802.11 Ad Hoc 網路下競爭窗口選取機制與 TCP 效能之研究

余心淳
東海大學資訊管理學系
hsyu@thu.edu.tw

林志驊
大葉大學資訊工程學系
r9106027@mail.dyu.edu.tw

摘要

無線網路因為其便利性和靈活性，所以近年來廣泛的被建置與整合在各式的網路環境中，以便於提供更多元的應用與服務。IEEE 802.11 媒介擷取控制（MAC）協定原先是以無線區域網路中的 infrastructure 模式為基礎來設計，當運用在無線多點 ad hoc 網路模式下，TCP 吞吐量（throughput）便會存在著不穩定（instability）[8]和不公平（unfairness）[5]的情形，進而降低網路的效能。本文的研究方向是藉由相關文獻的探討找出問題的癥結，並以修改目前 802.11 標準中分散協調式功能（DCF）機制下競爭窗口（contention window）的選取規則來提升網路 TCP 的吞吐量。在本文中藉由模擬的數據結果，顯示所提出的機制在多種網路拓撲下 TCP 流量改善的情形與效能的比較。

關鍵詞：IEEE 802. 11、ad hoc、分散協調式功能、競爭窗口、TCP 吞吐量。

一、簡介

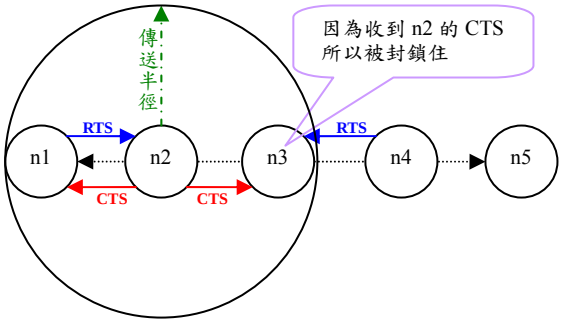
由於無線 ad hoc 網路有著組織靈活性和移動便利性的優點，所以近年來廣泛的被建置與整合在各式的網路環境中，以提供更多元的應用與服務。以 IEEE 802.11 標準[3]為主的無線 ad hoc 網路在 MAC 層提供二種傳輸媒介擷取技術：2-way 基本存取(basic access)與 4-way RTS/CTS 存取。其中 4-way RTS/CTS 存取又稱為虛擬載波偵測（virtual carrier sensing）的運作模式，傳送端與接收端藉由交換 RTS 和 CTS 控制訊框來設定或是更新鄰近節點上的 Network Allocation Vector（NAV），並延緩這些節點資料的傳輸，以便取得頻道的使用權。這種運作機制非但可減少碰撞的發生，也可解決遮蔽節點（hidden node）問題的發生。另一方面運用 DCF 機制，藉由一個二次方指數倒退（binary exponential back-off）演算法隨機選取倒數時間的方法，確保每一個傳輸節點可以公平的搶到頻道，也可以減少在同一頻道（channel）內資料流發生碰撞的情形。但也因為這個 RTS/CTS 存取機制的過度保護，所以暴露節點（exposed node）的問題仍無法有效的解決。至於 2-way 基本存取的方式，非本文所欲探討，因此在此不做詳細述說。

IEEE 802.11 MAC 協定使用 RTS 控制訊框和 CTS 控制訊框來解決隱藏節點（hidden node）的問題，但卻引發暴露節點問題的產生。因為無線網路裡存在著暴露節點的現象，以及 802.11 標準中 DCF 協定傳送次數的限制（RTS 傳送次數最大值為七次，若在七次內皆未收到 CTS 控制訊框的回應，則判斷路徑已經失連，並將佇列裡封包丟棄），若加上不適當的視窗選取規則，將導致 TCP 吞吐量易產生不穩定的問題，進而降低網路的流量。在本文第二部份的 DCF 重傳機制的設計裡，我們將探討適當的改變 DCF 機制裡的選取規則，藉此可減少發生路徑失連的機會，以及增加每一個欲傳送資料的節點搶到頻道的機會。

