混合光纖同軸網路上 MCNS 標準中

截斷二元指數後退演算法之研究

Study on Truncated Binary Exponential Back-off Algorithm for the MCNS Standard for Hybrid Fiber Coaxial Networks

朱國志　　戴義剛　　*李維聰　　詹寶珠
Kuo-Chi Chu　　Yi-Gang Tai　　Wei-Tsong Lee　　Pau-Choo Chung
國立成功大學電機工程系
Department of Electrical Engineering
National Cheng Kung University
Tainan, Taiwan, R.O.C
kuochi@vlan.ee.ncku.edu.tw

*逢甲大學資訊工程系
Department of Information Engineering
Feng Chia Universit
Taichung, Taiwan, R.O.C
wtlee@fcu.edu.tw

摘要

一個碰撞解決演算法的好壞影響整個通訊協定的整體效能。Multimedia Cable Network System(MCNS 是目前混合光纖同軸(Hybrid Fiber Coaxial; HFC)有線電視網路中最具影響力的一個標準。在 MCNS 標準中採用了截斷二元指數後退(Truncated Binary Exponential Back-off; TBEB)演算法來解決碰撞問題，這個演算法經驗證在網路負載(load)重時，效能便會急速的下降。針對此問題，MCNS 在頭端(Cable Modem Termination System : CMTS)中預留了控制參數，希望未來有人可以利用這些控制參數，來修改原先演算法中上述的缺點。因此在本篇論文中，我們將利用MCNS 標準中預留的控制參數，提出一個架設於 CMTS 的 DWS(Dynamic Windo Selection)演算法，經過我們的模擬驗證，我們發現搭 DWS 後的 TBE 演算法，在負載重時碰撞解決的能力將獲得大幅改善。

關鍵字：碰撞解決、多媒體纜線網路系統、混合光纖同軸網路、有線電視網路、寬頻存取

Abstract

The efficiency of a co llision resolution algorithm affects the overall performance of a communication protocol. Currently, Multimedia Cable Network System (MCNS) protocol is the most influential standard for hybrid fiber coaxial (HFC) bi-directional cable television network. In this standard, Truncated Binary Exponential Back -off (TBEB) algorithm is adapted to resolve collision. The performance of this algorithm will drop dramatically when the load is high. Therefore, in the MCNS standard some parameters are reserved for future use. They can be used to improve the performance of the original algorithm. In this paper, we are going to use these reserved parameters in the MCNS standard to propose an algorithm constructed on the CMTS, called DWS, Dynamic Window Selection. Through simulation, we find that TBEB algorithm, cooperated with DWS, can have greatly enhanced performance of collision resolution at high load.

Key Words : collision resolution 、MCNS 、Hybrid Fiber Coaxial Network 、CATV 、Broadband Access.

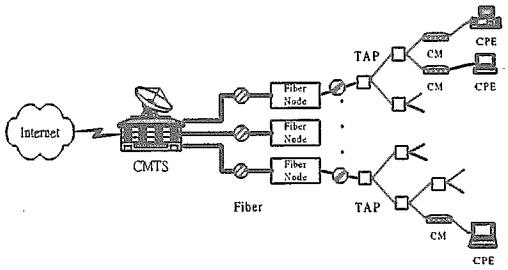
本論文研究為國科會補助經費之研究成果，
計畫編號；NSC88-2213-E-035-045

1. 簡介

近年來網路的快速興起，各項網路的服務不斷的推陳出新，造成上網的人數急遽增加，對於網路的頻寬需求也與日遽增。傳統家中電話網路經由數據機所能提供的頻寬，早已無法滿足大多數人的需求，此時具有高頻寬及高普及率 HFC 有線電視網路，便成為家庭中取代傳統電話網路對外的最佳方案之一。在過去一年，隨著法令的修正，國內家庭用戶架設寬頻 HFC 有線電視網路的用戶數量大增，再再的說明 HFC 有線電視網路將在下一世紀家庭對外網路中扮演重要的角色。

