第三章

研究方法

3.1 問題描述

從上述流程可以觀察到，節點在進行資料轉送時，媒體存取時間主要分爲兩個期間：第一是三方交握的期間；第二是完成交握到資料確實送出的等待期間，亦即三方交握後所得到的 minislot 位置。這兩個時期都有部分機制會造成封包等待時間變長，我們將分析如後。

3.1.1 TO Interval

Mesh 網路中的 SS 在每次得到 TO 之後，就必須計算下一次的 TO 爲何，由於在 mesh 環境中的 TO 主要是用來發送 MSH-DSCH 信號，因此我們將兩個 TO 所間隔的時間定義爲一個「MSH-DSCH interval」。MSH-DSCH interval 的長度會隨著節點競爭的結果而不同。當節點數目越多，則節點周圍的 eligible neighbor 也增多，而 mesh election algorithm 只會爲每一個 TO 決定一個 winner，故節點取勝的機率降低。競爭失敗的節點會繼續競爭下一個 TO，因此節點越多，平均 MSH-DSCH interval 就會越長。而資料傳輸至少必需經過一次成功的 3-way handshake，故平均 MSH-DSCH interval 延長的效應會放大三倍，使每個封包平均延遲時間增加，若再考慮多跳環境的累積，封包延遲的問題將愈來愈可觀。

國立政治大學 National Chengchi University

3.1.2 Minislot allocation

在 WiMAX 標準中的 Data subframe 被分作 256 個 minislots，這些 minislot 就像是公用空間，只要傳輸兩方的周圍節點都尚未表態要使用，則這兩個節點即可在發送 MSH-DSCH 的時間進行三方交握，表明這兩個節點有意使用哪些 minislots。在 IEEE 802.16 標準中，minislot 的分配是以 slot_range 以及 frame_range 兩個參數來定義；slot_range 表示每一個 frame 必須要使用多少個 minislot，frame_range 表示這個需求會持續多少個 frame。Figure 8 是一個 request 的範例，灰色部分表示 control subframe，黃色部分是 data subframe。這個 request 的 frame_range 爲 4，slot_range 爲 2，因此在 frame n 到 frame n+3 的四個 frame 中，每個 frame 都分配給這個 request 兩個 minislots，如藍色部分所示。

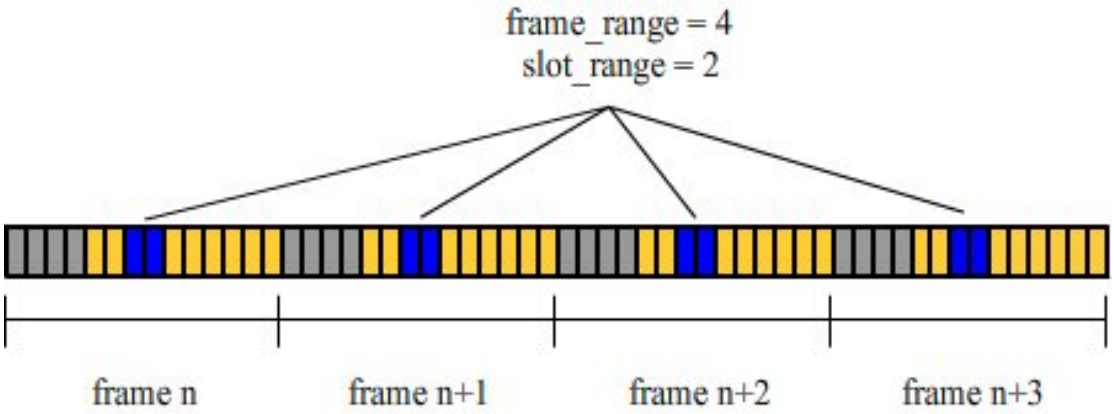

Figure 8: allocate minislot to a request

對於兩個相鄰的 SS 來說，它們的 minislot 相當於是共用的，因爲在同一個頻道上不能同時有兩個無線電信號被發送，因此每個 SS 都必須維護一個這樣的 bitmap，記錄在自己周遭的 minislot 使用情形。SS 透過 MSH-DSCH 的發送進行三方交握後，才可確定要在哪一個 minislot 進行資料傳送。同時經過三方交握後才能保證其他鄰點不會在同一時間內也進行傳輸。故在 Figure 9 中，我們將四個

相鄰之 SS 的 bitmap 排在一起，當鄰點進行傳輸的時段，就將自己的該時段標爲不可使用的灰白色。

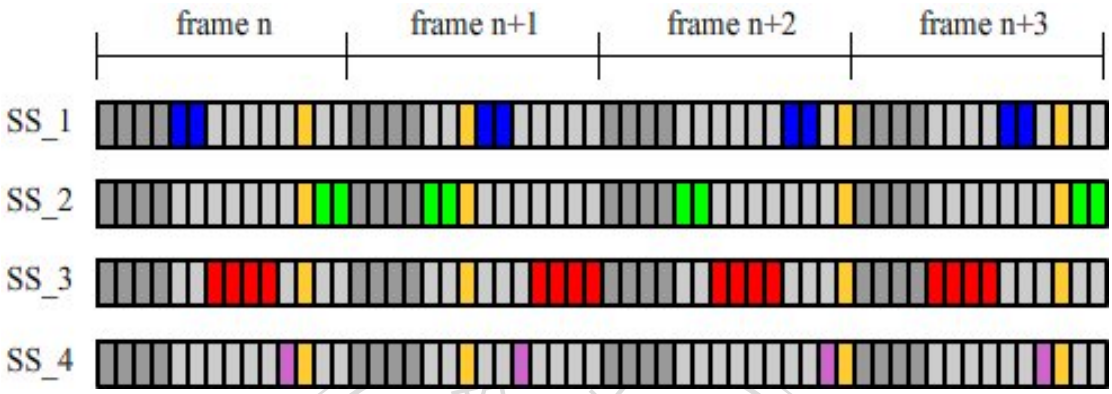

Figure 9: neighborhood minislot bitmap

爲了避免 hidden node 產生同時傳輸的干擾問題，三方交握的 grant 階段以及 confirm 階段都會將「使用哪些 minislot」的資訊廣播給鄰點。即使是某些 SS 只收到 grant 而未收到最後的 confirm，也會將 grant IE 中預定要進行傳輸的 minislot 標示爲不可使用，因此 minislot 的取用在 grant 階段就已經大致決定。換句話說，在 granter 端所進行的 grant 動作，包括 minislot 的數量、持續的長度、開始的位置都會影響到資料流傳輸的穩定程度，因此 granter 怎麼進行 minislot 的安置是一個重要的課題。

對於一個 TDMA (Time Division Media Access)系統來說，minislot 的開始時間具有價值上的不同，因爲 minislot 的開始時間越靠近目前時間，表示在越短時間內這個 minislot 就會到達，若將具有時效性的封包安排在較早發生的 minislot 進行傳輸，則可降低更多的封包延遲。然而，在 mesh 環境下卻不易作到這樣的安排，原因是當節點在安排 minislot 時，並無法知道除了自己以外其他節點的封包內容，逕行安排自身所要使用的 minislot 後，就會減少其他鄰點的 minislot 選擇，當短期內可用的 minislot 均被預約，節點就必須將接下來的 minislot 預約向