對於改善上述問題的相關研究可以從不同的技術方向來探討。例如於[8][9]文中提及修改 TCP 擁塞窗口(congestion window)，藉由縮小 TCP 擁塞窗口的大小來改善 TCP 吞吐量不穩定的情形。文獻[6]中亦提出一個符合每一個資料片段（fragment）的固定倒數時間的想法，藉此提升 TCP 吞吐量的效能。[1]和[10]皆提出加大競爭窗口大小的方法，文獻[1]將網路上所有節點分為兩類，當節點成功傳送資料片段後，下一次傳送資料片段前加大最小競爭窗口（CW_{min}），反之則遵循 802.11 標準，藉此避免媒介永遠以最小的倒數時間持續佔用頻道，此外另配合增加 RTS 控制訊框重傳的次數（從最大重傳次數七次改為九至十次）來改善吞吐量的效能。 [10] 則是利用訊號對雜訊比（signal to noise ratio）來判斷周遭網路情形，若是繁忙，則加大競爭窗口和縮小 TCP 窗口。

在文獻[1]以及[9]的研究中發現到 TCP 吞吐量產生不穩定以及不公平的現象，是因為暴露節點和 802.11 DCF 協定中限定 RTS 訊框的重送次數所導致的問題。可藉由圖一來解釋這個情形，假設存在一個 ad hoc 網路使用 DSR 路由協定的線性拓撲，其中存在五個節點，n1、n2、n3、n4 和 n5，如圖一所示。FTP 資料流分別從 n1 傳至 n5 以及由 n5 傳至 n1，其中 n2、n3、n4 只有進行封包轉送（packet forwarding）的動作：

一開始，當 n1 與 n2 互傳資料時，n3 會被 n2 傳出的 CTS 控制訊框給鎖住，可是由於 n4 不知道 n3 已被封鎖住，所以 n4 傳送 RTS 控制訊框給 n3



圖一 線性雙向 FTP 資料流網路拓撲圖

時，雖然 n3 已準確收到 RTS 控制訊框，但因為 n3 被 n2 封鎖住所以並不會回應 CTS 控制訊框給 n4。等到 n4 的 RTS 控制訊框重傳次數超過七次時，n3 有可能還未能回傳 CTS 給 n4。因此 n4 便會判斷路徑已經失連，並回傳一個路徑失連的訊息給來源端 n5，同時將在節點佇列裡的封包丟棄。因為在這段時間裡，要重新尋找路徑，所有封包皆無法傳送[2][4]，導致於整體的 TCP 吞吐量降低。這便是 TCP 吞吐量不穩定現象產生的原因，至於 TCP 吞吐量不公平的產生情形亦類似於此。在本文第二部份則提出一個倒數時間的探討以及選取。第三部份提出模擬相關的系統參數。第四部份會進行模擬結果的數據分析以及討論。最後的結論會在第五部份詳述，且細述本文所提方法改善的效能。

二、DCF 重傳機制的設計

由文獻探討中可明確的知道造成 TCP 吞吐量不穩定的現象是因為傳送端傳送 RTS 控制訊框給接收端時，接收端被鄰近正在傳送資料的節點給封鎖住，不能回傳 CTS 控制訊框。再加上 IEEE 802.11 標準中 DCF 協定的競爭倒數時間機制的設計，所以當傳送端經過七次 RTS 控制訊框傳送後，仍未收到接收端所回應的 CTS 控制訊框時，即誤認為接收端已經離開此傳送路徑，於是回傳給上一個來源端告知路徑失連，並請求重新尋找且建立新的路徑，同時丟棄放在佇列裡的所有封包，進而造成 TCP 吞吐量下降。本研究針對 TCP 吞吐量不穩定的現象，設計一個 DCF 競爭窗口選取規則給競爭不到頻道的節點使用。當節點第一次傳送 RTS 控制訊框且沒有收到相對應的 CTS 控制訊框時，傳送端節點便假設接收端節點可能身為暴露節點而無法回應，於是當要傳送第二次 RTS 控制訊框之前，傳送端節點便改變其競爭窗口的大小。

將上圖一 n1 及 n2 正在傳送資料的資料流，以及其餘節點的運作情形，繪出如圖二的時序圖。其中一個資料的傳送時間為 $DIFS + CW_k + RTS + CTS + DATA + 2\times SIFS$，而重複傳送 m 次 RTS 控制訊框所需時間為 $m\times(DIFS + CW_k + RTS + \tau)$，而參數 τ 為等待接收一個 CTS 控制訊框所需要的全部時間（CTS_Timeout）。T_{data} 表示圖二中 n1 與 n2 之間扣除掉 RTS/CTS 控制訊框交換以外的資料（包含回應）所需要的傳送時間，在此 $T_{data} = Data + SIFS$，而 T_{idle} 則定義為 $DIFS + CW_k$ 的時間。如圖二所示，若圖一的節點 n4 欲在 RTS 重傳七次內收到圖一的 n3 的 CTS，則圖二的 n4 需要在 T_{idle} 的時間內傳送出 RTS 且收到圖一的 n3 的 CTS，換言之便是圖二的 n4 需要在 T_{idle} 的時間內剛好倒數完競爭窗口的隨機倒數時間且發送 RTS 並接收到圖一 n3 的 CTS 搶到頻道。假設 $\tau \cong SIFS + CTS$ 因此可以得出公式如(1)所示：

$$ST + m\times(DIFS + CW_k + RTS + \tau) - (DIFS + CW_k + RTS + SIFS + CTS + T_{data}) < T_{idle} \quad (1)$$

