行政院國家科學委員會專題研究計畫　成果報告

子計畫四：無線行動隨意網路上之移動支援與電源管理協定

(2/2)

計畫類別： 整合型計畫
計畫編號： NSC93-2219-E-009-003-
執行期間： 93 年 08 月 01 日至 94 年 07 月 31 日
執行單位： 國立交通大學資訊工程學系(所)

計畫主持人： 曾煜棋

報告類型： 完整報告
報告附件： 出席國際會議研究心得報告及發表論文
處理方式： 本計畫可公開查詢

中　華　民　國 94 年 8 月 9 日

行政院國家科學委員會補助專題研究計畫結案報告

無線行動隨意網路上之移動支援與電源管理協定 (2/2)

計畫類別： 整合型計畫
計畫編號： NSC 93-2219-E-009-003
執行期間： 93 年 08 月 01 日至 94 年 07 月 31 日

計畫主持人：曾煜棋

執行單位：國立交通大學資訊工程研究所

中 華 民 國 94 年 8 月 9 日

摘要

此子計畫包含了兩大工作重點，(1)多階層無線隨意行動網上移動之管理和(2)多階層無線隨意行動網上電源之管理。在移動管理方面，我們已知 Mobile IP 是一個被廣泛使用於支援無現網際網路環境的標準，而隨意網路也已提供於多跳躍(Multi-hop)的無線區域網路連上網際網路的可能，所以在移動管理方面，我們建構一個以 Wi-Fi 為基礎的具多步跳躍(Multi-hop)通訊子網路，作為在多階(Multi-tier)行動隨意網路架構中主要支援行動隨意網路的部分。而為了達到多系統整合與漫遊的目的，我們並提出解決多階行動隨意網路與 Mobile IP 整合在一起的解決方法。

在無線網路中電池可說是行動主機的生命來源。如何做好電源管理藉以延長行動主機之運作時間，是無線行動隨意網路能否長期運作的一大關鍵，針對上述之問題我們分別在多階多步跳躍通訊子網路與藍芽通訊子網路在各層協定上提出有效的解決方法，並且對於多階隨意行動網路在整合不同系統時所需考量的省電機制，提出解決良策。此外我們也針對隨意網路提出省電的網路協定，例如繞路協定。

關鍵詞： 無線隨意行動網路，省電模式，IEEE 802.11，藍芽，移動管理，Mobile IP

目錄

報告內容

I. 前言

在多階層無線隨意行動網路上，由於行動主機具有可任意移動的特性，因此如何使行動主機的使用者感受不到移動性的影響，而能如同在有線的環境中一樣，以相同的方式運作及接受服務，是無線行動隨意網路能否成功的一大關鍵。計劃第一部分著重於如何掌握行動通訊裝置目前之位置與狀況，使得通訊模式、資料傳輸率可視情況彈性切換，以節省資源並達最大資料傳輸量。

因此本子計劃的其中一個目標便是設計適用於隨意網路的移動管理方法，我們整合了 Mobile IP 與隨意網路，使得在隨意網路下的行動主機感受不到因為移動所產生的影響。Mobile IP 的作用，簡單地說，就是讓同一台電腦，從原本所在的網路移至另一個網路時，可使用原來的 IP address。我們會在研究方法中去敘述，我們如何去結合 Mobile IP 與隨意網路，以達成移動管理的目的。

此外多階層無線隨意行動網路上電源之管理也是此子計劃的另一個目標，我們知道無線隨意行動網路是由行動主機所構成之網路，對於所有的行動主機而言，電池是其唯一的動力來源，沒有電源，行動主機將無法運行。然而，在可預期的近幾年之內，電池科技將不會有很大的突破，因此，如何延長電池的使用時間，對於無線隨意行動網路而言，是一個十分重要的課題。

本計劃在電源管理方面之主要目標之一，是在以 IEEE 802.11 為基礎之無線隨意行動網路上，研究省電模式之管理。當行動主機進入省電模式後，行動主機必須要去預測其他行動主機醒來的時機，以便彼此可以傳送或接收訊息，然而，由於沒有中央控管的機制，再加上因為不可預期之移動性與無線電干擾造成的封包延遲，使得無線隨意行動網路上的同步難以達成，也因此讓這項預測的工作變得十分複雜而且困難。除此之外，在無線隨意行動網路上設計省電模式的另一個挑戰，即是行動主機可能無法察覺其鄰居的存在，會發生這個現象的主要原因，是因為彼此醒來時間的不相同，再加上傳送與接收訊息的次數也減少了所造成的，而不精確的鄰居資訊可能會使得繞徑協定發生錯誤。為了解決上述之問題，我們以發送更多的 beacon、讓行動主機定期醒來的時間有所重疊、以及醒來時間的預測等策略，設計了三種省電協定，分別為： dominating-awake-interval protocol、periodically-fully-awake-interval protocol 與 quorum-based protocol，其中，每種省電協定其醒來的模式都不相同，但都能保證不同步的行動主機的醒來時間可以相互重疊。
本計劃的另一個目標，即是去探索在藍芽規格裡一種省電模式 – sniff mode。在這個模式下，一個角色為 slave 的移動設備，只要定期醒來接收 master 端的資料，而不須時時刻刻待命，浪費無謂的電池能量。然而，一個重要且具挑戰性的問題是，master 該如何去排程所有

在子網內 slaves 的睡眠週期，使得在省電的同時，還能兼顧網路傳輸效能，這樣的一個問題，我們稱之為"省電排程問題"。 針對這個問題，我們提出一個可調式的排程機制，動態調整每個 slave 的睡眠參數，使其符合每個 slave 與 master 之間經常改變的不對稱交通流量需求，並達到 slave 端的省電目標。

此外既有的通訊協定(protocol)大部分都只在某些層上去加進省電的功能，例如實體層(physical layer)、媒體擷取層(MAC layer)、網路層(network layer)，但並非全部。所以我們也發展出一套整合各層的能量感知(power aware)、位置感知(location aware)的通訊協定，以達成隨意網路的電源管理。

II. 研究目的

本子計劃的主要目的為提供無線行動隨意網路的移動管理與電源管理，以細項來說明的話，本計畫的目的包括：

（一）整合 Mobile IP 與隨意網路，設計無線行動隨意網路下的移動管理協定。

（二）Bluetooth 多個 piconets 共存之干擾模型

（三）IEEE 802.11 為基礎之無線隨意行動隨意網路上的省電協定。

（四）藍芽網路中省電模式之可調式排程機制。

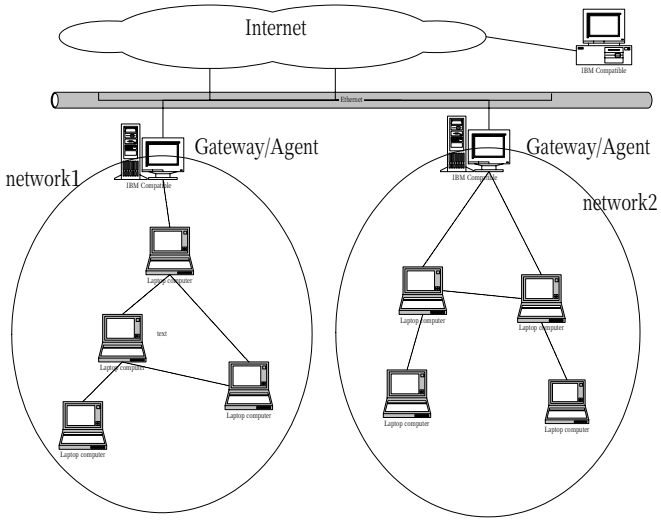
（五）整合網路各層的能量感知(power aware)、位置感知(location aware)的通訊協定，以達成隨意網路的電源管理。