幾年前在 HFC 網路中，並沒有一個公認的標準存在，因此各家廠商都必須各自開發屬於自己的產品及通訊協定，因此造成各家的纜線數據機(Cable Modem)產品因規格不同而無法互連的窘境，為此各家廠商無不希望能有一公認的標準被及早推出，以供大家遵循。在大家對於一個標準的急迫需求下，1997 年 3 月一個由 Multimedia Cable Network System Partners Ltd.制訂的 MCNS 標準被公布，這個標準規格被稱為 Data Over Cable System Interface Specification (DOCSIS)[1]，這個規格並於 1998 年 3 月間通過 International Telecommunication Union (ITU)的認證，同時也獲得大多數廠家的支持，並於 1998 年底已有符合其標準規格的產品上市，MCNS 已成為目前 HFC 網路中最具影響力的一個標準。

因為整個 MCNS 標準是在急切需求中匆忙制訂的，所以整個標準所採用的各項規格多沿用既有的技術，以求穩定及能及早制訂完成。在處理碰撞解決問題這一方面，MCNS 所採用的是截斷二元指數後退(Truncated Binary Exponential Back-off ; TBEB)演算法[1]，這個演算法被提出已經有一段很長的時間，他具有容易實現及簡單易懂的優點，但是經過驗證他在網路負載量大時，會有效能急速下降的缺點存在[2]，有關這個演算法我們將在第二章為大家介紹。如圖一，以目前一個 HFC 網路的架構，每個光纖節點(Fiber node) 都被要求要能服務 500-2000 個用戶，在如此多的一個用戶數目下，網路很容易有負載重的情形出現，使得TBEB 這一個碰撞演算法常常不能有效的解決碰撞。



圖一 HFC 網路架構圖

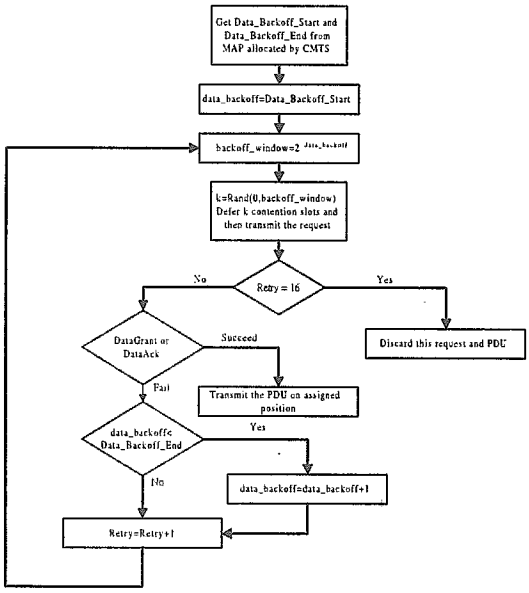
MCNS 在制訂標準的當時，就知道這個問題，但受限於早日完成標準制訂的壓力下，他們依然決定採用 TBEB 這一個演算法，但在標準中他們預留了參數，並建議大家可以經由調整這些參數，提升 TBEB 解決碰撞的能力。因此在本篇論文的第三章將提出一個新的演算法 DWS(Dynamic Window Selection)，這個演算法會根據網路的負載，經由 MCNS 上述所提供的參數，動態的調整用戶端的後退視窗(Back-off window)的範圍大小，以增加碰撞解決的機率 進而提升網路的效能。第四章我們將利用程式模擬，比較搭配我們所設計的 DWS 演算法後的 TBEB 演算法和原始 TBEB 演算法兩者之間，在解決碰撞時的效能優劣，藉此來印證我們的演算法將可以有效改善 TBEB 在負載重時，效能會急速下降的缺點。最後的第五章，我們將對這一篇論文做一個結論。

2. 截斷二元指數後退演算法

HFC 有線電視網路上行頻道是屬於共享的頻道，因此用戶要傳送資料時必須經由競爭以獲取頻寬，這時則需要一個碰撞解決演算法解決各用戶之間的競爭。MCNS 標準採用截斷二元指數後退(Truncated Binary Exponential Back-off ; TBEB)演算法來解決用戶彼此間的碰撞。MCNS 標準中的 TBEB 這個演算法和在乙太網路(Ethernet)中所使用的二元指數後退(Binary Exponential Back-off ; BEB)演算法[3][4]大致相同，只是針對 HFC 網路的特性略做了一些修改，以下我們就將對此演算法做一個