化簡(1)式得到式(2)

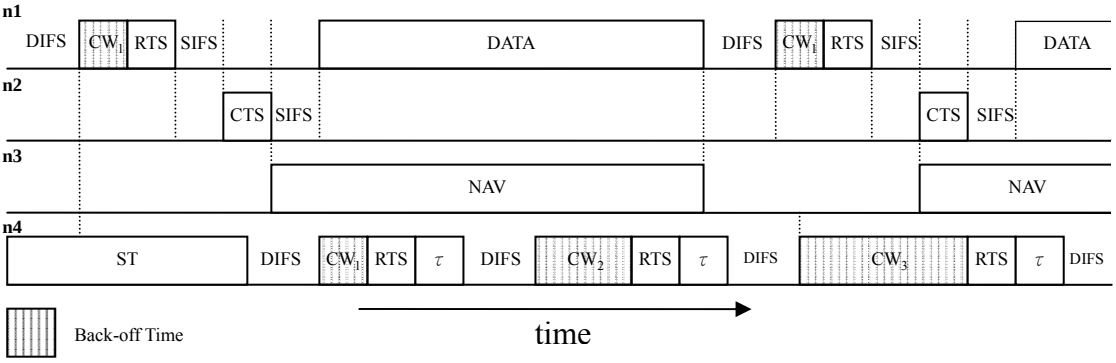
$$ST + (m-1)\times(DIFS + CW_k + RTS + \tau) - T_{data} < T_{idle} \quad (2)$$

因為我們無法知道 n4 何時會傳送 RTS 控制訊框給 n3，所以我們先加上一個隨機的時間 ST，其中 ST 代表經過 ST 的時間後 n4 開始做資料傳送。亦可藉由此時間的設立，進而證明下列公式的正確性。CW_k 為第 k 次隨機選取競爭窗口的倒數計數次數上限。依照 802.11 DCF 協定中 k 的最大值為七。$CW_k = 2^{k-1}\times(CW_{k-1}+1)-1$，$k\in(1,\ 7)$，其中 $CW_0 = CW_1 = 31 = CW_{min}$，同時 $CW_6 = CW_7 = 1023 = CW_{max}$。

將上述幾個式子整理後，發現 n4 在下列式(3)的區間裡發送 RTS 控制訊框可以成功的搶到頻道（亦要在區間內收到 CTS 才可），其結果如下：

$$T_{data} < ST + (m-1)\times(DIFS + CW_k + RTS + \tau) < T_{data} + T_{idle} \quad (3)$$

當資料片段大小為 1524 位元組時，T_{data} 可以轉換成相當約略於 6 毫秒的時間長度 [6]。故 $ST + (m-1)\times DIFS + CW_k + RTS + \tau$ 的隨機倒數時間至少必須大於 6 毫秒。因為 ST 的時間無法明確訂立（不知道節點何時進行傳送的動作），所以先給 ST 一個平均的時間 3 毫秒（每一次循環為 6 毫秒）。相對於現行 DCF 競爭窗口的選取規則，競爭窗口選擇的隨機倒數時間等於或是大於 3 毫秒為 255（第四次選取機會，如表一所示。其中一時槽為 0.02 毫秒，0.02×255=5.1 毫秒），競爭窗口為 255 選中的隨機倒數時間等於或是大於 3 毫秒的機率為 106/255（第 150 格至第 255 格，總共 106 格）。但是考慮當 ST 時間過小時，倒數時間要加大。所以在做模擬前訂立兩個基礎點，分別做 255 以及 511 的效能評估。最後發現到必須改以選取競爭窗口為 511 作基底，來配合 ST 時間的不確定因