後延，逐漸就會造成未來預約的 minislot 都必須再等待許久，形成等待 media access 的 delay。我們以下面的例子說明：

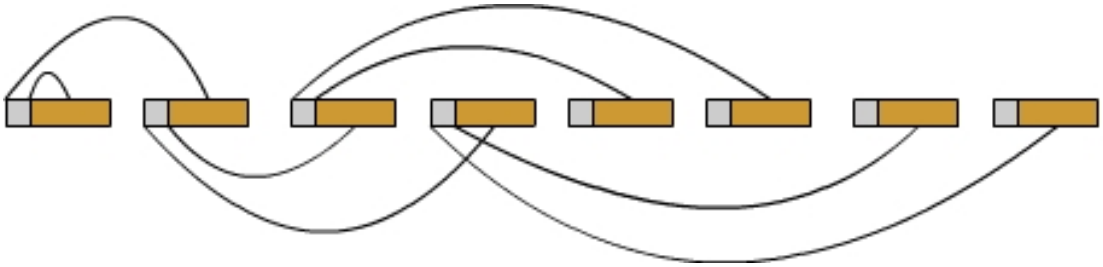

Figure 10: Minislot assignment increases waiting time

Figure 10 中 frame 的前半部爲 control subframe，後半部爲 data subframe。該圖表示當 data subframe 滿載的時候，control subframe 的預約會逐漸被推移到更遠的 data subframe。我們可從圖中觀察到，假若每個 control subframe 所指定使用的 minislot 數量是兩倍的 data subframe 時，第四個 control subframe 就被迫預定在兩個 frame 之後的 data subframe，造成 control subframe 與 data subframe 有兩個 frame time 的間隔。在高負載的環境下，這個間隔會逐漸造成等候時間被延長，直到最大可預約期限爲止。因此即使在 control subframe 已經確定得到了 minislot，卻有可能必須繼續等待一段時間後，傳輸所用的 minislot 才抵達，如此不但白白增加延遲，並且只要環境維持滿載，該延遲就不會縮小。相反的它總是維持在最大可預約期限，因爲超過最大預約期限的 request 會被 reject，而隨著 time frame 的推進，系統漸漸從最後插入新的 availalble minislot，因此新的 request 也一直被安排在最後，故依照排隊理論，間隔的延遲不會隨時間縮小，而會接近於最大可預約期限。

此問題是由於在原先機制中並無 QoS 的設計，所有服務都是等值的，故只要使 minislot 能盡量使用而非閒置即可達到最大使用效益。並且 minislot 的分配是由 SS 各自分散決定，缺乏通盤考量，故當節點盡力滿足目前 request 的同時，

卻造成可預約的 minislot 不斷向後推遲。這在一個負載較輕，節點數較少的環境下運作或許無礙，然而卻無法在負載升高時提供網路可靠度及服務穩定性，因此我們希望能改善 minislot 的分配機制，使節點對 minislot 的選取不只考慮傳送端與接收端的狀況，還將鄰接節點的 request 狀況一併考慮，使整個 neighborhood 範圍內較重要的 request 得以先行分配，避免過度分配 minislot 給低優先權的資料流，以達到 minislot 的有效利用、較高的服務效益，及提高網狀網路的整體傳輸效益。

3.2 研究方法與原因

本研究針對前節所提出產生延遲的原因，分別使用不同方法來減少傳輸延遲，以下介紹其概念：

在競爭 TO 階段中，最主要的問題在於節點的數量。節點數量越多，則參與競爭的節點數就越高，節點獲勝的機率就越低，必須經過更多次的競爭才能取得 TO。為降低競爭，WiMAX 定義了 holdoff time，因為節點無需在短時間內頻繁的發送控制信號，故節點送出控制信號後，就會有一段 holdoff time 不參與 TO 的競爭，也不會發送控制信號，以降低參與競爭 TO 的節點數量。適當的 holdoff time 長度應與活動中的節點數量有關，最理想的狀況是使節點要進行 TO 競爭時，大部分的鄰點都恰好處於 holdoff time，可以使競爭的狀況降到最低，也不增加不必要的延遲。

Holdoff time 的大小取決於 holdoff exponent，較小的 holdoff exponent 會使 holdoff time 較短，因此節點參與競爭的頻率較高，在 request 被 reject 後也能在較短的時間內再次提出 request。然而，過小的 holdoff exponent 也會造成 nxmt

interval 過短，容易在 interval 結束後仍然無法得到 TO。我們以 Figure 11 爲例，Figure 11 的下圖是假設 holdoff exponent 等於 3，mx 等於 3，則 nxmt 的範圍是從 25 到 32，可以連續競爭 8 個 TO。而上圖則假設 holdoff exponent 等於 1，mx 仍等於 3，nxmt 的範圍是從 7 到 8，只能競爭 2 個 TO。因此，我們將針對當時網路中節點的活動狀態，估算參與競爭者的多寡，動態調整節點的 holdoff exponent，使 holdoff time 的長度不至於過長而造成控制訊息交換太緩慢，也不使 nxmt 的長度過短，造成競爭成功的機率過低，減少因競爭而造成的媒體允入延遲。

Figure 11: Holdoff exponent and nxmt

此外，在進行 minislot 的安排時，原始的 802.16 機制並未進行資源控制，只定義了 TO 的競爭方式，因此每個節點得到 TO 時，就能對可使用的 minislot 進行安排。然而，不同的服務重要性也不同，若不針對節點可支配的 minislot 進行限制，則節點會傾向盡力滿足所有 request，亦即將 minislot 盡力分配出去。如此一來，容易造成節點將低優先權的 request 安置後，卻阻礙了其他鄰點的高優先權 request 配置，或被安置於較晚到達的 minislot，徒增封包的延遲。

分析上述的情況，我們發現 minislot 的分配問題是由於下述幾個原因造成，其一是 request 沒有優先權的設定，因此無法針對 UGS 或 rtPS 此類 delay-sensitive 的 request 給與較高的優先權；另外，即使我們對於 delay-sensitive 資料流進行優先配置，但節點發送 MSH-DSCH 的順序仍會造成優先權反轉的問題。我們以 Figure 12 與 Figure 13 的兩個例子來說明優先權反轉是如何發生的。

Figure 12 與 Figure 13 都有著相同的初始條件，圖中四個節點均彼此相鄰，A 與 C 各向 B 與 D 提出 3 個 slots 的 request，並將自己的 available slot 送交出去。B 的第 3 與第 4 個 slot 處於使用狀態，以深灰色標示。Figure 12 與 Figure 13 唯一的差別在於：Figure 12 中，B 的 TO 順序早於 D，故 B 會比 D 早一步送出 grant，而 Figure 13 則是 D 比 B 早送出 grant。

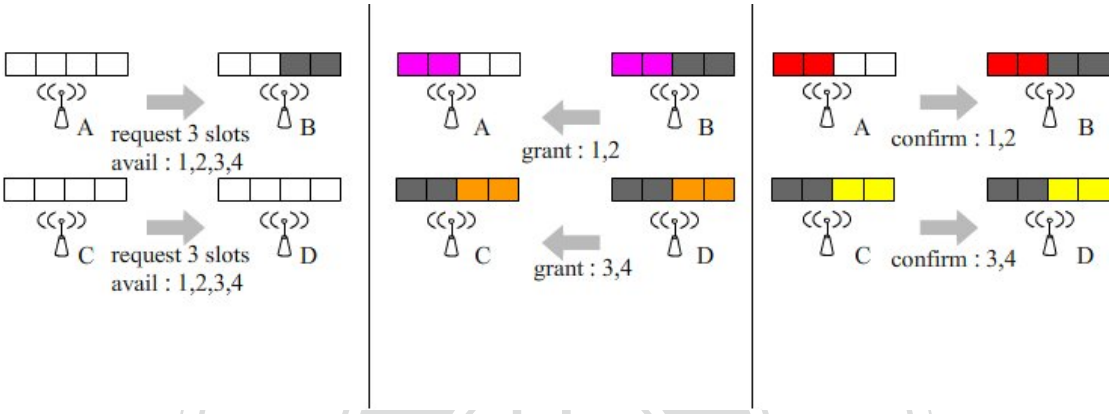

Figure 12: node B's grant is earlier than node D's

在 Figure 12 的情況中，B 所使用的 TO 早於 D，因此在 grant 階段時，B 先決定所要使用的 minislots，由於第 3 與第 4 個 slot 不能使用，B 只能回應給 A 兩個 slots，故 B 送出 grant{1,2}的信號給 A。而此信號也同時發給整個 neighborhood 的其他節點，故 C 與 D 均假設 slot{1,2}將有人使用，故我們將 slot{1,2}標示爲「已佔用」的深灰色。此時 D 能 grant 的部份就只剩下 slot{3,4}，因此 D 送出 grant{3,4}給 C。最後 confirm 階段即可看到分配的結果：兩組通訊各獲得兩個 slots。

然而，若將 B 與 D 的 TO 順序交換，即形成 Figure 13 的情況，在 grant 階段時，由於 D 發出 grant 時，四個 slots 均可使用，故 D 送出 grant{1,2,3}給 C，而 A 與 B 將 slot{1,2,3}標示爲「已佔用」。但此時 B 會發現 minislot 已全數用盡，故回應 grant{none}給 A，因此最後的結果是 A 與 B 沒有獲得 slot，而 C 與 D 使