詳細說明。

圖二是 TBEB 的運作流程圖，平時用戶端的纜線數據機(Cable Modem; CM 會定期接收到由 HFC 網路頭端(Cable Modem Termination System ; CMTS)所傳送的 Data Back-off Start (S 及 Data Back-off End (E)這兩個參數。這兩個參數的作用，其中 S 是表示後退視窗(Back-off window)的起始值；E 則表示後退視窗(Back-off window)的最大值。在目前 MCNS 的標準中，這兩個參數 S 和 E 是一個由 CMTS 決定的固定值不會變動。



圖二 TBEB 運作流程圖

當用戶端有資料要送時，CM 就在 0 到 2 的後退視窗次方中隨機選擇一個數字 k（第一次傳送時，後退視窗等於 S），此後 CM 需等待 k 個競爭時槽通過後，就可以立刻在下一個競爭時槽到來時，傳送頻寬需求至 CMTS。如果傳送過程沒有發生碰撞的現象，CMTS 在收到 CM 端的頻寬需求後，就會送出 Data Grant 或 Data Ack 的訊息告知已接收到了 CM 端的頻寬需求，CM 端在收到這兩個訊號之一後，則結束這一次碰撞解決過程；反之，若 CM 的頻寬需求在傳送途中發生了碰撞，則 CM 端將接收不到 CMTS 的回應，此時若 CM 端的後退視窗值尚未超過 E 值，則後退視窗值加 1。之後則重複，隨機從 0 到 2 的後退視窗次方中選擇一數字的步驟，如此反覆下去直到傳送成功或連續傳送超過 16 次都不成功被強制放棄傳送為止。以下表一是 TBEB 的演算法。

```
Variables:
Data_Backoff_Start: determine the initial size of backoff window, assigned by CMTS
Data_Backoff_End: determine the maximum size of backoff window, assigned by CMTS
k: a randomly selected integer within the backoff window
Rand(x,y): a function that generates a random integer between x and y
success: to mark if the transmission is successful
retry: the number of retries
backoff_window: the size of backoff window
data_backoff: 2 to the data_backoff'th power (2^data_backoff) is the size of backoff window
Data_Grant/Data_Ack: sent by CMTS to inform stations that a request is received

Algorithm:
TBEB()
{
        retry = 0;
        success = FALSE;
        Get Data_Backoff_Start and Data_Backoff_End from MAP allocated by CMTS;
        data_backoff = Data_Backoff_Start;
        while ( success == FALSE )
        {
                Set backoff_window to 2 to the data_backoff'th power (2^data_backoff);
                k = Rand( 0, backoff_window );
                Defer k contention slots and then transmit the request;
                if ( retry == 16 )
                {
                        Discard the request and PDU;
                        Break;
                }
                else
                {
                        if ( Data_Grant or Data_Ack is received )
                        {
                                Transmit the PDU on assigned position;
                                success = TRUE;
                        }
                        else
                        {
                                if ( data_backoff < Data_Backoff_End )
                                        data_backoff = data_backoff + 1;
                                retry = retry + 1;
                        }
                }
        }
}
```

表一 TBEB 演算法

接下來，我們舉一個例子來做說明，假設 CM 端在要傳送頻寬需求時，接收到 CMTS 傳送下來的 S,E 值分別為 2 和 5，那 CM 端的後退視窗起始值便為 2，最大值就為 5。因此 CM 端第一次傳送時，就必須在 0 到 2 的 2 次方 4 中，選擇一個數字，假設 CM 端選擇的是 3，那 CM 端就需等待上行頻道中 3 個競爭時槽通過後，第 4 個競爭時槽到來時則就立刻傳送。如果未發生碰撞，CM 端會收到 CMTS 的回應後結束；如發生碰撞，CM 就會在不超過 E 值 5 的前提下，自動增加後退視窗的大小 1，故此時 CM 端後退視窗則變為 3。之後演算法重新要 CM 從 0 到 2 的後退視窗次方中，隨機選擇一數字，以目前的例子而言，後退視窗是 3，因此 CM 要從 0 到 2 的 3 次方 9 中隨機選擇一個數字 k，並等待 k 個競爭時槽後才可傳送，如此重複直到傳送成功或傳送 16 次都失敗為止。在這個例子中，後退視窗因受限於 E 值為 5 的因素，所以就算碰撞次數再多，後退視窗的最大只能是 5，換句話說，CM 可以選擇最大的等待競爭時槽數範圍是介於 0 到 2 的