圖二 多點傳送以及競爭時序圖

素。至於第二次以後的競爭窗口增長狀態，則保留 802.11 DCF 協定裡的規範，以符合本文提出方法修改原協定最少為目標的原則。根據上述的結論，本研究設計的競爭窗口選取規則比較表如下表一，節點會依據是否有收到 CTS 控制訊框來決定下次選取的競爭窗口大小。若有收到 CTS 控制訊框的回覆，則遵照原先 802.11 MAC 的協定，進行資料傳送。若無收到 CTS 控制訊框則遵照本文所設計的競爭窗口選取規則流程。本文與李雲[1]所提的方法相似點僅在於競爭窗口大小設定上面同以 511 為探討基礎，但是對於變更倒數時間的媒介節點卻為相異。表一是 IEEE 802.11 MAC 協定下現行 DCF 的競爭窗口選取規則和本文所提出的競爭窗口選取規則比較表。至於在 RTS 最大重傳次數上方面，本文分別取七次和十次兩種方法組合，藉此比較整體效能上的差異。

三、系統模擬環境

系統模擬相關參數設定如表二，模擬環境假設為一非常理想之網路環境狀態。其中飽和吞吐量（Saturation Throughput）的狀態為網路上的節點其傳送佇列皆為滿載的狀態。

模擬項目解釋：依據現行 IEEE 802.11 標準以及相關文獻的方法規格，定出如表三的模擬協定列表與說明。

其中組別 2 為我們所提出的另一種方法，讓身為暴露節點的節點進行額外的動作。誠如上述，造成 TCP 吞吐量不穩定的關鍵是身為暴露節點的節點，雖然收到 RTS 控制訊框，但是因為本身被鎖住，所以不能回傳 CTS 控制訊框，因此我們加上一個機制讓暴露節點收到 RTS 控制訊框時做計數（RTScounter）的動作，當計數器到達所設定的值（N）時（RTScounter = N），在本身的 NAV = 0，發送一個保留頻道的訊號（RTSreserve）。重傳多次 RTS 節點收到訊號時，立即變更倒數的狀態改用 DIFS + CW_{min} 的時間重新傳送 RTS 控制訊框。但是其 RTS 控制訊框的重送計數次數未作初始化的動作，換言之仍有可能在競爭失敗後很快便發生

表一 不同的 DCF 競爭窗口選取規則比較表

	802.11	提出的選取規則
CW_1	31	31
CW_2	63	511
CW_3	127	1023
CW_4	255	1023
CW_5	511	1023
CW_6	1023	1023
CW_7	1023	1023

表二 模擬環境參數

模擬軟體	SNT QualNet 3.7[7]
環境拓撲	五點線性直線、七點 Y 字、九點十字、二十二點井字、四十九點網狀
節點距離	兩點之間單位距離 300 公尺
模擬時間	120 秒
TCP 版本	TCP NEW RENO
資料流型態	FTP
封包數目	10^7 個（飽和吞吐量）
資料大小	1500 位元組/封包

表三　模擬協定列表與說明

協定組別	說明
組別 1	表示遵循原 IEEE 802.11 MAC 協定的機制 ，(其七次的重傳規則為 31、63、127、255、511、1023、1023)
組別 2	本研究另外提出一個方法來改善整體效能
組別 3	當節點成功傳送完資料後，最小競爭窗口改為 255 或是 511，且重傳最大次數由七次改為十次（而十次的重傳規則以 255 為例為 255、511、1023、1023、1023、1023、1023、1023、1023、1023），根據文獻[1]
組別 4	當第二次重傳 RTS 時競爭窗口大小改從 511 開始，當最大重傳次數為十次時，第八至十次的競爭窗口皆為 1023。

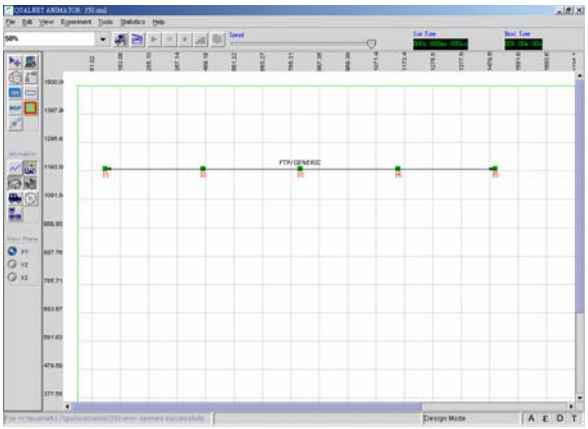
路徑失連的情況，使得整體網路競爭更加公平化。

組別 4 是本文所提出的方法，在模擬數據結果裡分別以 RTS retransmit limits = 7 和 10 來代表 RTS 最大重傳次數設定值為七次以及十次。