用 slot{1,2,3}的情況。

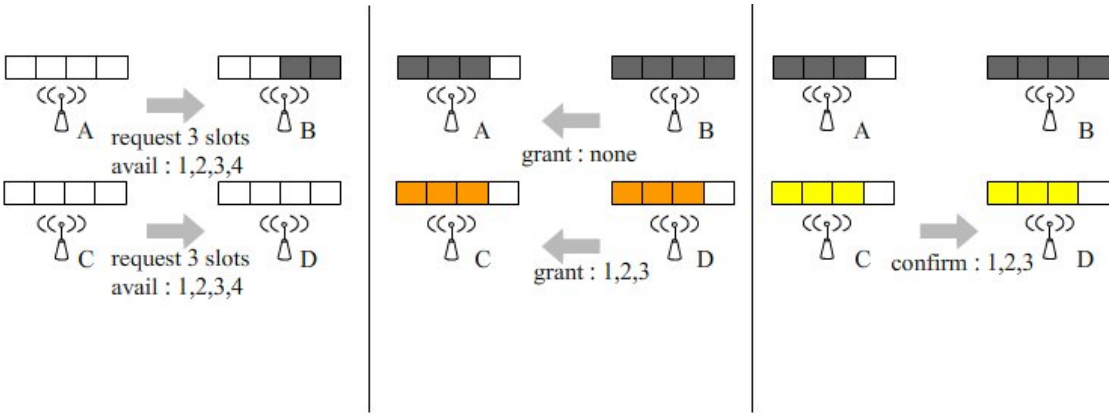

Figure 13: node B's grant is earlier than node D's

由 Figure 12 及 Figure 13 的例子中，我們看到不同的 TO 順序會造成不同的 minislot 分配結果，並對於 TO 順序較先的節點較有利。然而，若 Figure 13 中的 A 與 B 正要進行 rtPS 的連線，而 C 與 D 只是等級較低的 BE 連線，就會形成優先權反轉的問題。我們希望改變這種以 TO 順序爲主的分配方式，使 minislot 的分配能針對整個 neighborhood 範圍進行考量，使整個 neighborhood 範圍中的低優先權資料流不至佔用到高優先權資料流的頻寬。

我們提出一個 Neighborhood-Based Minislot Allocation (NBMA)的作法來解決 minislot 分配的問題。NBMA 的作法是先評估各個 request 的重要性，以 request 被拒絕的成本作爲評估重要性的因子(importance factor)，並與其他鄰點交換重要性因子的分佈曲線，統計出 neighborhood 範圍的整體分佈曲線後，即可歸納出重要性最高的一群 requests。各節點即可知道自身所收到的哪些 requests 應優先處理，並得知需保留多少 minislots 給其他 neighbors，如此即不會發生優先權反轉的情況，降低不當的 minislot 分配造成媒體允入延遲。

國立政治大學 National Chengchi University

3.3 研究進行步驟

本研究的進行步驟可分爲三個部分。第一部分是增加 WiMAX mesh mode 對於不同服務型態的資料流提供優先權的支援，依照不同的 QoS 等級給與 request 不同標記，使節點接收 request 後，能依照目前網路的環境來調整 Holdoff exponent 及 minislot 的分配方式。第二部分是將原先的 static holdoff exponent 改變爲 dynamic holdoff exponent，使節點依照周圍節點的活動情況來調整 holdoff exponent，設定適當長度的 holdoff time，減低 TO 的競爭程度。第三部分是透過一個評量機制，對自身接收到的 request 進行重要性估計，並與其他鄰點交換彼此的重要性分佈情形，藉以得知在 neighborhood 範圍內的重要性排序，最後依照 available minislot 的數量，來決定過濾的門檻值，使 minislot 能確實分配給重要性最高的 request。我們將 IEEE 802.16 的流程繪製如 Figure 14，而增加及修改的機制則繪製爲 Figure 15。Figure 15 增加了一個 QoS 分類器，將四種不同 QoS 形態的封包分送至不同的佇列中分別處理，同時透過 dynamic exponent 的機制縮短 3-way handshake 的長度，並在 MSH-DSCH 訊號中加上 IM-IE (Importance-factor Information Element)資訊，以 NBMA 的方法來縮短 minislot 分配時的延遲時間。

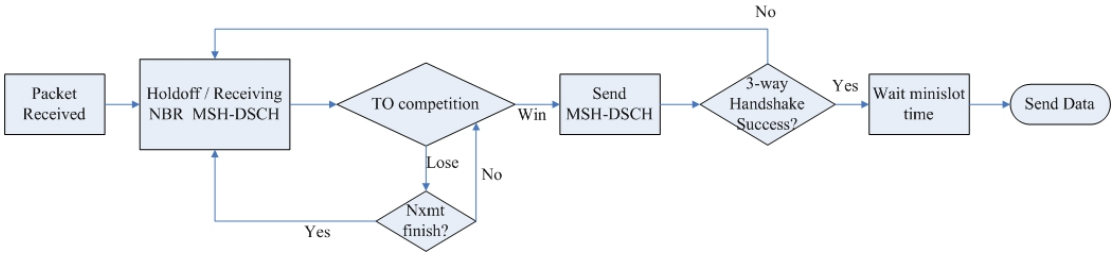

Figure 14: The original IEEE 802.16 bandwidth allocation flow chart

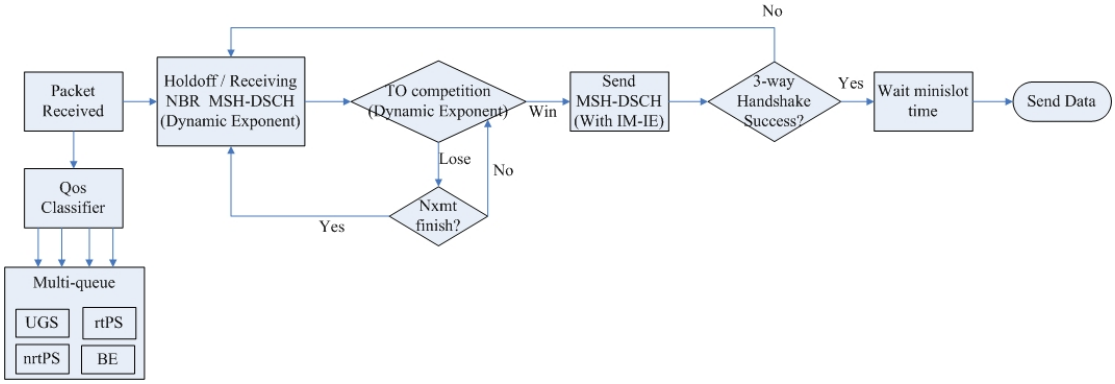

Figure 15: The new IEEE 802.16 bandwidth allocation flow chart

3.3.1 Delay-Sensitive Request and Virtual Node

本論文期望達到的目的是降低 IEEE 802.16 分散排程環境中，在媒體允入階段產生的延遲現象。由於不同 QoS 型態的 traffic 對於 packet loss、delay、jitter 的需求不同，因此在處理 media access 時必須依 QoS 型態分開處理。爲了與 PMP 模式相容，我們亦採用 PMP 模式的四種 QoS 階層，即 UGS、rtPS、nrtPS、BE 四種層級，使其取得頻寬的優先權不同，並利用 Multi-queue 及 virtual connection 來分離不同型態的資料流，使 delay-sensitive traffic 能盡早被服務，減少延遲的發生。

我們採用 Kuran [6]的 virtual node 概念，Kuran 的 virtual node 技術主要是用來讓四個層級的 traffic 分開進行 request，將所有 request 提交給 BS 後統一安排 minislot，故四種 QoS 型態的頻寬各自獨立，不互相流用，SS 也不再依現有頻寬對封包進行重新排程與分配。而在本研究中，我們使用 virtual node 作爲四種 QoS 型態的對應，並對 virtual node 之間建立的 virtual connection 指派不同的 connection ID (CID)，如 Figure 14 所示。使 MAC 在進行 frame 封裝時，將相同 QoS 等級的封包以相同的 CID 封裝於同一個 mac frame 中，故下一個接收點或是中繼點即可由 mac 標頭中的 CID 資訊辨識 QoS 等級，解除 frame 封裝後即可將封包擺放至