5 次方 32 之間。

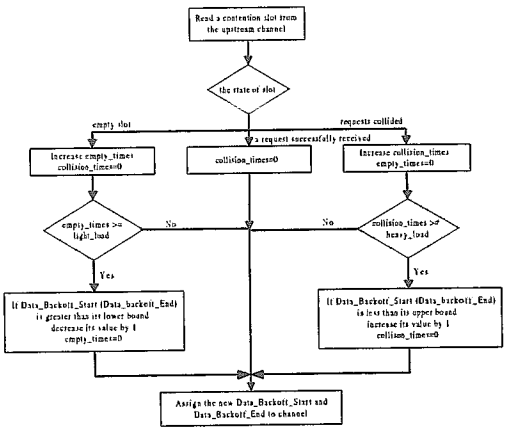
我們現在把焦點放在 S 及 E 這兩個參數上，這兩個參數決定了後退視窗的大小範圍，直接影響著解決碰撞的效能[5]。換句話說，當我們把 S,E 定的較小時，CM 端在負載輕時，從提出頻寬需求到頭端成功接收到要求並回應的競爭延遲時間會很小，但是一旦負載重時，用戶端提出頻寬申請被頭端接受的整體競爭成功率會降的很低；反之，我們把 S,E 定的較大時，CM 端在負載輕時，競爭延遲時間會較大，但在負載重時，網路整體的競爭成功率卻不會降的很多。所以最好的作法，就是能有一個動態的機制，可以偵測網路的負載量，在負載輕的時候，把 S,E 定的小一點，負載增重的時候，則把 S,E 加大，如此將可以兼具在負載輕時競爭延遲時間小，負載重時競爭成功率高的優點。

MCNS 標準中，即建議 CMTS 端如能動態根據網路負載調整 S,E 值，將可以使得整個 TBEB 演算法有較好解決碰撞的能力，但在其標準中則未對 CMTS 端該如何動態調整 S,E 值做一規範，所以針對 MCNS 標準中此一未規範處，我們提出了一個架設於 CMTS 的動態選擇視窗(Dynamic Window Selection; DWS)演算法，他將能根據網路的負載輕重，動態調整 S,E 值，使 TBEB 具有較好解決碰撞的能力，下一章我們將詳細介紹著一個演算法。

3. 動態選擇視窗演算法

動態選擇視窗(Dynamic Window Selection; DWS)演算法是運作在 CMTS 上的演算法，他最主要的目的就是動態的提供 S,E 值給 CM 端的 TBEB 演算法參考，使得 TBEB 演算法得以根據網路的負載量，機動地調整出合適大小的後退視窗。

圖三 是 DWS 演算法的運作流程圖。DWS 決定一個上行頻道是否擁塞的方法，最主要是根據競爭時槽的使用狀況來做判斷，如果 CMTS 收到了一連串發生碰撞的競爭時槽，那就表示目前



圖三 DWS 演算法流程圖

這個網路正處於重負載的狀況；反之，若 CMTS 收到了一連串沒人使用的競爭時槽，那就表示目前這個網路正處於輕負載的狀況。因此在 DWS 演算法內，我們使用了兩個變數，來記載目前已有連續幾個空或發生碰撞的競爭時槽，其中 empty_time 這個變數是記載連續有多少個空的競爭時槽送達頭端，另一個變數 collision_tim e 則是記載了有多少個連續碰撞的競爭時槽送抵 CMTS，因此當有一個空的競爭時槽被 CMTS 收到時，empty_time 則加 1，collision_ti m 則被清為 0；反之，如有一個發生碰撞的競爭時槽被 CMTS 收到時，則 collision_times 加 1，empty_tim 則被清為 0。此外，當 CMTS 收到一個中途未發生碰撞成功送達要求的競爭時槽時，此時 collison_time 將被設為 0，empty_times 則是維持不變。