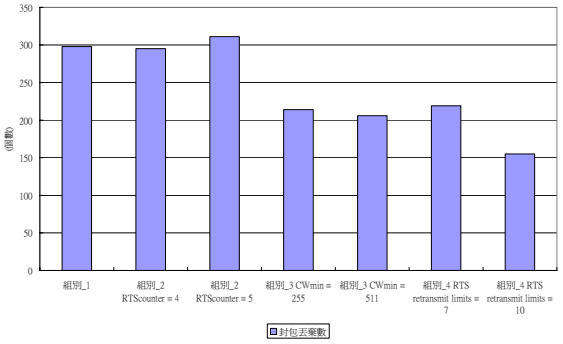
綜合上述，TCP 吞吐量的不穩定現象是因為在 802.11 DCF 協定下，傳送端傳送七次的 RTS 控制訊框後皆沒收到接收端回覆的 CTS 控制訊框，因此誤以為路徑失連，進而丟棄在佇列裡的封包，停止上溯所有路線的傳送動作，導致吞吐量下降。因此，觀察其中所丟棄的封包數目較少，且 TCP 吞吐量較好，將其視為擁有較為優異效能表現的方法。

四、模擬結果與分析

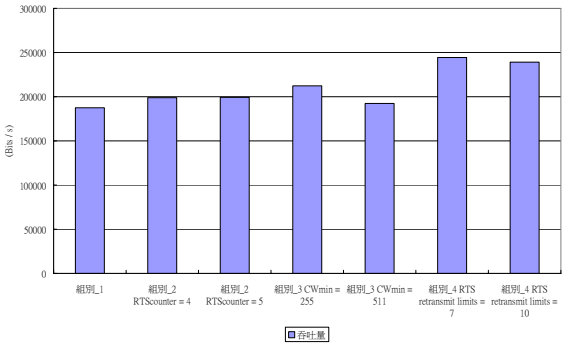
首先探討線性拓撲的情形，其中拓撲圖如圖三所示，模擬結果如圖四和圖五所示。由圖三可以發現到封包丟棄數目最少為組別 4 RTS retransmit limits = 10，最多為組別 2 的 RTScounter = 4。如文前所述，封包丟棄是因為節點的 RTS 控制訊框傳送超過七次後，皆未收到 CTS 控制訊框所以判斷路徑失連，丟棄佇列裡的封包數，導致 TCP 吞吐量效能不佳。但是封包丟棄的數目並不代表路徑失連的次數，因為無法得知當時在佇列裡的封包數量，所以每次丟棄的數目皆不同，因此會產生數目上總和的差異。因此本研究將封包丟棄數目當作觀察數據資料的輔助。在正常情形下，封包丟棄數目和路徑失連的次數是成正比。雖然在圖三裡，封包丟棄數目最少的是組別 4 RTS retransmit limits = 10，直覺性的想法會認為吞吐量效能最好應該為組別 4 RTS retransmit limits = 10，但是觀看圖五的結果，可以發現到，吞吐量最高的組別為組別 4 RTS retransmit limits = 7。