國立政治大學 National Chengchi University

對應等級的佇列中。如此，資料流在傳送過程中就不需重覆進行分類。

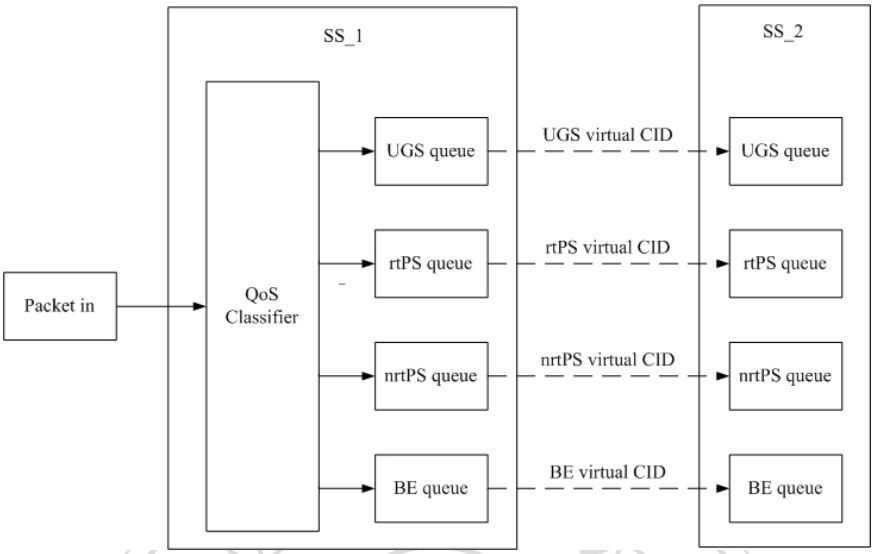

Figure 16: virtual node and virtual connection

有了 virtual node 的設計，節點就必須採用 multi-queue 的方式來儲存封包，並且依照各個佇列的封包到達率來提出頻寬需求。我們爲每一個 virtual connection 產生一個 virtual connection ID。Virtual connection ID 是用在填入 MSH-DSCH 中的 request IE、grant IE 的 Link ID 欄位，以辨識頻寬被分配給哪一個 virtual connection。因爲這些 virtual connection 是作爲 QoS 對應之用，request 也因不同的 Link ID 而能被辨識出 QoS 等級，故我們將四種等級不同的 request 稱爲 UGS-request、rtPS-request、nrtPS-request 及 BE-request。其中 UGS-request 和 rtPS-request 有較嚴格的延遲限制，將其歸類爲 delay-sensitive request；至於 nrtPS-request 及 BE-request 則屬於 non-delay-sensitive request。透過不同的 QoS 標記，節點就可依 request 的 QoS 型態進行優先權評估，決定每個 request 的重要性。Figure 17 中的上半表格是 802.16 標準中 MAC 標頭部分的 CID 欄位，下半表格是在 Request IE 中的 Link ID 欄位，我們將其更改爲 virtual connection ID 的數值即可。

Syntax	Size	Notes
CID {		
if (Xmt Link ID == 0xFF) {		
Logical Network ID } else {	8 bits	0x00: All-net Broadcast
Type	2 bits	0x0: MAC Management 0x1: IP 0x2-0x3: *Reserved*
Reliability	1 bit	0x0: No retransmissions 0x1: Up to 4 retransmissions
Priority/Class	3 bits	
Drop Precedence	2 bits	
}		
Xmt Link ID	8 bits	0xFF: MAC management broadcast
}		

Syntax	Size	Notes
MSH-DSCH_Request_IE() {		
Link ID	8 bits	
Demand Level	8 bits	
Demand Persistence	3 bits	
reserved	1 bit	Shall be set to zero.
}		

Virtual Connection ID

Figure 17: replace “Link ID” and “Xmt Link ID” to virtual connection

3.3.2 Dynamic Holdoff Exponent

在 IEEE 802.16 標準中，Holdoff exponent 值的長度爲 3 bits，範圍可從 0 到 7。這個值的大小不只影響 holdoff time 的長度，也影響了 next transmission interval 的長度。Holdoff time 的長度越長，表示節點就有越長的時間不能參與競爭，但也能增加 nxmt 的長度，使節點增加連續競爭 TO 的次數，提高取得 TO 的機率，然而，若在 nxmt 期間都未能競爭成功，節點就必須再次進入 holdoff time，如此會加長 MSH-DSCH 的間隔，並增加封包的延遲時間。換句話說，將 holdoff exponent 增加的結果是增加競爭獲勝的機率，但也會增加競爭失敗時的 holdoff time，並降低 MSH-DSCH 的發送頻率，在節點數較多的環境下能降低競爭情況。因此本研究希望能根據網路負載的狀況，動態調整 holdoff exponent，使僅有

National Chengchi University

non-delay-sensitive 封包需傳輸的節點延長 holdoff time，加快其他節點處理 delay-sensitive traffic 的速度，使 TO 的競爭更有效率。

Dynamic Exponent 是引用 Bayer 的概念。Bayer 考慮到節點並非時常都需要傳送資料，但當節點要傳送資料時，就必須頻繁的發送 MSH-DSCH 訊號，以便跟鄰點交換 schedule，並進行 3-way handshake 來決定傳送資料的 minislot。因此縮短 Mesh-BS、sponsor node、active node 的 holdoff exponent 以加快 handshake 速度。Bayer 所定義的 holdoff exponent 按照下列的規則：

$$\boldsymbol{0 < mesh\text{-}BS < active\ node < sponsor\ node < non\text{-}active\ node < 7} \tag{7}$$

然而，Bayer 的規則卻忽略了當網路處於擁塞狀態時，節點會時常處於 active 的狀態，故 Bayer 以節點是否爲 active 狀態來調整 holdoff exponent，無法達到動態調整 holdoff exponent 的效果，而會退化爲 static 的 holdoff exponent。因此我們將原本 Bayer 所定義的 active node 及 sponsor node 改變爲 delay-sensitive (DS)與 non-delay-sensitive (Non-DS)兩種等級。對 UGS-request 與 rtPS-request 這類 delay-sensitive request 設定爲較短的 holdoff time；而對 delay 容忍度較高的 nrtPS 與 BE 設定爲較長的 holdoff time。我們也考量到在兩個節點的 holdoff time 不等長的情況下，requester 的 holdoff time 較短可使已被 granter 預定的 minislot 較不易被鄰點搶先使用，故 delay-sensitive 設定爲較小的 holdoff exponent，才能盡快進行 confirm。而 non-delay-sensitive 則設定爲較大的 exponent，使 delay-sensitive 足以搶先，因此我們將規則調整如下：

$$0 < mesh\text{-}BS < \underline{DS} < \underline{Non\text{-}DS} < non\text{-}active\ node < 7 \tag{8}$$