III. 研究方法

A. 無線行動隨意網路上的移動管理協定 – 結合 Mobile IP 與隨意網路

在移動管理的部份，我們實作了一個架在 Layer 3(Network Layer)以及 Layer 2(Datalink Layer) 的無線隨意網路通訊協定。並加上 Mobile IP 的機制，使它可以支援跨大到一整個網路的移動管理。

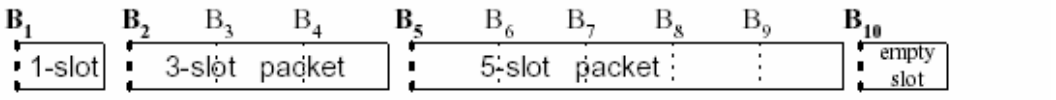
Mobile IP 的作用，簡單地說，就是讓同一台電腦，從原本所在的網路移至另一個網路時，可使用原來的 IP address。我們使用 Mobile IP 來做子網路之間的資料封包傳遞，對於子網路之內的資料封包傳遞，我們則用 Ad Hoc DSDV protocol 來做轉送。如下圖所示，Network1 和 Network2 分屬於不同的子網路，兩者之間的資料傳送靠 Mobile IP 來服務，而子網路之內的資料傳送則使用 Ad Hoc DSDV protocol 的服務。



B. Bluetooth 多個 piconets 共存之干擾模型

在藍芽(Bluetooth)操作的頻帶(operating band)上，藍芽微網(piconet)間會互相干擾. 我們研究在多個微網共存的環境（multi-piconet environment）中，試著從數學機率分析的角度，來評估因為共同頻道干擾（co-channel interference）而導致封包碰撞（packet collision）所造成對整體網路效能的影響程度，以供藍芽網路佈署時有用之參考依據。

在我們的分析中，考慮 N 個微網同時存在，且彼此間的距離非常接近，足以互相影響。因此，每一個微網有 N-1 個可能影響到其傳送的對手。而在任意的時間點上，若有兩個微網欲使用同樣的頻率作傳輸用，則視此傳送的封包被損壞。並假設 $\lambda 1$, $\lambda 3$, $\lambda 5$ 為 1-slot, 3-slot , 5-slot 之封包的 arrival rates 。由於在多時槽的封包中，只有第一個時槽算入 arrival，則顯然 $\lambda 1+3\lambda 3+5\lambda 5 \leqq 1$。再將未使用的時槽視為許多單時槽(single slot)的封包，以 $\lambda 0$ 代表其所佔的比率，則 $\lambda 0=1-(\lambda 1+3\lambda 3+5\lambda 5)$。現考慮環境中同時存在兩個微網 X 與 Y，欲分析 X 所受到的干擾，推倒出 X 中，長度為 i 的封包之傳送成功率 Ps(i)，其中 i = 1, 3 ,5。接著我們將引入時槽分隔線(slot delimiter)的觀念來分析。就 X 中的任一時槽做考慮，由於 X 和 Y 不同步，則 Y 中可能有 1 或 2 個 slot delimiters 可能跨越(cross)此 X 的時槽。由於我們考慮的是連續的機率，此後的討論會將兩個 slot delimiters 跨越的機率忽略。以 X 中的一個 1-slot 的封包為例：此封包唯有在 slot delimiters 前後的封包接不造成干擾的情況下方能成功。故成功率依 Y 微網的傳送與否，可能為 1 ,(78/79)或(78/79)2。
下圖為 slot delimiter 之示意圖：



i. B1,B3,B5:分別為為 1-slot, 3-slot, 5-slot 的封包的開頭。
ii. B3,B4:分別為 3-slot 的封包中，第二與第三時槽的開頭。
iii. B6,B7,B8,B9：分別為 5-slot 的封包中，第二，三，四，五時槽的開頭。
iv. B10：為未使用之時槽的開頭。

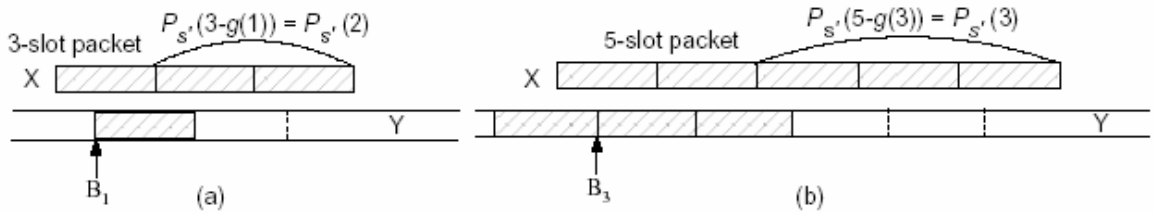
顯然，B1 所出現的比率為 $\lambda 1$，B2,B3,及 B4 出現的比率皆為 $\lambda 3$；B5,B6,B7,B8,B9 出現，比率機各為 $\lambda 5$，而 B10 出現的機率為 $\lambda 0$。以函式的方式來表示 Bj 則得: $\lambda(Bj)j=1..10$。X 中的一個封包，若被 Y 中一個類型 B1/B2/B5 為 delimiter 跨越(cross)，則 delimiter 前後各有一個封包，則 Y 中即有兩個封包有可能對 X 造成影響，反之，若是其他類型的 delimiter，則 Y 中只有一個封包會對 X 造成影響。在推導成功率 Ps(i)時，我們先推出一個特殊情況 Ps’(i)。Ps’(i)只代表兩種情況的成功傳送比率。(1)Y 中第一個 delimiter 跨越 X 封包的型態為 B1/B2/B5/B10。(2)第一個 delimiter 前面的所有時槽並未發生干擾的情況。依此定義我們可以推導出 Ps(i)：

$$P_S(i)=\sum_{j=1}^{10}\lambda(B_j)\bullet f(j)\bullet P_{S'}(i-g(j))$$

其中 P0=(78/79)

$$f(j)=\begin{cases}(1-\lambda_0)\cdot P_0^2+\lambda_0\cdot P_0 & \text{for } j=1,2,5\\ (1-\lambda_0)\cdot P_0+\lambda_0 & \text{for } j=10\\ P_0 & \text{otherwise}\end{cases},$$