圖三 線性網路拓撲圖



圖四 線性網路拓撲圖_封包丟棄

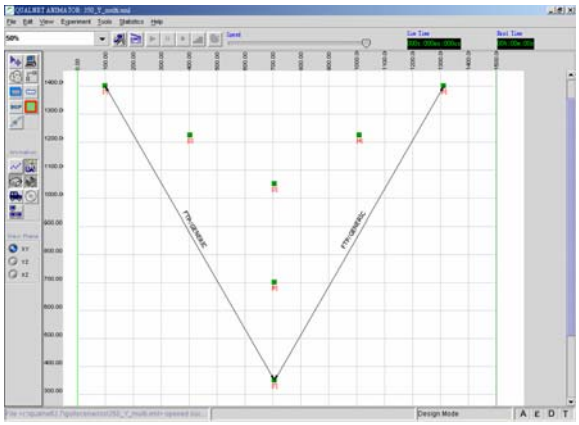


圖五 線性網路拓撲圖_吞吐量

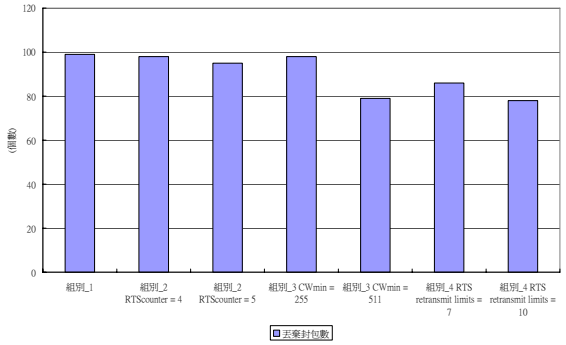
誠如上述，封包丟棄的數目可做為一個事前觀察的參考。圖四中封包丟棄數較少的兩組分別為組別 3 和組別 4，接下來觀察其吞吐量，兩者亦皆擁有不錯的效能結果。相對於組別 1 的模擬結果，組別 4 RTS retransmit limits = 7 和 RTS retransmit limits = 10 的吞吐量分別為組別 1 的的 1.304 和 1.276 倍。因此在線性拓撲網路圖裡，組別 4（本

文所提的方法）擁有不錯的效能結果。

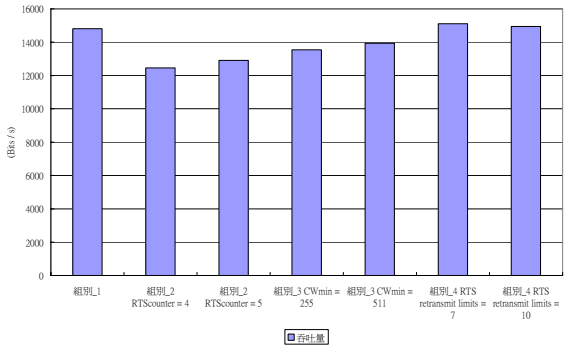
圖六是 Y 字形網路拓撲，圖七及圖八分別為 Y 字形拓撲下的封包丟棄以及吞吐量結果。由圖八所示，組別 4 RTS retransmit limits = 7 其吞吐量為最高。在圖七中，組別 4 RTS retransmit limits = 10 時，封包丟棄數目為四組模擬組別中最少的。在各種數據結果比例的比較下，組別 2 以及組別 3 在吞吐量上的表現並不突出，其產生的原因應該是由於網路拓撲圖上的設計以及重疊路徑造成的影響。因為 Y 字形底端的節點和 Y 字形頂端兩個節點分別扮演來源以及目的端（分別為一對二或是二對一的狀態），這和其餘拓撲圖的設計不同。因此所有組別在效能上的表現，差異度並不明顯。



圖六 Y 字形網路拓撲圖

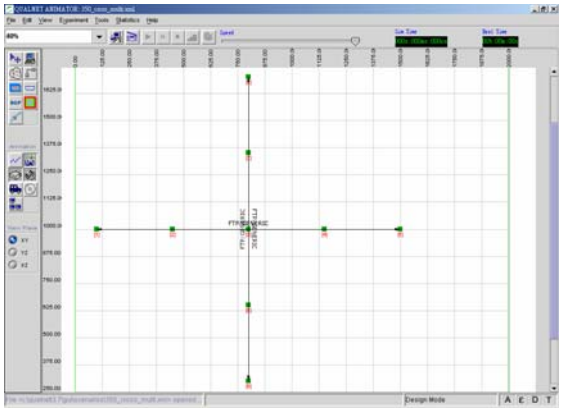


圖七 Y 字形網路拓撲_封包丟棄

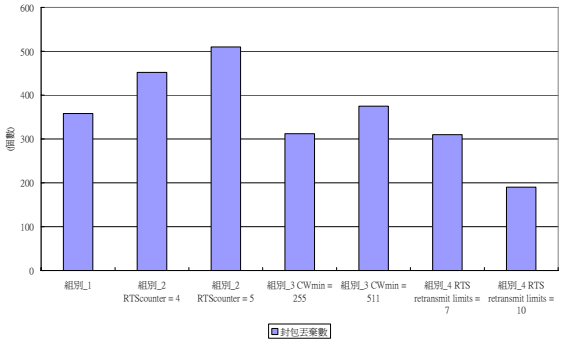


圖八 Y 字形網路拓撲_吞吐量

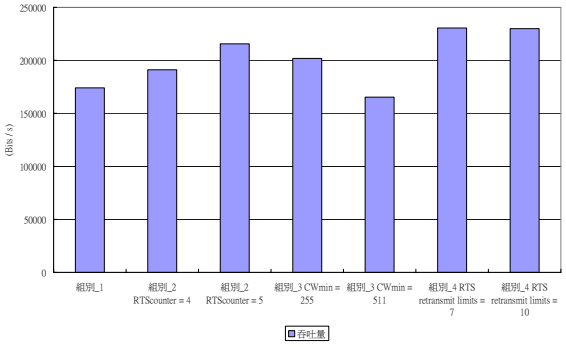
圖九為十字形網路拓撲，在圖十一裡，吞吐量表現最好為組別 4，而且在封包丟棄數目方面，組別 4 RTS retransmit limits = 7 為最低。其餘組別在吞吐量的上能上面亦皆有不錯的表現，其中組別 4