之後 CMTS 就會判斷 empty_time 及 collision_ti m e是否已經累加到達我們所定義的輕負載及重負載的值，如果未到達的話則不做處理，已到達的話則再判斷目前的 S,E 值是否已經是我們所定義的上下限。如果 S,E 值已經到達上限或下限的話，那就維持已到達界限的值不做更動；如未到達我們所定義的上下限時，在輕負載時我們就將 S,E 值分別減 1，重負載時則加 1。在這邊我們需要對 S,E 的值分別設上、下限的主要目的，在下限部分是不希望 S,E 這兩個值降的太低，避免日後負載量突然急速上升時，很容易會有一連串的碰撞產生；至於設上限的原因，則是不希望 S,E 兩者升的太高，使得系統從重負載漸漸變成輕負載時，系統會沒有辦法即時反應過來，快速縮小後退視窗的範圍，導致雖然網路負載輕但是資料的競爭延遲時間卻還是很大的不正常現象發生。

最後，我們再定期性把經過 DWS 演算法所計算過的 S,E 值，經過下行頻道分送給各 CM 端，

讓 CM 端的 TBEB 演算法可以真正根據網路負載輕重來加大或減小後退視窗的範圍。以下表二是 DWS 的演算法：

```
At CMTS

Variables and Parameters:
Data_Backoff_Start: initial backoff window size assigned to stations
Data_Backoff_End: maximum backoff window size assigned to stations
DBSUB: upper bound of Data_Backoff_Start
DBSLB: lower bound of Data_Backoff_Start
DBEUB: upper bound of Data_Backoff_End
DBELB: lower bound of Data_Backoff_End
empty_times: the number of contention slots that are empty consecutively
collision_times: the number of consecutive slots where collisions occur
light_load: if the number of consecutively empty slots exceeds this threshold, the
channel goes into the state of light load
heavy_load: if the number of consecutively collided slots exceeds this threshold, the
channel goes into the state of heavy load

Algorithm:
while ( Read a contention slot from the upstream channel )
{
    switch ( the state of slot )
    {
        case "empty slot":
                empty_times = empty_times + 1;
                collision_times = 0;
                break;
        case "requests collided":
                collision_times = collision_times + 1;
                empty_times = 0;
                break;
        case "a request successfully received":
                collision_times=0;
                break;
    }
    if(empty_times>=light_load)
    {
      if(Data_Backoff_Start>DBSLB)
      {
        Data_Backoff_Start =Data_Backoff_Start-1;
      }
      if(Data_Backoff_End>DBELB)
      {
        Data_Backoff_End = Data_Backoff_End-1;
      }
      empty_times=0;
    }
    if(collision_times>=heavy_load)
    {
      if(Data_Backoff_Start <DBSUB)
      {
        Data_Backoff_Start =Data_Backoff_Start+1;
      }
      if(Data_Backoff_End<DBEUB)
      {
        Data_Backoff_End = Data_Backoff_End+1;
      }
      collision_times=0;
    }
        Assign the new Data_Backoff_Start and Data_Backoff_End to channel;
}
```

表二 DWS 演算法

接下來我們舉一個例子來做說明，變數部分則根據上面演算法的定義，初值部分假設剛開始的 Data_Backoff_Start=2, Data_Backoff_End=5, DBSUB=5, DBSLB=2, DBEUB=10, DBELB=4, Light_load=3, Heavy_load=2, empty_times=0, collision_times=0。競爭時槽的狀況如下，其中 C 表示此時槽發生碰撞，E 表示此時槽是空的，S 則表示此時槽只有一個人傳送，就是等於傳送成功。以下表三是 DWS 演算法的運作過程說明。

Time	Status	Operation Result
1	C	empty_times=0, collision_times=1
2	E	empty_times=1, collision_times=0
3	S	collision_times=0
4	E	empty_times=2, collision_times=0
5	E	collision_times=0 Data_Backoff_Start=2, Data_Backoff_End=4 empty_times=0
6	E	empty_times=1, collision_times=0
7	C	empty_times=0, collision_times=1
8	S	collision_times=0
9	C	empty_times=0, collision_times=1
10	C	empty_times=0 Data_Backoff_Start=3, Data_Backoff_End=5 collision_times=0

表三 DWS 運作過程

4. 模擬與分析

在這一章中，我們將利用程式來做模擬。針對 TBEB 演算法在附加 DWS 演算法後，對於解決碰撞的效能是否會有所提升，來做一個詳細的探討。在這邊的模擬我們利用兩個實驗，來比較有無使用 DWS 演算法對 TBEB 演算法的影響。下面的表四是整個模擬過程中 HFC 網路所使用的參數。