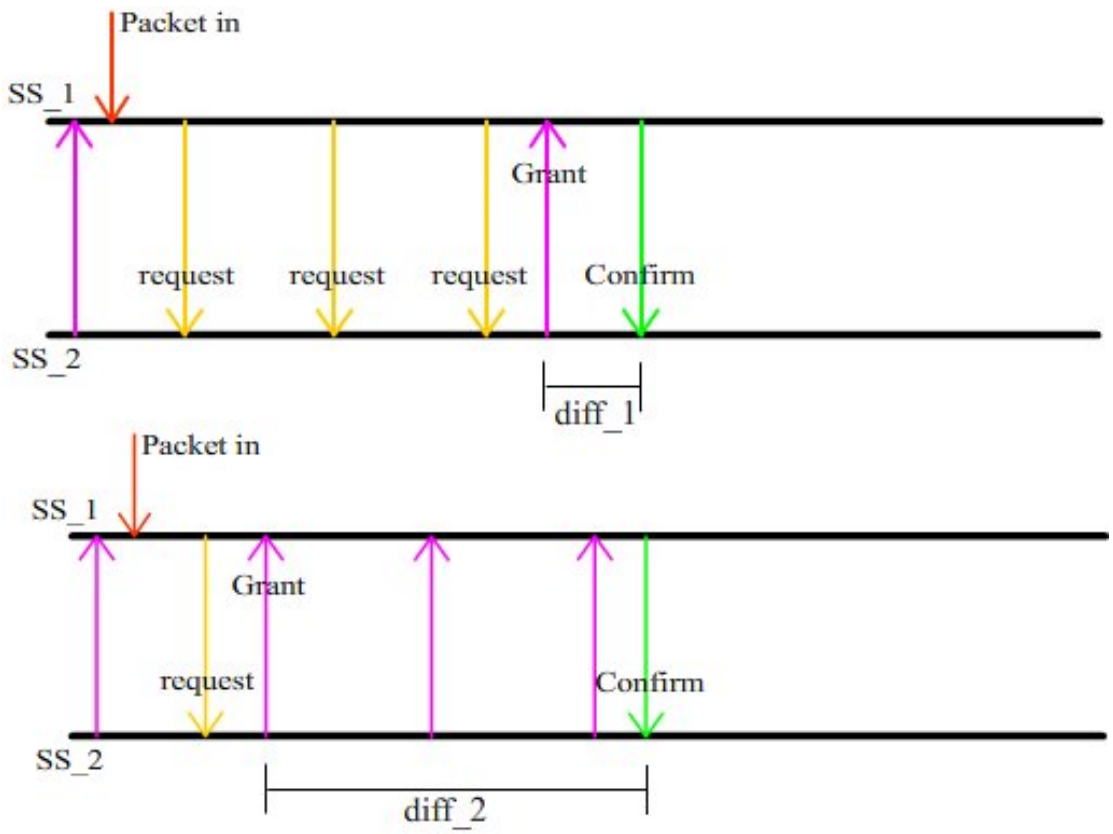

Figure 18: Distance of grant and confirm

在競爭 TO 時最理想的狀況，是每個節點的第一個 nxmt 都恰爲其他 neighbors 的 holdoff time，這樣節點就無需進行競爭，並順利取得第一個 TO。換句話說，也就是自身的 holdoff time 長度恰足夠使所有鄰點發送一次 MSH-DSCH，故我們可以得到 holdoff exponent 的最低要求：

$$2^{\exp+4} \geq number\ of\ neighbor \tag{9}$$

經過化簡可以得到下式：

$$\exp \geq \lceil \log(number\ of\ active\ neighbor) \rceil - 4 \tag{10}$$

我們將上式作爲 delay-sensitive traffic 的標準，並依照 3 式的定序規則，最

小不能低於 0，將每個 node 狀態的 Dynamic Exponent 定義如下：

- MBS : $\max\{0, \lceil \log(number\ of\ active\ neighbor) \rceil - 5\}$
- Delay-sensitive : $\max\{0, \lceil \log(number\ of\ active\ neighbor) \rceil - 4\}$
- Non-delay-sensitive : $\max\{0, \lceil \log(number\ of\ active\ neighbor) \rceil - 3\}$
- In_active : 7

若自身是 Mesh BS，則狀態固定爲 MBS，除此之外，節點狀態的轉換是由自身節點的 request IE 來決定，當節點的 UGS-queue 或是 rtPS-queue 有封包進入，則在 MSH-DSCH 中放入 UGS-request IE 或是 rtPS-request IE，在最近一個 MSH-DSCH 發送時，一併將新的 holdoff exponent 發出。而一個 MSH-NCFG 期間內都沒有進行 request 的節點可將其設定爲 In_active。

3.3.3 Request Scoring

將節點的活動狀況與 holdoff exponent 結合後，可以確保高優先權資料流通過節點時，可提高節點的控制信號交換頻率。此外，我們也要在網路負載量提高時，將頻寬優先分給重要性較高的 request，故我們須針對各個 request 進行重要性的評估，最後再依重要性的排序來分配目前所剩餘的可用時槽。

我們將 request 的重要性定義爲「request 被 reject 所造成的負面效應」。若 request 被評估爲「reject 後的負面效應大」，則表示此 request 應該優先被處理。我們從 packet loss、delay、jitter 三方面來評估其負面效應，爲每個 request 定義其重要性因子，作爲 request 的比較標準，重要性因子高的 request 將被優先服務。

國立政治大學 National Chengchi University

3.3.3.1 Packet loss rate

無線網路中的封包流失主要肇因於以下幾種原因，第一是無線電波的干擾導致封包無法解析，因此判定爲未收到封包。第二是網路壅塞，封包可能在傳送過程中送入一個佇列滿載的中繼點，被迫丟棄某些封包。第三是傳輸過程中消耗太多時間，造成傳輸層的 TCP time out，此時封包可能還存在中繼點的佇列中，也可能確實已因干擾而被消滅。TCP 協定雖無法確認原因，但會認定封包已因不明原因流失，因此兩端的 sliding window 都會被放棄，重新進行傳輸，此時封包亦被視爲已流失。

由於干擾導致的封包流失必須透過 PHY 層的 OFDM 調變技術或 CRC 修復來克服，故我們在此暫時不考慮 PHY 層的因素。至於傳輸過程使用過長的時間則多半也是肇因於中繼點等待傳輸的封包過多，造成佇列消化的速度過慢，因此前段所述的第二及第三兩個問題多半發生於一個長度過長的佇列，通常這個佇列的特徵是封包離開佇列的速度小於進入佇列的速度，亦即中繼點在接收封包後，向下一個接收點要求頻寬失敗引起的，故此項與 request 被 reject 有密切的關係。因此，我們除掉 PHY 層的因素後，以「request 的 minislot 資料量」與「佇列剩餘空間」的比例作爲 packet loss rate 的危險指標，來衡量 request reject 後對 packet loss rate 可能產生的危害程度。

關於「request 的 minislot 資料量」意謂著原先的 request 若被接受，則節點將被賦予的 minislot 總量，也等同於節點因成功取得頻寬，所能從佇列中 pop 出來的資料量。然而，正因爲 request 未成功，故這些資料仍被留在佇列中，我們將其與佇列剩餘空間進行比較，當佇列越接近滿載時，這些被留置的封包就越有可能被丟棄，也就是在佇列中的「儲存成本」越高，故造成更大的負面效應。

國立政治大學 National Chengchi University

至於「佇列剩餘空間」的估算，必須以「現有佇列實際可用空間」再加上「request 的 minislot 資料量」，因爲在 request 被拒絕時，等待儲存的資料其實已經被儲存在佇列中。爲了估計擺放資料的成本，必須先以「資料尚未占用空間之前的可用空間」爲分母，其估算公式如下：

X : MSH-DSCH 期間內所能傳輸的資料量

MaxQ : 佇列最大儲存空間

CurQ : 佇列目前使用空間

IM_PLR: 封包流失率的重要性因子

$$IM_PLR = \frac{X}{[MaxQ - (CurQ - X)]} \tag{11}$$

3.3.3.2 Delay

另一個衡量標準是封包的延遲時間(end-to-end delay)，亦即封包在網路中傳送所花費的時間。在 IEEE 802.16 的網狀架構中，封包須透過 SS 的轉送，才能到達目的地；由於每一次的轉送都必須進行三方交握，因此當 request 被 reject 時，就必須等待下一次 TO 才能重新提出 request，再試圖進行一次三方交握，這過程將額外增加封包被延遲的時間。

節點的三方交握是在 control subframe 中的 TO 時間內送出的。一個 TO 僅能供給一個 2-hop neighborhood 範圍一次發送 MSH-DSCH 的機會，因此對一個節點來說，其 MSH-DSCH 的間隔是由 holdoff exponent 及與鄰點的競爭結果而定。

國立政治大學 National Chengchi University

由於我們採用 dynamic holdoff exponent，故兩個節點的 holdoff exponent 不一定相同，再加上競爭 TO 會使 MSH-DSCH 的長度略增，故將 requester 與 granter 的 MSH-DSCH 間隔相比有下列三種情形：

第一種情況下 SS_1 的 MSH-DSCH 間隔較 SS_2 為大，由下圖可以看出原本若 request 成功取得 minislot，則應該在 SS_1 第二次發送 MSH-DSCH 時就完成三方交握，但由於 request 失敗，故須再延後到 SS_1 第三次的 MSH-DSCH 發送時間，故 reject 一次造成的 delay 就等於 SS_1 的一次 MSH-DSCH 間隔，如 Figure 19 所示。