圖九 十字形網路拓撲圖



圖十 十字形網路拓撲_封包丟棄



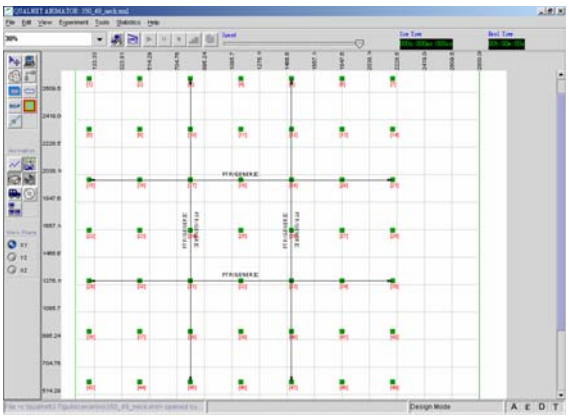
圖十一 十字形網路拓撲_吞吐量

RTS retransmit limits = 7 和組別 4 RTS retransmit limits = 10 的吞吐量分別為組別 1 的 1.33 以及 1.32 倍。封包丟棄數目亦為組別 1 的 0.87 以及 0.53 倍。由此可見，在十字形網路拓撲環境下，本研究所提出的方法，能有效改善 TCP 吞吐量的效能。

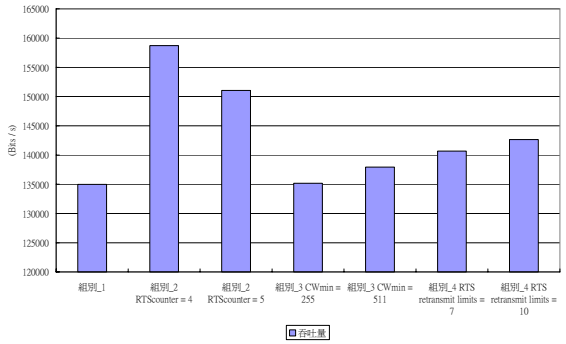
圖十二為一井字形網路拓撲，圖十三和圖十四為二十二點井字形拓撲的模擬結果，這是模擬最複雜且最忙碌的網路情形，每條路徑皆有五個轉送節點，且總共有四條雙向資料流做相互交叉傳送的動

作。在此網路拓撲下，本文將所有方法的結果一起比較，假若能在如此複雜的情況下，有不錯的效能產生，此為一個不錯的改善方法。藉由圖十四可以發現到所有組別在吞吐量上面的表現皆比組別 1 優異，但是效能表現較為突出的組別卻是本文所提出的另一個方法，組別 2。尤其是組別 2 RTScounter = 4 的吞吐量為組別 1 的 1.18 倍。因此可以發現到當網路為極度壅塞的狀態時，利用加大倒數的時間來減少路徑失連的機率，遠小於利用節點變成空閒時自動通知的機制。雖然組別 2 在吞吐量上面的表現較為優異，但是所需要付出的成本價值卻高出其他組別甚多。