Common Parameters	
Parameters	Values
N (Number of active stations)	100
Number of contention slot per frame	3
Retransmission time	25 mini-slots
Time length of a mini-slot	16 micro second
Round-trip delay	400 micro second
Mean packet size	5 mini-slots
Traffic model	Poisson process Mean inter-arrival time = (N*Mean packet size)/Load
DWS Parameters	
Parameters	Values
Initial Data_Backoff_Start	2
Initial Data_Backoff_End	5
DBSUB (Upper bound of Data_Backiff_Start)	5
DBSLB (Lower bound of Data_Backiff_Start)	2
DBEUB (Upper bound of Data_Backiff_End)	10
DBELB (Lower bound of Data_Backiff_End)	4
Light_load (Number of consecutively empty slots)	9
Heavy_load (Number of consecutively collided slots)	2

表四 模擬參數

4.1 實驗項目

在我們底下所做的實驗中，我們使用了如表五的三組實驗組。其中第一組 TBEB(0,10)的設定，是和目前乙太網路中 BEB 演算法的設定相同。底下我們就對我們所做的兩組實驗內容作一個說明。

Case	Description

TBEB (0,10)	S=0, E=10 ; Only use TBEB Algorithm
TBEB (2,5)	S=2, E=5 ; Only use TBEB Algorithm
DWS	Use TBEB and DWS Algorithm

表五 實驗分組表

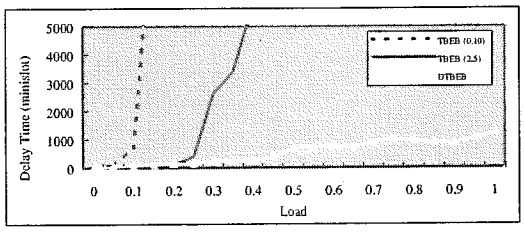
- 實驗一：在這個實驗中將探討 TBEB 演算法搭配使用 DWS 演算法前後的效能差異。在此我們模擬在不同負載量時，上述三組實驗組競爭延遲時間及競爭成功率做比較。
- 實驗二：根據先前研究[6]，要在網路上傳遞語音資料，必須符合只能有百分之 1 到 2 的封包遺失和延遲時間不能大於 10 到 20 個 ms。這裡我們將在三個實驗組中模擬傳送語音資料，並計算這三個實驗組於不同負載量下，在 10ms 及 20ms 內能完成傳送的成功率有多少，藉此分析語音資料在 HFC 網路中傳送的可行性。

4.2 實驗結果

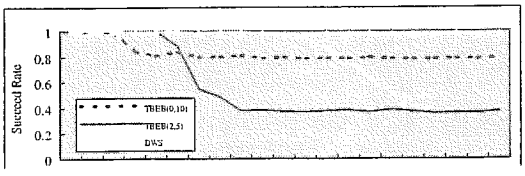
底下將說明我們在上述實驗中所得到的結果，並對該結果作一個分析說明。

- 實驗一：圖四是在實驗一中，三組實驗組在不同負載量下，所分別得得到的封包延遲時間。我們由此圖可以發現到下列幾點。

1. TBEB(0,10)及 TBEB(2,5)的實驗組中，隨著網路負載的上升，碰撞也跟著增多，因此後退視窗也會跟著急速上升。但在 TBEB(2,5)實驗組中，後退視窗最大被限制在 5，因此競爭延遲時間在初期還可維持在一範圍內，不會似 TBEB(0,10)快速上升，TBEB(2,5 直到負載上升至後退視窗 5 已經無法有效解決碰撞時，此時競爭延遲時間也就跟著大幅上升。



圖四 實驗一競爭延遲時間



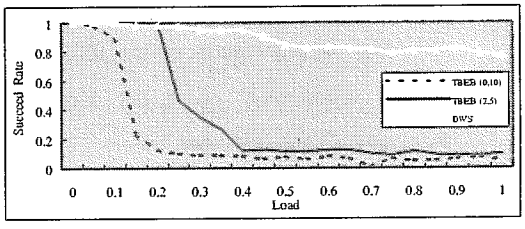
圖五 實驗一競爭成功率

2. 至於搭配了 DWS 演算法的這一個實驗組，由於 S,E 值可隨著負載輕重動態變動，因此我們可以發現在競爭延遲延遲這一個項目，他比未使用 DWS 演算法的實驗組好了許多。