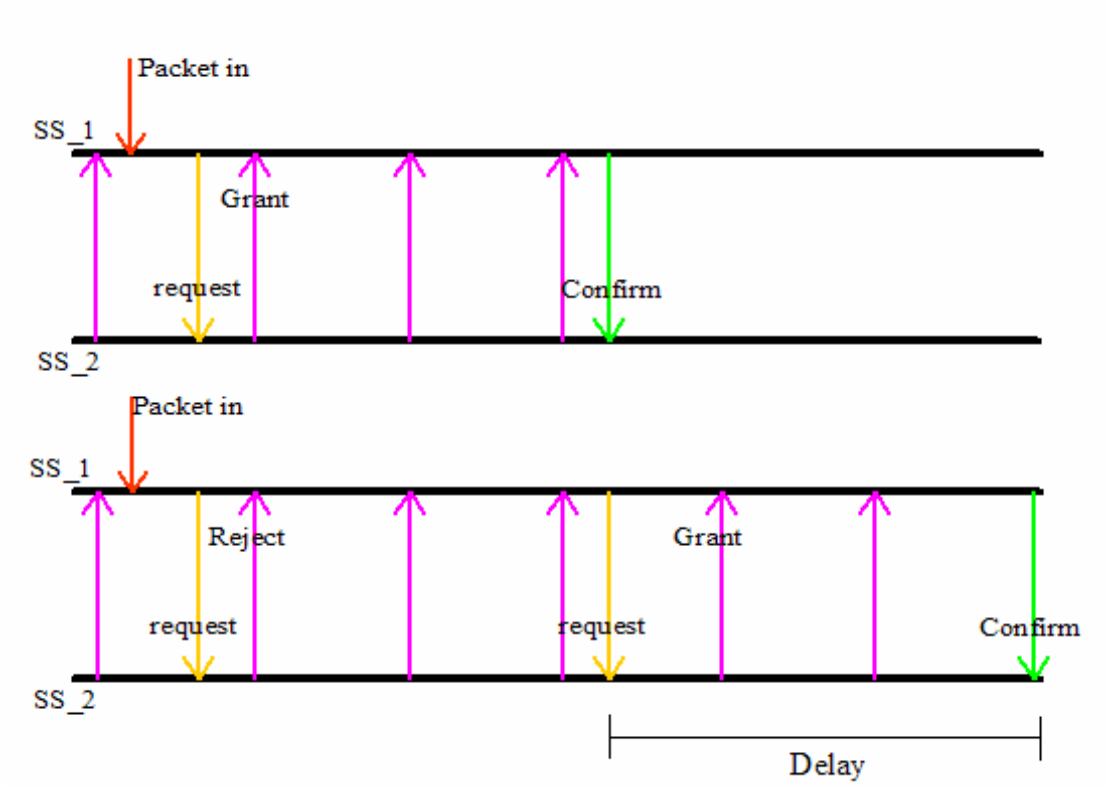

Figure 19: Delay of request rejected, this delay equals to SS_1's MSH-DSCH interval

第二種情況下 SS_1 的 MSH-DSCH 間隔較 SS_2 為小，如 Figure 20 所示，從 Figure 20 可看出原本應該結束於上圖的 confirm，但卻因為 reject 而移到了下圖的 confirm，這個長度不易直接從 SS_1 的角度描述，因為實際上我們並不知道會經過多少個 SS_1 的 MSH-DSCH 間隔；但是我們觀察到兩次 confirm 的結束都

恰爲 SS_2 的 grant 信號之後的第一個 SS_1 MSH-DSCH 時間，因此我們以 SS_2 的 MSH-DSCH 間隔作爲衡量基準，再分爲兩種最大和最小的情況。

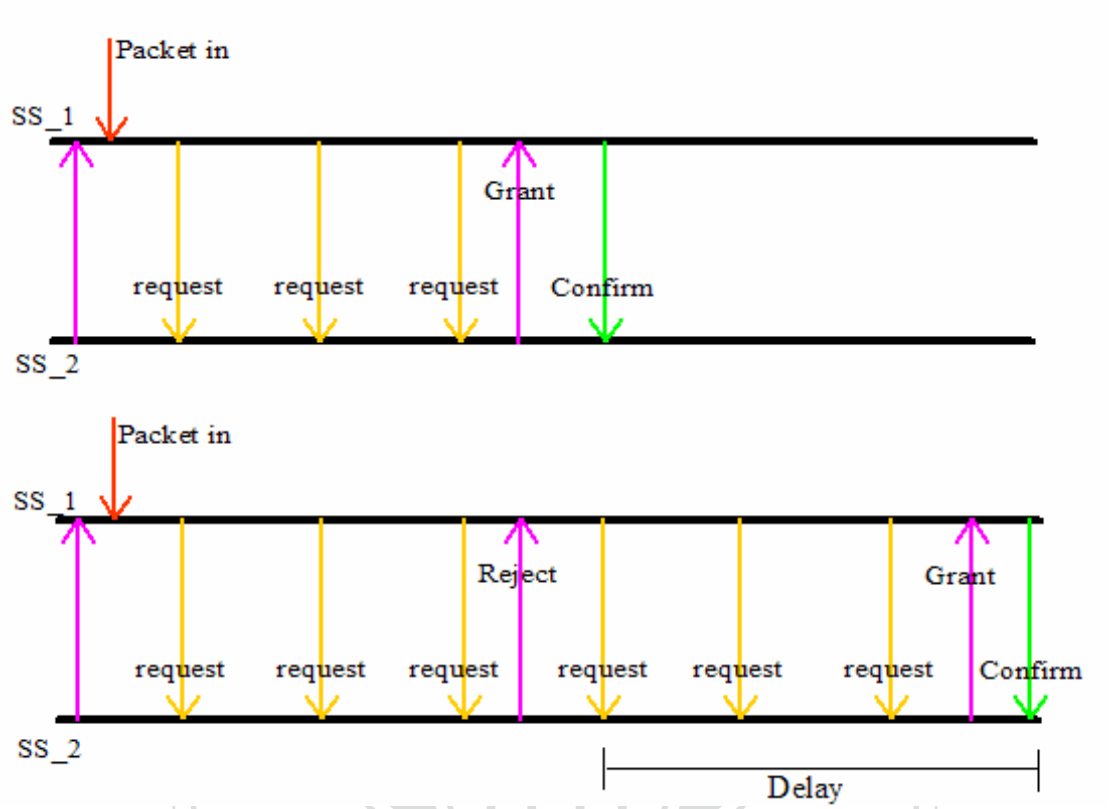

Figure 20: Delay of request rejected, this delay is close to SS_2's MSH-DSCH interval

最小的 min_delay 情況是使上圖的 confirm 盡量遠離 grant，且下圖的 confirm 盡量靠近 grant，如下頁 Figure 21 的情形。最大的 Delay_cost_2 則反之，如下頁的 Figure 22 所示，故最小長度爲「SS_2 的 MSH-DSCH 間隔 - SS_1 的 MSH-DSCH 間隔」，最大長度爲「SS_2 的 MSH-DSCH 間隔 + SS_1 的 MSH-DSCH 間隔」。

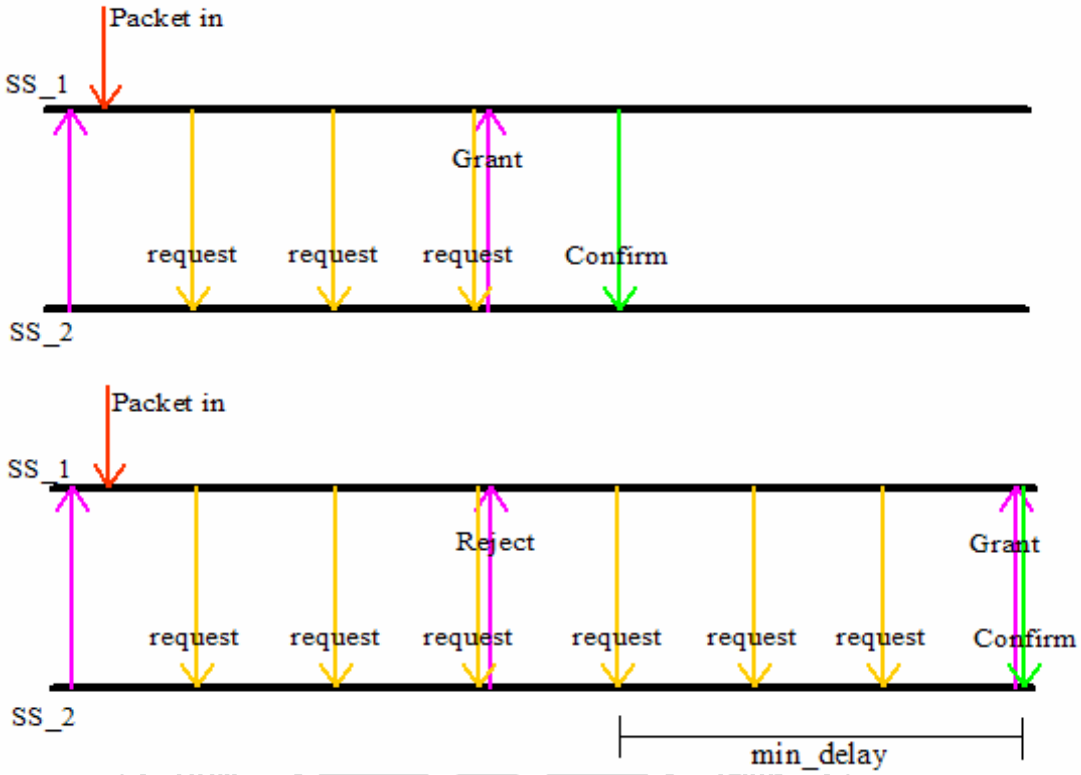

Figure 21: Minimum delay of request rejected, this delay is close to SS_2's MSH-DSCH interval minus SS_1's MSH-DSCH