由此可見儘管網路的狀態很擁塞複雜，但是本文提出的改善方法，在提升網路流量方面仍然有不錯的效能改善。



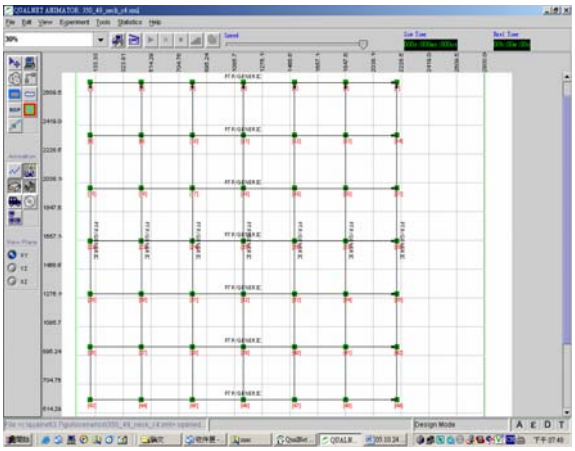
圖十二 井字形拓撲圖



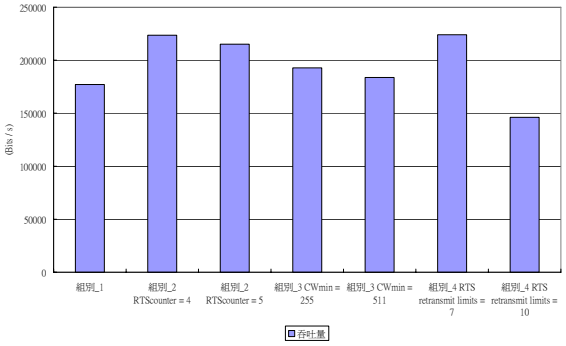
圖十三 井字形拓撲圖_吞吐量

圖十五是網狀網路拓撲，其資料流改為單向傳送，圖十六及圖十七為網狀拓撲下的模擬結果。由模擬結果可以發現到，除了組別 4 RTS retransmit limits = 7 在吞吐量上面表現的效能為最高外，組別 2 在吞吐量上面的效能仍有不錯的改善及提升。由此可知，當網路極度壅塞時，除了加大倒數時間外，亦可藉由組別 2 的方法進行改善（當不考慮成本價值的狀況下）。

與 802.11 標準設定下的模擬數據相比，在 TCP 吞吐量上，本文所提出組別 4 RTS retransmit limits = 7 為組別 1 的 1.26 倍。因此可以發現本文所提出的方法在修改成本最小化的情況下，能獲得不錯的效能改善。



圖十四 網狀拓撲圖



圖十五 網狀拓撲圖_吞吐量

五、結論

本文的研究方向以 802.11 ad hoc 網路中 MAC 層的 DCF 機制作為探討對象，藉由相關文獻的探討改變 DCF 裡的競爭窗口選取規則，以及加大 RTS 最大重傳次數。並以修改原本 DCF 機制幅度與成本最小為考量，作為本文所提出改善 TCP 吞吐量的最佳方法。經由上述所有的模擬結果，可以知道改變競爭窗口選擇規則，可以有效的提升 TCP 吞吐量效能。另外再加上改變 RTS 重傳的上限次數，對於整體的網路流量亦有明顯的改善。

在未來的研究中，會繼續針對本研究所提的方法，在 TCP 吞吐量上做穩定性的探討。以及在其他網路拓撲下的效能和提出更有效益的修正與改進，以期能提升無線自組織網路使用上的便利性。

六、參考文獻

[1] 李雲、陳前斌、隆克平、吳詩其，“無線自組織網路中 TCP 穩定性的分析及改進”, 軟件學報, Vol. 14, No. 6, pp.1178-1186, 2003 年.

[2] Broch J, Maltz DA, Johnson DB, Hu YC, Jetcheva J. “A performance comparison of multi-hop wireless Ad Hoc network routing protocols”, The 4th Annual ACM/IEEE International Conference on Mobile Computing and Networking, ACM Press, pp. 85-97, 1998.

[3] ANSI/IEEE Standard 802.11, “Part 11：Wireless LAN Media Access Control (MAC) and Physical Layer (PHY) Specifications”, http://standards.ieee.org/getieee802/, 1999.

[4] Josh Broch, David B. Johnson, David A. Maltz., “The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks”, Internet-Draft, March, 1998.

[5] Kaixin Xu, Mario Gerla, Lantao Qi, Yantai Shu “Enhancing TCP Fairness in Ad Hoc Wireless Networks Using Neighborhood RED”, ACM MobiCom, pp.16-28, 2003.

[6] Krishna Kanth T., Sabeel Ansari, Anurag Kumar, and Mohammed H. Mehkri, “Performance Enhancement of TCP on Multihop Ad hoc Wireless Networks”, in Proceedings of IEEE International Conference on Personal Wireless Communications, 2002.

[7] Scalable Network Technologies Qualnet Ver. 3.7, www.scalablenetworks.com.

[8] S. Xua, T. Saadawi, “Does the IEEE 802.11 MAC Protocol Work Well in Multihop Wireless Ad Hoc Networks?”, IEEE Communications Magazine, Vol. 39, pp.130-137, 2001.

[9] Shugong Xua, Tarek Saadawi, “Revealing the problems with 802.11 medium access control protocol in multi-hop wireless ad hoc networks”, ELSEVIER Computer Networks, Vol. 38, pp. 531–548, 2002.

[10] Yong Xiao, Xiuming Shan, Yong Ren, “Game Theory Models for IEEE 802.11 DCF in Wireless Ad Hoc Networks”, IEEE Radio Communications, March, pp. S22-S26, 2005.