上面的式子中，我們定義 g(j)為 delimiter Bj (j=1..10)後剩餘的時槽數目。例如:g(1)=1,g(3)=2,g(7)=3 …。以下為推導的概念：λ(Bj)是第一個 delimiter 為 Bj 的機率。f(j)是 X 在 delimiter 前後的封包都沒有發生碰撞(collision)的機率。在此之後，X 尚有 i-g(j)的時槽需要傳送，且對這 i=g(j)的時槽來看，第一個 delimiter 必為 B1/B2/B5/B10 中的一種。故我們可以套用 Ps`(i-g(j))。下圖為上述概念的例子：



(a)圖，X 中一個 3-slot 的封包，第一個 delimiter 為 B1，首先我們知道 X 中第一個時槽成功傳送的機率為 f(1)，則剩下的兩個時槽的成功機率為 PS’(2)，故對整個封包而言，成功的機率為兩者相乘，即 f(1) × PS’(2)

(b)圖為一個 5-slot 的封包，所遭遇到的第一個 delimiter 為 B3，同理，其傳送成功的機率為 f(3) × PS’(3)

再者，可以用遞迴(recursive)方式來表示 Ps’(k)如下：

$$P_{S'}(k)=\frac{\lambda_0}{\lambda_0+\lambda_1+\lambda_3+\lambda_5}\times P_{S'}(k-g(10))+\frac{\lambda_1}{\lambda_0+\lambda_1+\lambda_3+\lambda_5}\times P_{S'}(k-g(1))+$$

$$\frac{\lambda_3}{\lambda_0+\lambda_1+\lambda_3+\lambda_5}\times P_{S'}(k-g(2))+\frac{\lambda_5}{\lambda_0+\lambda_1+\lambda_3+\lambda_5}\times P_{S'}(k-g(5))$$

而我們假設在 K≦0 時，Ps’(K)=1。這可以想做假設第 0 個時槽，或更早以前的時槽失敗的機率為零。上述的推導為兩個微網的環境，當環境有 N 個微網同時啟動的狀況下，延伸上述的推導，此時對微網 X 而言，有 N-1 微網成為干擾源(interference source)。由於這 N-1 個干擾源各自獨立，X 中第 i 個時槽的成功機率為 Ps(i)N-1

故單一網路 X 的生產量(throughput) T 為：

T=λ1 × Ps(1)N-1 × R1 +3 ×λ3 × Ps(3)N-1 × R3 + 5 ×λ5 × Ps(5)N-1 × R5

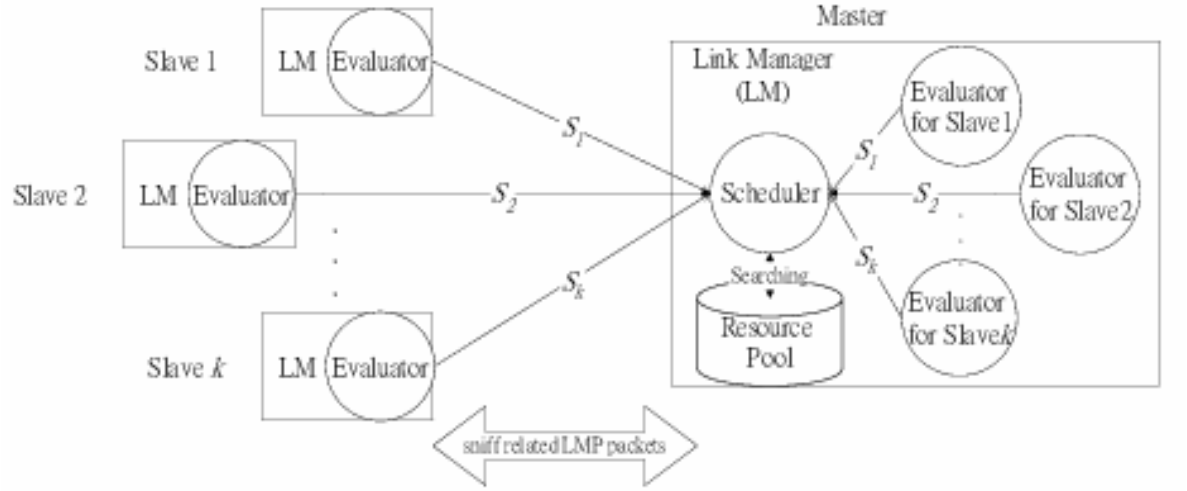
其中 R1, R3 和 R5 分別為 1-slot, 3-slot, 5slot 封包各自的資料負載量，其單位為(位元/時槽)，而整個網路的生產量(throughput)為 N × T。

C. Bluetooth 系統上電源模式之管理

藍芽技術(Bluetooth)在智慧家庭(Smart Homes)是一個很重要的元件，在智慧家庭(Smart Homes)中，大部分的藍芽裝置都是可攜帶(portable)且裝電池(battery-operated)的，所以省電

(power saving)是個很重要的議題。我們研讀了藍芽技術管理的低耗電監聽模式(sniff mode)，從屬裝置(slave)是被允許可以週期性醒來(awake)的，一個很重要的議題就是去解決如何分配(schedule)每個從屬裝置的週期，使得網路流量需求(traffic requirement)和使用電量需求(powering saving requirement)能達成較好的平衡。

我們發展出了一個調整監聽模式相關參數之策略(由於在藍芽規格裡主裝置與從屬裝置可以透過一些控制封包（control packet）來協調從屬裝置進入監聽模式之運作參數（sniff parameters），然而規格裡並無明確交代該如何設計這些參數)。讓主裝置(master)與從屬裝置(slave)可以根據目前的數據流量(traffic)，在不犧牲工作效能的條件下，動態協調出一組合適的監聽參數，使得從屬裝置(slave)能進入低功率之監聽模式(sniff mode)以節省不必要的電池電源消耗，達到網路生命週期、效能提昇之目的。



上圖顯示出我們監聽排程(sniff scheduling)的協定，主裝置有三個元件，評估器(Evaluator)、排程運算器(Scheduler)、時槽資源庫(RP(resource pool))，評估器的工作是去評估衡量從屬裝置使用目前的監聽參數有沒有效率，如果不適合，會去觸發排者運算器重新調整監聽參數。時槽資源庫負責管理可用的監聽時槽(sniff-attempt slots)，這些可用的時槽可以給從屬裝置來使用。主裝置會週期性對每個從屬裝置執行評估器來評估狀況，如果有需要的改變的話，會產生一個訊息通知排程運算器，然後排程運算器會去時槽資源庫尋找適合這個從屬裝置的新的監聽參數。