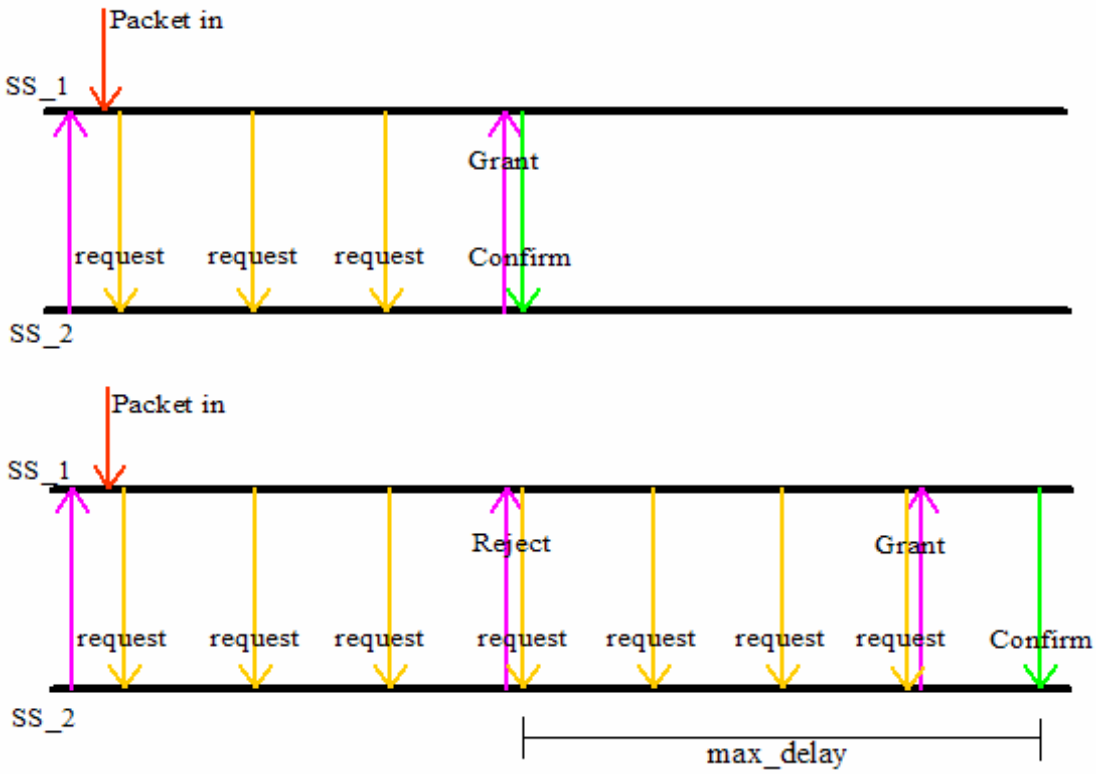

Figure 22: Maximum delay of request rejected, this delay is close to SS_2's MSH-DSCH interval plus SS_1's MSH-DSCH

第三種情況下 requester 與 granter 的 MSH-DSCH 間隔相同，在此種情況下，request 被 reject 的代價固定爲任一個節點自身的 MSH-DSCH 間隔。

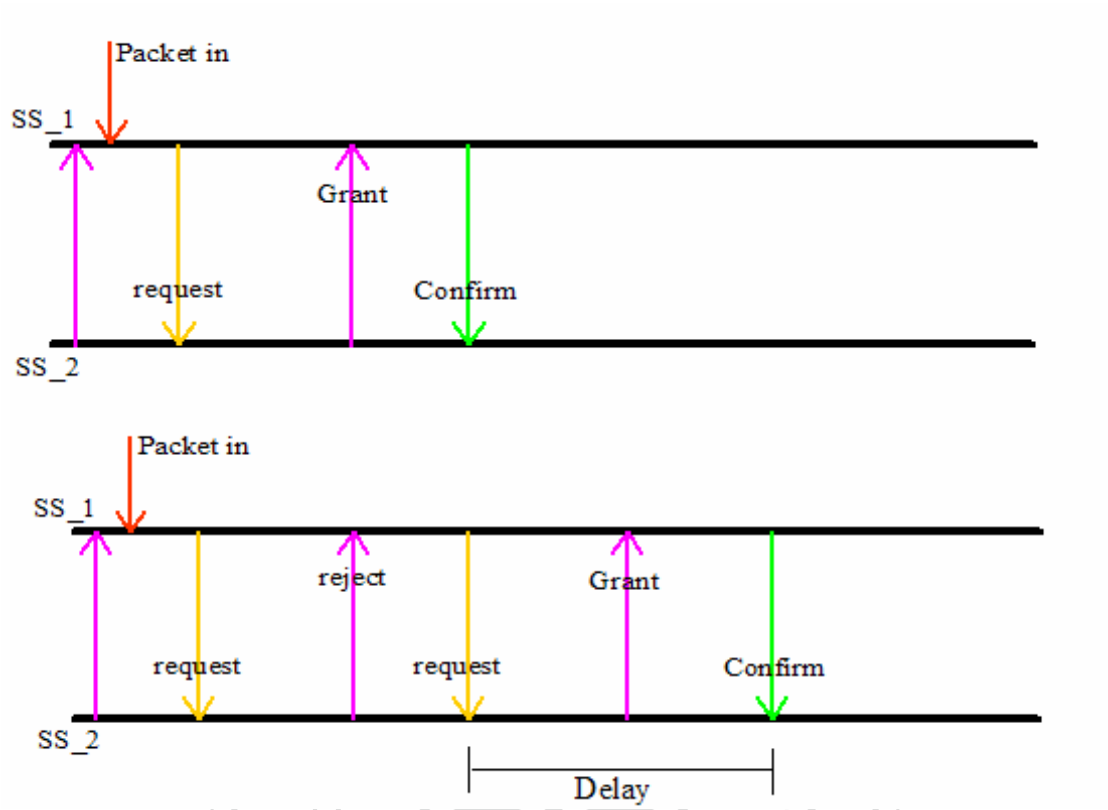

Figure 23: Delay of request rejected, this delay equals to SS_1's MSH-DSCH interval

因此綜合以上三種情形，節點可比較自身與 requester 的 holdoff 大小關係，估計出「可能延遲時間」，且若發生連續 reject 的情況，必須將「可能延遲時間」累計，再由系統提供的 maximum latency 以及節點相對 BS 的 hop count 算出 MaxLatencyPerHop，再計算「可能延遲時間」與「MaxLatencyPerHop」的比值，作爲 reject request 的負面效應值。

reject_delay：目前若 reject request 會造成的延遲，即「可能延遲時間」。

Reject_count：目前被連續 reject 的次數。

MSH-DSCH length：節點兩次 MSH-DSCH 訊號的間隔，分爲下列三種情況

$$\begin{cases} 2^{\exp_{ss_1}+4} & , \quad if\ \exp_{ss_1} > \exp_{ss_2} \\ 2^{\exp_{ss_2}+4} \pm 2^{\exp_{ss_1}+4} & , \quad if\ \exp_{ss_1} < \exp_{ss_2} \\ 2^{\exp_{ss_1}+4} & , \quad if\ \exp_{ss_1} = \exp_{ss_2} \end{cases}$$

(12)