圖五，則是三個實驗組在實驗一中不同負載下的競爭成功率，我們可以經由此圖發現以下幾點。

1. TBEB(0,10)因為後退視窗的最大範圍為 10，所以有較充足的空間來解決碰撞，因此競爭成功率較最大值只有 5 的 TBEB(0,5)好了許多。
2. 這邊我們發現搭配 DWS 演算法的 TBEB 演算法在競爭成功率上的表現勝過未使用的 TBEB 演算法許多。至於 DWS 這一組的表現會比 TBEB(0,10)好的原因，是因為 TBEB(0,10)剛開始遇到碰撞後退視窗都要從 0 開始遞增，因此在負載重的時候，會比可動態調整起始值的 DWS，容易發生重傳超過 16 次封包被強制丟棄的現象。

- 實驗二：圖六及圖七是在實驗二中，三組實驗組在不同負載量下 10ms 及 20ms 內傳送語音資料的成功率。由這兩個圖我們可以明顯的看出，原本未搭配 DWS 演算法的 TBEB 演算法只要負載量稍微一高，就沒有辦法傳



圖六 實驗二中 10ms 內傳送成功率

圖七 實驗二中 20ms 內傳送成功率

送語音的資料。反觀搭配 DWS 演算法後，這種現象則完全改觀，傳送語音資料能力有明顯的提升。

5. 結論

在這篇論文中，我們首先為大家回顧了目前 HFC 有線電視網路上，最具影響力的一個標準 MCNS。其次我們針對這個標準在解決碰撞問題上，所採取的 TBEB 演算法有一個深入的探討。隨後我們提出了一個新的演算法 DWS，這是一個架設於 CMTS 上的演算法，他可以根據網路的現在使用情況，提供資訊給在 CM 端的 TBEB 演算法，使其能夠調整出較適合於目前網路負載量的後退視窗範圍。由於 DWS 這個演算法是利用 MCNS 原先預留的參數來傳遞資料，所以在架設及使用 DWS 演算法時，CM 端不需要做任何的更動，CMTS 端也只要把此演算法直接加入原先運作流程中即可。

最後我們設計了兩個實驗來判別，DWS 演算法對 TBEB 演算法的影響。經過模擬驗證，我們可以發覺加上 DWS 演算法運作的 TBEB 演算法，不論在競爭延遲時間及競爭成功率上的表現都遠勝過於單獨運作的 TBEB 演算法，同時也大幅提升了 HFC 網路直接傳遞語音資料的能力，增加了 MCNS 標準的整體效能。

參考資料

[1] Cable Television Laboratories, Inc., Data-Over-Cable Service Interface Specifications – Radio Frequency Interface Specification, Mar. 1999.

[2] Kennet J.Christensen, 'Performance Evaluation of the Binary Logarithmic Arbitration Method (BLAM)," In proceedings of the IEEE 21 st Annual Conference on Local Computer Networks, pp.396-403, 1996.

[3] 黃能富，區域網路與高速網路，維科出版社，八十七年六月。

[4] 楊鴻岳，「MCNS MAC 通訊協定規格[下]」，經濟部中央標準局，工業財產權與標準，八十七年六月。

[5] Ying-Dar Lin, Chen-Yu Huang and Wei-Ming Yin, "Allocation and Scheduling Algorithm for IEEE 802.14 and MCNS in Hybrid Fiber Coaxial Networks," IEEE Transactions on Broadcasting, Vol. 44 ,No 4, pp. 427 -435, DEC. 1998

[6] Gonsalves, T., andTobagi, F. "Comparative Performance of Voice/Data Local Area Networks," IEEE Journal on Selected Areas in Communication, Vol. 7, No. 5, pp. 657-669, June 1989.

[7] N. Golmie, F. Moiveaux and D. Su, " Comparison of MAC Protocols for Hybrid Fiber/Coax Networks: IEEE 802.14 vs. MCNS," In proceedings of the 1999 IEEE International Conference, Vol.1 ,pp.266-272, June 1999.