跟其它已有的協定比較，我們的排程考慮同時有多個從屬裝置在運作，而且我們的排程根據情況做更精確、更動態的調適。

D. IEEE802.11 為基礎之無線隨意多跳躍 (Multi-hop) 網路上的電源管理協定

我們提出以下的策略來設計 multi-hop 隨意行動網路上的省電協定：

1. 發送更多的 beacon：為了避免鄰居資訊不精確的問題，我應該讓進入省電模式的行動主機多發送 beacon。因此，在每個 beacon interval，每個行動主機都會發送一次 beacon，即使別的行動主機已經成功發送 beacon。
2. 讓行動主機定期醒來的時間有所重疊：我們將重新設計行動主機的睡眠模式，使得行動主機即使在不同步的種況下，也能保證彼此醒來的時間相互重疊，因而讓彼此都能

收到對方的 beacon。

3. 醒來時間的預測：當收到鄰居的 beacon 後，我們便可知道鄰居的存在，同時也可知道鄰居的時脈。此時，便可依據彼此的時間差來預估鄰居醒來的時間，以便在鄰居醒來時傳送訊息給對方。在此，我們只是去預測鄰居的醒來時間並不做同步的動作。

依據以上的策略我們提出了三種省電協定，每種省電協定，其醒來的模式都不相同，但都能保證不同步的行動主機的醒來時間可以相互重疊。此外，我們也重新設計了 beacon interval 的結構。在每個 beacon interval 中包含了三個 window，即 beacon window、MTIM window 與 active window。Beacon window 是讓每個行動主機發送 beacon 的，MTIM window 是讓行動主機發送 MTIM(Multi-hop TIM)訊框的，MTIM 訊框是用來通知其他行動主機有資料封包要傳送給它，在 active window 中，行動主機會保持清醒，以便接收其他行動主機重送之訊息。在不同的省電協定中，其 beacon interval 的結構也會略有不同，我們將會在後面說明之。以下我們將分別說明三種不同的省電協定：

1. Dominating-Awake-Interval 協定

   要讓進入省電模式且不同步的行動主機醒來的時間互相重疊，最簡單的方法就是讓行動主機在每個 beacon interval 醒來的時間超過一半。但是，如此並不足以保證彼此都能收到對方的 beacon。因此，我們必須調整 beacon interval 的結構。我們將 beacon interval 分為奇數週期與偶數週期。在奇數週期 beacon interval 由 beacon window 開始，接下來是 MTIM window 與 active window；在偶數週期，則是由 active window 開始，接下來是 MTIM window 與 beacon window。如此一來，便可保證進入省電模式且不同步的行動主機彼此都能收到對方的 beacon。

2. Periodically-Fully-Awake-Interval 協定

   如果我們讓進入省電模式的行動主機每 T(T 是自然數)個 beacon interval 完全醒來一次，也可以保證進入省電模式且不同步的行動主機彼此都能收到對方的 beacon。當行動主機進入省電模式後，每三個 beacon interval 便會醒來一次，如此一來，每三個 beacon interval 便可收到鄰居的 beacon 一次。

3. Quorum-Based 協定

   我們依據分散式系統 quorum 的概念設計了這個省電協定，如上圖所示，我們將 n2 個 beacon interval 分為一群，每個進入省電模式的行動主機從 n2 個 beacon interval 中任意挑選一行與一列的 beacon interval 完全醒著，如此也能保證進入省電模式且不同步的行動主機彼此都能收到對方的 beacon。

當我們能保證進入省電模式且不同步的行動主機彼此都能收到對方的 beacon 後，行動主機便可依據上述三種省電協定的清醒模式來預測鄰居醒來的時間，因而便能順利的傳送

資料給對方。因此，所有的行動主機都進入省電模式，各種繞徑協定仍能正常運作。

E. 無線行動隨意網路上以電源為考量之通訊協定

既有的通訊協定(protocol)大部分都只在某些層上去加進省電的功能，例如實體層(physical layer)、媒體擷取層(MAC layer)、網路層(network layer)，但並非全部。所以我們發展出一套整合各層的能量感知(power aware)、位置感知(location aware)的通訊協定。

我們的方法是利用行動裝置的位置資訊來達成所有協定層(protocol layers)的能量管理(power conservation)，這些協定層包含實體層(physical layer)、媒體擷取層(MAC layer)、網路層(network layer)，是個完全能量感知(power aware)和位置感知(location aware)的通訊協定。

類似手機網路(cellular network)，我們的方法首先也是把網路區域分割成正方形或六角形(hexagons)稱作方格(grid)，這個方法可以使得管理電源和管理移動性(mobility)變得比較容易處理。每個行動裝置要裝上全球衛星定位系統(GPS)使得可以分割網路區域變得一個個的方格(grid)。

在路由層(routing layer)，為了選擇良好的路由路徑(routing path)，我們為每個方格(grid)定義了一個聚集能量等級(collective energy level)，這個等級表示方格內所有可用電量的總和，等級愈高表剩餘電量愈多，路徑會依照這個等級選擇達成省電。

電源管理(power management)方面，基本的概念，只有一些行動裝置需要醒來(awake mode)，其它的行動裝置可以進入休眠狀態(sleep mode)，可以達成省電的效果。我們的協定中，為了省更多的電，當在維持網路連接性(network connectivity)時，有一個電源管理機制(power mode management mechanism)來減少作用中的行動裝置(awake hosts)，在方格內只有電量最高的醒著，這個裝置稱作領導者(leader)，其它都進入休眠模式。方格間的通訊是領導者跟領導者直接通訊。而在方格內通訊，領導者(leader)必需負責<1>維持本地方格的路由表(local grid's routing table)。<2>轉送 RREQ、RREP、資料封包(data packets)。<3>維持監看本地格內其它的行動裝置。<4>暫存已經在休眠中的行動裝置的封包。當領導者(leader)要離開或電量不足時，必需從方格內選出一個新的行動裝置來當領導者(leader)。

在媒體擷取層(MAC layer)，避免不必要的碰撞，我們把方格內和方格間的通訊用時間分開，隨著時間的不同來決定是否是屬性方格內通訊時間或方格間通訊時間。最後，在實體層(physical layer)我們的電源控制會隨著行動裝置的位置來做調整。

因此我們的方法是結合了網路的各層，整體的做考量，以達成於隨意網路下電源管理的目的。

IV. 研究成果

本計畫依時程順利進行，並獲致豐碩成果，相關論文發表如下，論文全文詳如附錄：

[1] Y.-C. Tseng, C.-C. Shen, and W.-T. Chen “Mobile IP and Ad Hoc Networks: An Integration and Implementation Experience”, ***IEEE Computer***, Vol. 36, No. 5, May 2003, pp. 48-55. (SCI, EI)

[2] T.-Y. Lin and Y.-C. Tseng, “Collision Analysis for a Multi-Bluetooth Picocells Environment”, ***IEEE Communications Letters***, Vol. 7, No. 10, Oct. 2003, pp. 475-477. (SCI, EI)

[3] T.-Y. Lin and Y.-C. Tseng, “An Adaptive Sniff Scheduling Scheme for Power Saving in Bluetooth”, ***IEEE Wireless Communications***, Vol. 9, No. 6, Dec. 2002, pp. 92-103. (SCI, EI)

[4] Y.-C. Tseng, C.-S. Hsu, and T.-Y. Hsieh, “Power-Saving Protocols for IEEE 802.11-Based Multi-Hop Ad Hoc Networks”, ***IEEE INFOCOM***, 2002.