Max latency per hop：在 QoS 品質可接受的前提下，封包每經過一個 hop 最多可忍受的延遲秒數。

System maximum latency：系統 QoS 參數，表示每種 QoS 等級最多可忍受的 end-to-end latency。

Syn_hop：節點經同步化過後的 hop 數，亦即節點到 BS 需經過的 hop 數。

IM_delay : 延遲重要性因子。

$$reject_delay = \sum_{k=0}^{reject_count} MSH - DSCH\ length$$

(13)

$$MaxLatencyPerHop = \frac{SystemMaximumLatency}{Syn_hop}$$

(14)

$$IM_delay = \frac{reject_delay}{MaxLatencyPerHop}$$

(15)

3.3.3.3 Jitter

Jitter 通常與 delay 共同發生，當資料流的傳輸不穩定時，不只會增加封包傳送的時間，也會造成通訊不順暢。當資料流的型態具有互動性或及時性時，瞬間通訊中斷容易造成影音不流暢，及服務品質低落，使用者無法接受的問題。

在 WiMAX mesh 架構下，request 被 reject 時亦會造成 jitter，並且我們評估的方式也與 delay 類似，以 reject 的時間點一直到下一次重提 request 這段時間作爲 jitter 的衡量值，亦採計連續 reject 的狀況，不同的是 jitter 在多次跳躍的狀況下不一定會累積，因爲必須考慮中繼點自身的佇列狀況。當自身的 UGS-queue 或 rtPS-queue 不爲空時，表示此中繼點即使要繼續進行轉送，也會先轉送目前這兩個 realtime-queue 裡的封包，故資料流不一定會因爲這次的 reject request 而中斷，因此在採計連續 reject 的狀況時會與 delay 略有不同。

Reject_count : 目前被連續 reject 且自身佇列爲空的次數。

$$reject_delay = \sum_{k=0}^{reject_count} MSH - DSCH\ length \tag{16}$$

最大的 jitter tolerance 也是由系統給定，每個節點的最大 jitter tolerance 都相同，故重要性因子爲：

$$IM_jitter = \frac{reject_jitter}{SystemMaximumJitter} \tag{17}$$

3.3.3.4 Scoring

進行過 request reject 的三種評估過後，我們需確定每個 request 的重要性有多高。由於 IEEE 802.16-2004 的 request 型態共有四種，而不同的服務型態對於 packet loss rate、delay、jitter 各自有不同的需求。因此在進行綜合評價時，我們採用不同的權重來進行加總，下表爲 IEEE 802.16 pmp mode 下四種 QoS 等級對於此三項評估標準的相關程度。同時我們利用下圖公式來進行加總：

國立政治大學 National Chengchi University

$$IM_factor = W_1 * IM_PLR + W_2 * IM_delay + W_3 * IM_jitter \tag{18}$$

	Packet Loss	Delay	Jitter
UGS	Low	High	High
rtPS	Low	Medium	High
nrtPS	High	Medium	Low
BE	High	Low	Don’t care

Table 1: QoS level and performance factor

權值 W_1,W_2,W_3 會影響重要性因子的分佈狀況，因此我們將三個 IM_factor 常態化(Normalization)後，再爲四種不同 QoS 等級的 request 賦予不同的 W_1,W_2,W_3 權值組合，便可得到四個常態曲線的分佈情形。適當的權值能使四個常態曲線的中位數錯開，且其重要性因子依序爲 UGS > rtPS > nrtPS > BE，之後依重要性排序來對 request 進行 grant 時，便能使大部分的 UGS 及 rtPS 資料流優先被服務，而少部分重要性較高的 nrtPS、BE 的 request 亦不至於被犧牲。

權值 W_1,W_2,W_3 可交由 ISP 業者依其定價策略及每週流量的分佈來進行調整，調整時需注意四個等級的中位數及四分位點之位置。因爲中位數位置表示該 QoS 等級的重要性因子落點分佈最密集之處，而四分位點則反映權值設計是否造成重要性因子的落點過度集中。若能使四個中位數與八個四分位點適度交錯，則表示權值設定合理，可使低優先權資料流不至發生 starvation 的狀況。至此，我們可以得到每一個 request 所相應的重要性指數，稍後我們將爲每個節點的 neighborhood 範圍計算過濾門檻，以此指數是否跨越過濾門檻作爲 request 是否應被 grant 或是 reject 的評估標準。

國立政治大學 National Chengchi University

3.3.4 Neighborhood distribution

在 IEEE 802.16 原有的頻寬分配機制中，節點在分配頻寬時僅考慮向自身所提出的 request，而不需考慮鄰近節點的情況，故會出現低優先權的 request 會被先行安置的不適當行爲。因此我們的構想是讓節點能得知在這個 neighborhood 範圍內的最低門檻值，唯有重要性指數超過該門檻，才能進行 minislot 安排，使 minislot 得以空出給 TO 順序較遲的鄰點，而非盡力滿足自身所接收到的 request，以避免 available minislot 被分配殆盡，造成預約間隔的延長。

該門檻需透過每個節點交換其 importance factor distribution 來決定，我們讓每個節點統計自身的 importance factor distribution，並將其製作成一個 information element，稱爲 IM-factor IE，安置在 MSH-DSCH 訊號當中發送出去，節點在 MSCH interval 當中需收集 neighborhood 範圍內其他 neighbor 的 IM-factor IE，並統計目前 neighborhood 範圍內的 importance factor distribution。同時，也計算目前剩餘的 available minislot 數量，以估計目前的 available minislot 最多能提供給多少 request 服務，以 neighborhood importance factor distribution 推算最低的 importance facot 門檻爲何。之後，以此最低門檻來過濾 request，未達最低門檻值的 request 即可 reject，超過門檻值且可找的到兩端點共同的 minislot，則進行 grant。

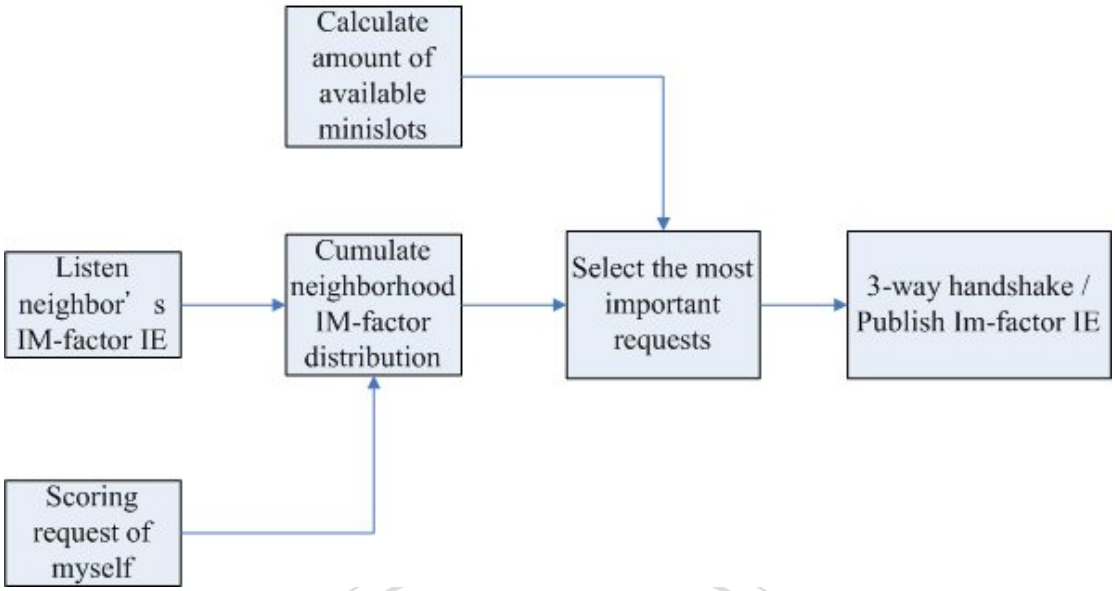

Figure 24: NBMA flow chart

此作法的優點在於，對每個節點來說，neighborhood 的構成節點雖不盡相同，然而每個節點都只允許 neighborhood 範圍內最重要的 request 存取 minislot，故對於網路壅塞嚴重的情況可以減輕 delay-sensitive traffic 網路品質降低的問題。另一方面，由於相鄰節點的 neighborhood 組成會有相當大的重覆性，故就 neighboorhood 範圍的資訊來形成決策的方式，可使各節點的決策具有相當的一致性，隨著資料流逐漸上移，neighborhood 的組成逐漸改變，故亦不至出現瞬間壅塞的狀況。在共同除去重要性較低的 request 狀況下，大部分 minislot 可被用在重要的資料流上，並降低因爲過度預約而產生的 minislot 配置延遲。