介於 0~1 之間，愈大的 FD 值代表愈公平。圖 2.7 爲模擬環境下所得出的 Fairness degree。

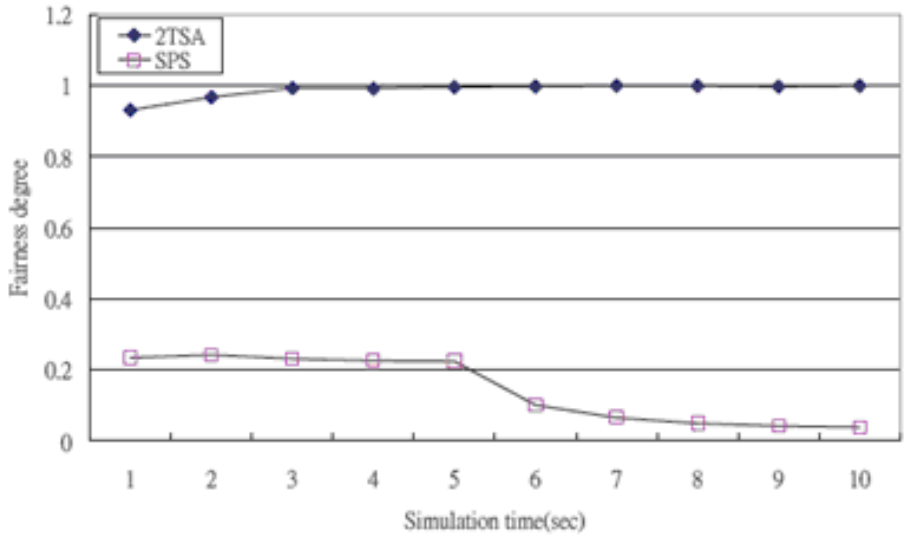

圖 2.7 Fairness degree [5]

2.2 Bandwidth Request

Ni[6]中調查了 uni-polling 及 contention-polling 傳送 BR 封包在 WiMAX 不同系統負載下的效率如圖 2.8。使用的參數如下 n (= 30) stations numbers、K (= 5) reserved BR contention slots、BEB parameters (L,m)=(1,4)、 BEB=binary exponential backoff (Wmin=L*K、m=retry limit)，其中 Wmin 爲最小的 contention window size。

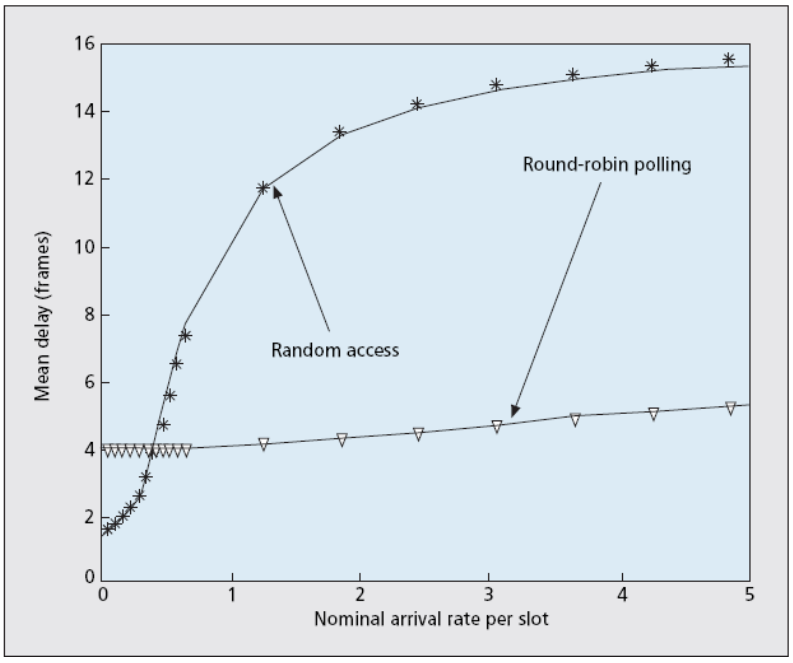

圖 2.8 Performance comparison:random access vs. polling [6]

圖 2.8 顯示，在 Request rate 較低時應使用 contention 較好，然而在 Request rate 較高時應使用 uni-polling 較好，因此最好設計一個可以動態切換的機制較能取得較好的 system performance。

Lin [7]顯示，reserved BR slot 的大小跟系統的產能(throughput)有關，如果設太大則可傳送的資料會太小，若設太小，grant delay time 又會增加很多，造成頻寬使用度不佳。Reserved BR slot 的設法大致上可根據二個方法，方法一是根據 active SSs 來決定 reserved BR slot 的大小，方法二是根據剩餘的 bandwidth。

方法一的缺點如下：

(a) 沒有考慮傳輸頻寬。

(b) 即使有夠多的 BR 可以成功送出，系統可能也沒有這麼多的頻寬可供使用，因此某些成功的 BR 將會被 drop 掉。

方法二，則先把剩餘可傳輸的頻寬計算出來，再看可容納幾個成功的 BR，假設可容納 K 個成功 BR 所需的頻寬，則規劃出 3K 個 reserved BR slot。

但[7]會有二個問題，首先是剩餘可傳輸的頻寬並非容易計算，此外每次成功的 BR 封包都配置相同的頻寬可能不符合實際的需求。

2.3 Scheduling

Huang[8]指出，使用相對性公平的方法(proportional fair)，可以獲得較佳的頻寬使用度，不會因爲 non-real-time 的連線變多，就導致 real-time 連線的品質被犧牲，若再搭配 EDF (Early Deadline First)將可得到很不錯的效能。**Wongthavarawat[9]**及**Liu[12]**指出，不同等級的連線最好使用不同的佇列管理方式，例如：rtPS 最好使用 EDF(Early Deadline First)，nrtPS 則可使用 WPQ(Weight Priority Queue)，BE 則使用 RR(Round Robin)即可。**Sayenko[10]**指出，scheudling 可細分成三步驟，首先先滿足 QoS 最少需求的連線，之後再比例分配剩下的頻寬以免低優先權連線發生 starvation，最後再重排 slot 順序以求得較佳的 QoS 品質。**Xergias[11]**指出使用 Frame Registry Tree 來決定當下最好的配置，尤其配置頻寬可跨 frame，這樣可以使頻寬使用度更爲有效。