[5] Y.-C. Tseng and T.-Y. Hsieh, “Fully Power-aware and Location-aware Protocols for Wireless Multi-Hop Ad Hoc Networks”, ***ICCCN***, 2002.

[6] Y.-C. Tseng, C.-S. Hsu, and T.-Y. Hsieh, "Power-Saving Protocols for IEEE 802.11-Based Multi-Hop Ad Hoc Networks", ***Computer Networks***, Elsevier Science Pub., Vol. 43, No. 3, Oct. 2003, pp. 317-337. (SCIE, EI)

附錄

I. 國外差旅費使用情形

本計畫之國外差旅費使用兩次，，兩次會議的行程如下:

- 林致宇，參加 IEEE BroadNets(First Annual International Conference on Broadband Networks)會議，2004/10/25 ~ 2004/11/1，San Jose，USA

    - 這一個研討會主要探討的研究主題是集中在寬頻網路(Broadband Network)的範疇，討論的議題包含了光學網路(Optical Network)的發展、無線網路(Wireless Networks)的發展及無線感測網路(Wireless Sensor Network)等等不同的領域，這些領域跟本子計劃的議題也是非常相關的。我們針對感測網路上的追蹤(Tracking)問題提出了一個省電的追蹤方法，並在會議中發表。

- 王友群，參加 IEEE WICON (First International Conference on Wireless Internet)，2005/7/10 ~ 2005/7/15，Budapest，Hungary

    - 本次會議所涵蓋的議題主要圍繞在無線網路中，包含有媒體層及網路層的協定設計(MAC Protocols & Routing Protocols)、基地台換手問題(Handover Issue)、無線感測網路(Wireless Sensor Networks)、網路傳輸品質議題(Quality of Services，QoS)、下一世代的無線網路服務(Next Generation Wireless Services)等相關議題。我們針對感測器佈置(Sensor Deployment)的議題提出一套節省成本的方法，並在本次會議中發表與討論。

詳細的報告於附於下頁，會議中發表的論文則附在附錄 II。

□ 赴國外出差或研習
□ 赴大陸地區出差或研習
□ 出席國際學術會議
□ 國際合作研究計畫出國

心得報告

計畫名稱	無線行動隨意網路上之移動支援與電源管理協定	計畫編號	NSC 93-2219-E-009-003-
報告人姓名	林致宇	服務機構及職稱	國立交通大學資訊工程學系博士班學生
會議/訪問時間地點	第 1 屆一年一期國際寬頻網路研討會 (First Annual International Conference on Broadband Networks) 2004/10/27~2004/10/19, San Jose, USA		
會議名稱	第 1 屆一年一期國際寬頻網路研討會(Broadnets 2004)		
發表論文題目	Structures for In-network Moving Object Tracking in Wireless Sensor Networks		

一、參加會議經過：

這一次研討會的舉辦地點是位於美國加州(California)聖荷西市(San Jose)的希爾頓飯店，會議中除了個人報告(Oral)以及張貼海報(Poster)外，還有邀請知名人士針對寬頻網路的發展做專題演講。

我個人報告的時間是會議的第三天(當地時間 10/28)，這個 Session 是針對無線感應網路 (Wireless Sensor Network)做討論的子會議，每個人約有 15 分鐘左右的時間可以報告自己所做的研究，並預留 5 分鐘的時間讓其他的聽眾得以詢問問題。在這個子會議中，大家談論的議題包括在無線感應網路中感應器的佈置(Placement)方法、在感應網路中如何快速地形成一個個區域的群組(Local Clusters) 、感應器如何藉著週期性的關掉電源以達到節省能源的目地等，讓我對這個領域有著更深入的了解。

二、與會心得：

這一次研討會主要探索的方向是集中在寬頻網路(Broadband Network)的範疇，討論的議題包括有光學網路(Optical Network)的發展、無線網路(Wireless Networks)的發展及無線感應網路(Wireless Sensor Network)等等不同的領域，由於這些領域和我目前的研究習習相關，因此藉由聆聽別人對這些領域的研究報告，讓我了解到目前別人在這些領域中是朝哪些方向努力、並且發掘到哪些問題，相信這對我目前的研究有著極大的幫助。

此外，藉由這次出國的機會，讓我得以和不同國家的人們交談並交換心得，除了能夠加強自己的語文能力外，也增加了自己的國際觀。

三、建議與結語：

在這次的會議中，有幾個與會的人員於會後針對我所做的報告詢問相關的問題，而我也於會後去和別的報告者交換心得，我覺得這是一個很不錯的經驗，我會建議學校或相關研究單位除了鼓勵學生出國於國際會議報告之外，更應鼓勵學生積極與其它學者做交流，這樣應更能提升出國報告的收穫。

四、攜回資料：

BROADNETS 2004 會議手冊：記載本研討會會議流程、報告人員、地點、及報告題目等與研討會相關的會議手冊，其中並包含了會議中所有的論文集。

五、其他：無

□ 赴國外出差或研習
□ 赴大陸地區出差或研習
□ 出席國際學術會議
□ 國際合作研究計畫出國

心得報告

計畫名稱	無線行動隨意網路上之移動支援與電源管理協定	計畫編號	NSC 93-2219-E-009-003-
報告人姓名	王友群	服務機構及職稱	國立交通大學資訊工程學系博士班學生
會議/訪問時間地點	第 1 屆無線網際網路國際研討會(WICON 2005)，7/10 日-7/15 日在匈牙利布達佩斯(Budapest)-維斯格蘭德(Visegrád)的 Danubius Spa & Conference Hotel Visegrád 旅館舉行		
會議名稱	第 1 屆無線網際網路國際研討會(WICON 2005)		
發表論文題目	Efficient Deployment Algorithms for Ensuring Coverage and Connectivity of Wireless Sensor Networks		

一、參加會議經過：

本次研討會的舉辦地點是位在匈牙利布達佩斯(Budapest)-維斯格蘭德(Visegrád)的 Danubius Spa & Conference Hotel Visegrád 旅館，會議中除了以口頭報告方式的技術性會議(technical session)外，還邀請學術及產業界的知名人士就相關議題發表主題演說(Keynote Speech & Tutorials)。

我個人報告的時間是會議的第三天下午(當地時間為7/12)，是歸類在無線感測網路的子會議(Wireless Sensor Networks)中。該子會議的組別共有4組，每組由一人代表報告，每個人報告的時間約有20分鐘，並且預留5分鐘的時間讓其他與會聽眾發問。在我參加的這個子會議中，大家所討論的議題包含有感測器(Sensors)的部署(Deployment)方式、網路傳輸資訊匯集(Data Aggregation)技術、以及省電的協定設計等。藉由別人的報告以及大家共同討論的問題當中，不僅讓我充分了解到自己研究的改進方向及空間、其他研究者在這個議題的研究方向、也同時讓我對這方面的領域有更進一步的理解。

二、與會心得：

本次會議所涵蓋的議題主要圍繞在無線網路中，包含有媒體層及網路層的協定設計(MAC Protocols & Routing Protocols)、基地台換手問題(Handover Issue)、無線感測網路(Wireless Sensor Networks)、網路傳輸品質議題(Quality of Services，QoS)、下一世代的無線網路服務(Next Generation Wireless Services)、以及其他相關議題等。我個人除了參加我本身所報告的子會議外，也同時參加了幾個我比較感興趣的子會議。藉由聆聽別人的技術報告以及詢問相關問題，讓我對其他領域目前的發展更有概念，也大致了解其他人的研究方向。

此外，在我個人報告的部分當中，不僅讓我吸取了許多寶貴的經驗，在報告過程中以及報告之後、聽眾們的問題以及建議，也讓我對目前的研究方向有著更進一步的

想法。另外，藉由出國參加國際研討會，不但能加強自己的語文能力外，與不同國家人們的討論當中，也能夠培養國際觀。

三、建議與結語：

我個人認為，能夠參加這次的國際研討會，對我來說可以是一個非常寶貴且收穫良多的經驗。因此，我建議學校能夠多鼓勵學生出國在國際研討會上報告自己的研究成果。這樣不但能夠開擴學生的視野、培養學生的國際觀、增進學生的語文能力、此外也能提升學校在國際舞台上的地位。

四、攜回資料：

WICON 2005 會議資訊：記載本次研討會的時間、地點、會議流程、報告人員、以及報告題目等與研討會相關的會議資訊。

附件一：WICON2005網址 (http://www.wicon.org)

附件二：飛機行程(為當地時間)

7/9	22:00	台北	出發
7/10	06:00	維也納	抵達
7/10	10:20	維也納	出發
7/10	11:10	布達佩斯	抵達
7/20	09:00	布拉格	出發
7/20	10:05	維也納	抵達
7/20	11:05	維也納	出發
7/21	06:30	台北	抵達

II. 發表論文全文

論文名稱：Mobile IP and Ad Hoc Networks: An Integration and Implementation Experience

作者：Y.-C. Tseng, C.-C. Shen, and W.-T. Chen

發表情況: ***IEEE Computer***, Vol. 36, No. 5, May 2003, pp. 48-55. (SCI, EI)

Mobile IP and Ad Hoc Networks: An Integration and Implementation Experience

Yu-Chee Tseng[†], Chia-Ching Shen[†], and Wen-Tsuen Chen[‡]
[†]Department of Computer Science and Information Engineering
National Chiao Tung University
Hsin-Chu, 30050, Taiwan
Email: {yctseng, jcsheen}@csie.nctu.edu.tw
[‡]Department of Computer Science
National Tsing Hua University
Hsin-Chu, 30050, Taiwan
Email: wtchen@cs.nthu.edu.tw

***Abstract*— *Mobile IP* has been widely accepted as a standard to support IP mobility in a wireless Internet environment to keep a session connected when a mobile host roams from subnet to subnet. Another emerging wireless network architecture that is gaining more and more popularity is the *mobile ad hoc network (MANET)*, which can be flexibly deployed in almost any environment without the need of infrastructure base stations. In order to move to an all-IP environment, there seems to be a growing demand to integrate these two architectures together.**

Typically, mobile hosts are served by access points that can connect to them directly (in one hop). In this paper, we propose to extend access points to multiple MANETs, each as a subnet of the Internet, and discuss how to support Mobile IP in such environment. Such integration is beneficial to both societies. From Mobile IP's prospective, Foreign Agents' service areas are not limited to hosts within a single (wireless) hop any more. From MANET's prospective, mobile hosts can immediately enjoy tremendous services already existing on the Internet through Mobile IP. This article reports our integration and implementation experience based on IEEE 802.11b wireless LANs. Issues such as overlapping of MANETs, dynamic adjustment of mobile agents' service coverages, support of local broadcast and various communication scenarios are addressed. Discussion also covers required adjustments of Mobile IP to support such architecture.

***Index Terms*— ad hoc network, mobile computing, Mobile IP, mobility management, routing, wireless Internet.**

I. Introduction

Wireless communications and mobile computing are gaining more popularity in recent years. Wireless communication devices have become standard features in most portable computing devices, such as laptops, PDAs, and handsets. People are becoming used to carrying computers while traveling around to enjoy the tremendous services on the Internet. Ubiquitous computing has added a new feature, *mobility*, to the world of computing and communications.

We have observed two strong growths of interests related to this trend. The first one is Mobile IP [15], which supports mobile hosts roaming from subnet to subnet without need of changing IP addresses. Mobile (home and foreign) agents are used to support seamless handoffs. The next generation IPv6 will include features of Mobile IP as inherent functionality. Another emerging wireless network architecture is the *mobile ad hoc network (MANET)*, which can be flexibly and conveniently deployed in almost any environment without the need of infrastructure base stations. MANETs have received intensive attentions recently [6], [11], [18], [20]. In the literature, most works are based on IEEE 802.11-like network interface cards to build a MANET. The recently proposed wireless sensor networks also have a similar architecture to the ad hoc networks.

In the trend of moving to an all-IP environment, there seems to be growing demand to integrate these two architectures together. In this paper, we propose to extend the typical *wireless access points* to multiple MANETs, each as a subnet of the Internet, and discuss how to support Mobile IP in such environment. Such integration is beneficial to both societies. From Mobile IP's prospective, Foreign Agents' service areas are not limited to hosts within a single wireless hop any more. From MANET's prospective, mobile hosts can immediately enjoy tremendous services already existing on the Internet without worrying about disconnection due to mobility. With such combination,

macro mobility is supported by the former, while micro mobility is supported by the latter.

This article reports our integration and implementation experience based on IEEE 802.11b wireless LANs. Various routing scenarios involving Mobile IP and MANET are discussed. One fuzzy area is the possibility of some MANETs overlapping with each other to form a larger MANET (which is sometimes inevitable), making the service boundaries of home/foreign agents vague. The dynamics of MANETs also necessitates redefining the service coverage of AGENT_ADVERTISEMENT and AGENT_SOLICITATION messages in Mobile IP so as to adapt to constant topology changes of MANETs. The support of local broadcast and various communication scenarios and issues like TTL, ARP, and registration, are addressed. Discussions also cover required adjustments of Mobile IP to support such architecture.

II. Preliminaries

A. Mobile Ad Hoc Networks (MANETs)

A MANET is a network consisting of a set of mobile hosts which may communicate with one another and roam around at their will. A routing path may consist of a sequence of wireless links without passing base stations (i.e., in a *multi-hop* manner). This requires each mobile host to serve as a router. Applications of MANETs occur in situations like battlefields, outdoor assemblies, and emergency rescues, where base stations or fixed network infrastructures are not available, but networks need to be deployed immediately.

Extensive efforts have been devoted to the routing issues on MANET. Routing protocols can be classified as *proactive* and *reactive*. A proactive protocol (such as the DSDV protocol [12]) constantly updates routing information so as to maintain a (close to) global view on the network topology. On the contrary, a reactive protocol searches for a path in an on-demand manner. This may be less costly than a proactive protocol when host mobility is high. Representative reactive protocols include DSR [5], ZRP [3], CBR [4], and AODV [11]. A review of unicast routing protocols for MANET is in [18]. Multicast is studied in [1], [2]. Broadcasting issues are studied in [7], [8], [9].

B. Mobile IP

The Mobile IP is defined by IETF to support IP mobility [15]. Transparent to TCP and UDP, it allows sessions to remain connected when mobile hosts roam from subnet to subnet without the need of changing IP addresses.

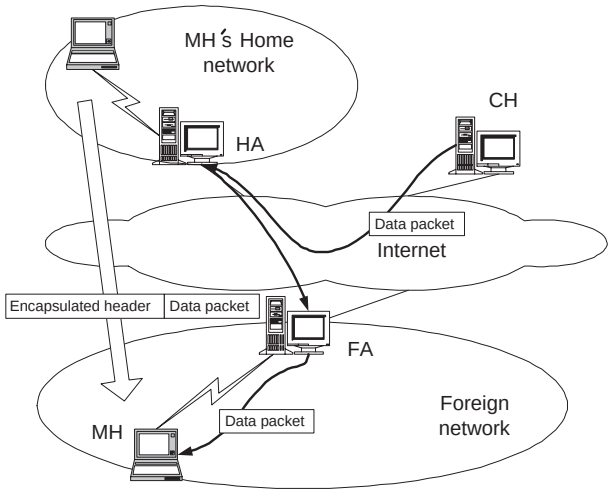


Fig. 1. The transmission scenario of Mobile IP.

Four roles are defined in Mobile IP: mobile host (MH), corresponding host (CH), home agent (HA), and foreign agent (FA), as illustrated in Fig. 1. A MH is a host or a router that may change its point of attachment from subnet to subnet. When a CH, which is a host in the Internet, sends an IP datagram to a MH, it will be delivered to the MH's home network. When the MH is away from home, the datagram will be *tunneled* to the foreign network. The HA will encapsulate the datagram with an IP header carrying FA's IP address or the MH's co-located CoA (care-of-address). In case of using FA's address, the FA should de-capsulate the datagram and forward it to MH. CoA can be dynamically obtained as a temporary address, through such as DHCP address configuration procedure. If the co-located CoA is used, the MH itself serves as the endpoint of the tunnel and performs decapsulation locally. In our implementation, we follow the former solution.

HA and FA need to advertise their services by periodically sending AGENT_ADVERTISEMENT. A MH unaware of any local mobile agent may inquire by sending AGENT_SOLICITATION. From time to time, MH needs to register with its HA its current CoA. HA keeps track of the mapping between each residential MH's permanent address and its CoA in a location dictionary (LD). Further extensions of Mobile IP also exist, such as smooth handoff [14] and extension for IPv6 [16].

C. Related Work

Cellular IP [21] and HAWAII [17] are two Internet protocols to support IP mobility. Cellular IP separates *macro* mobility from *micro* mobility. Originally, mobile IP is to support macro mobility. To reduce frequent registrations to HA as a MH is roaming around, Cellular IP adopts a hierarchical approach. A FA can provide ser-

vices to multiple base stations. As long as a MH is covered by base stations belonging to the same FA, no re-registration is required. In this case, handoff delay may be significantly reduced. As such, micro mobility to support seamless handoff is achieved. HAWAII (Handoff-Aware Wireless Access Internet Infrastructure) adopts a domain-based approach to support mobility. Base stations can be connected as a tree. It uses specialized path setup schemes which install host-based forwarding entries in specific routers to support intra-domain routing. This results in the same advantage of supporting micro mobility and fast handoff as in Cellular IP. Compared to Cellular IP, HAWAII breaks the tie between gateway and FA, and thus is more tolerant to failure of gateways and simplifies the design of gateways.

References [10], [13] also address the construction of MANET by providing continuous Internet access based on Mobile IP. How to extend Mobile IP to allow MHs to use CoA even if they are more than one hop away from FAs is addressed. The conflict between the management of routing tables in Mobile IP and MANET is resolved. Implementations on both OS/2 and AIX are reported. In particular, two separate daemons are used by Mobile IP and MANET. To coordinate these two daemons, a route manager is used to control the system's routing table.

Compared to [21], [17], our network architecture does not rely on hierarchical (wireline) routers. Instead, following the basic idea of ad hoc networks, mobile hosts are used as routers to extend the coverage of FAs. Thus, our framework also support micro mobility as well as macro mobility. While Cellular IP and HAWAII restrict mobile hosts be resident in one (wireless) hop from the base stations, our framework allows mobile hosts in multiple (wireless) hops from the base station. Also, our work is compatible with current design of MANET, thus easily extending MANET for IP mobility support. In addition, since the topologies of MANETs may change dynamically, the service ranges of FAs may also change accordingly. An advantage is a higher fault-tolerant capability — if one FA crashes, a mobile host may rely on MANET's routing capability to connect to neighboring FAs. Compared to [10], [13], which considers only one single MANET, we consider the existence of multiple MANETs in a vicinity area. Specified by the local FA, the service range of a FA may be dynamically adjusted. Negotiation between mobile agents and mobile hosts on FAs' service ranges is possible. This may greatly improve the flexibility of MANETs and reduce the service overhead of mobile agents. Also, MANETs may overlap with each other, and thus can support each other and offer a higher fault-tolerant capability in terms of Internet access. Further, to enjoy the flexibility of MANETs, direct communications between hosts under two FA's coverages, through MANETs' links, are possible.

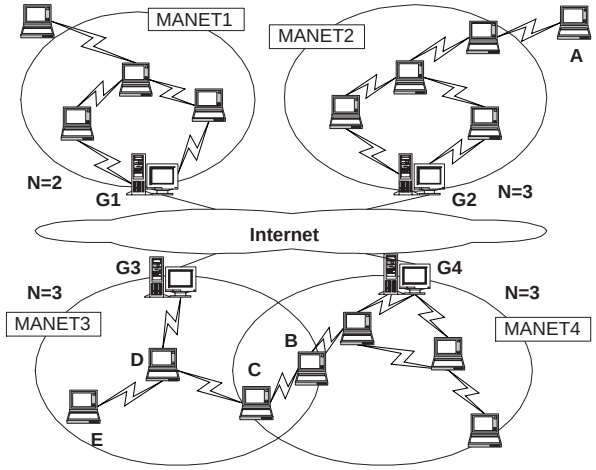


Fig. 2. The proposed network architecture, which extends each access point to a MANET.

III. Network Architecture and Communication Scenarios

A. Network Model

We consider the network consisting of multiple MANETs, each of which has a point of attachment to the backbone Internet. The host connecting a MANET to the Internet is called the *gateway*. We use gateways to define the ranges of MANETs. Each gateway has two network interface cards (NICs), one wireless and one wireline. Gateway hosts have no mobility since they have fixed links. However, non-gateway hosts can roam around freely, and thus the definition of MANETs actually changes by time. Several MANETs are shown in Fig. 2. Gateways are responsible of intertworking MANETs with the Internet by forwarding/relaying packets. To support Mobile IP, each gateway also serves as the FA in its local MANET. So it should periodically broadcast AGENT_ADVERTISEMENT messages to announce its service to members of its MANET. (In our discussion, we may interchangeably use gateway and FA according to the context.)

Since members of MANETs are mobile, it is likely that a MANET is partitioned into multiple MANETs, or some MANETs may join and overlap with each other. In such cases, the boundaries between MANETs become vague, making the service ranges of FAs unclear. We propose to define the service ranges of gateways by associating with each gateway a parameter N. Any mobile host within N wireless hops from the gateway can join the MANET served by the gateway. This is

achieved by specifying a TTL = N in each gateway's AGENT_ADVERTISEMENT. For example, in Fig. 2, host A, though connected to MANET2, can not be a part of the network.

In case that a host is within the service ranges of multiple gateways, it can choose the shortest-distance one as its default gateway. By so doing, the boundaries of subnets are clearly defined even if MANETs are overlapping with each another. For example, in Fig. 2, host C belongs to MANET3, while host B belongs to MANET4, and their HAs will tunnel IP datagrams accordingly from the proper gateways. Also, note that each gateway can define its own N independently based on its willingness/capability to provide services.

When MHs move around, it is even possible that a MH is disconnected from its gateway, but still remains connected to other MANETs. For instance, in Fig. 2, if the link between G3 and D breaks, hosts D's and E's connections to the Internet will become broken because they are beyond the service range of G4. To dynamically adjust a gateway's service range, we propose that a MH, on missing AGENT_ADVERTISEMENT for a certain period, may broadcast or multicast an AGENT_SOLICITATION message with a TTL = N'. The value of N' can be gradually increased to avoid the *broadcast storm* problem [7] caused by flooding. The solicitation can be heard if $N' \geq N$ and the MANET is connected. On receiving the AGENT_SOLICITATION, the gateway may decide, based on its willingness, whether to increase its N or not. In the above example, if host E's AGENT_SOLICITATION has an $N' = 5$, G4 will receive the request, and may increase its service range to cover D and E.

B. Some Communication Scenarios

Based on the above network architecture, several different communication scenarios may exist. In the following, we discuss the possible combinations and the corresponding routings. In the discussion, we assume that routing in MANETs is supported by DSDV (however, any proper routing protocol for MANETs is applicable).

- *Intra-MANET communication:* The communications are supported by DSDV. In the DSDV protocol, hosts will exchange routing information periodically and compute the next hop to reach the destination with the least metric (such as hop count). Proper route entries will be written into the kernel routing table by system calls. So whenever a route entry leading to the destination is found, the packet is directly forwarded to the next hop. The transmission from A to B in Fig. 3 fits into this category.

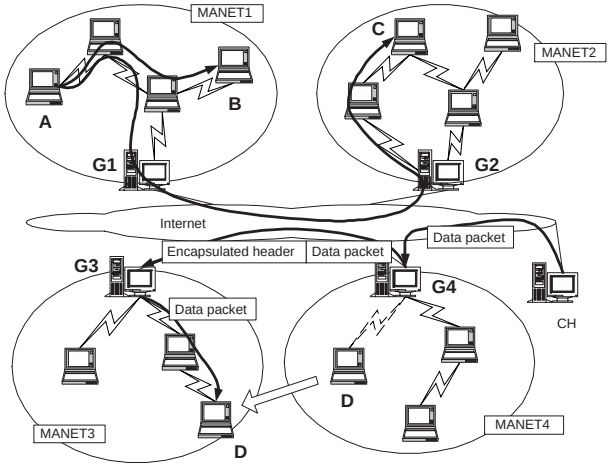


Fig. 3. Intra- and inter-MANET routing scenarios.

- *Inter-MANET communication (direct):* For any packet whose destination is not listed in the kernel routing table, it will be forwarded to the gateway of the local MANET. The gateway will then forward the packet to the Internet. The transmission from A to C in Fig. 3 is such an example. Packets travel on MANET1 based on DSDV, then on the Internet to G2 based on IP routing, and then on MANET2 to B by DSDV again.
- *Inter-MANET communication (with Mobile IP):* A MH may roam away from its home network. In this case, Mobile IP will be involved to forward packets between MANETs. In the transmission from CH to D in Fig. 3, packets will arrive at G4 by IP routing. These packets will be encapsulated and tunneled, by Mobile IP, to G3, which will then forward them to D by DSDV.
  To support such scenario, MHs have to monitor any existing AGENT_ADVERTISEMENT. Registration and deregistration procedures in Mobile IP should be followed. The routing of these packets will be supported by DSDV. HAs should maintain the current locations of its MHs. FAs should maintain the visiting MHs in their MANETs. HAs should execute proxy ARP for roaming MHs.
- *Inter-MANET communication in overlaid MANETs (direct):* When two MANETs overlay with each other, a MH may be aware of a route to another MH that belongs to a neighboring MANET. This is made possible by the frequent exchange of routing information by DSDV. In this case, directly routing between these MANETs is allowed. For example, in Fig. 4, since A has a route entry leading to B, direct inter-MANET transmission is possible.
  To support such scenario, we propose to associate

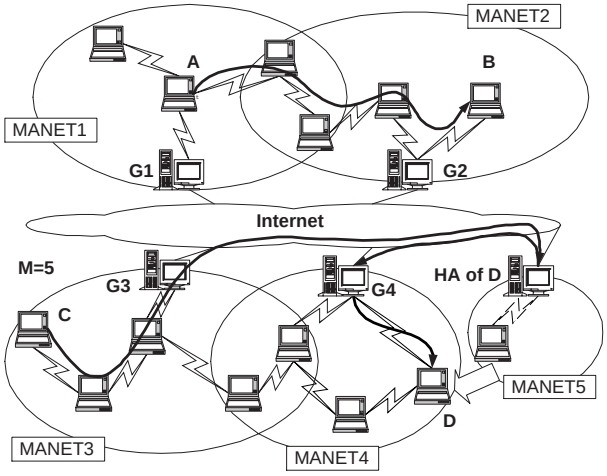


Fig. 4. Inter-MANET routing scenarios in overlaid MANETs.

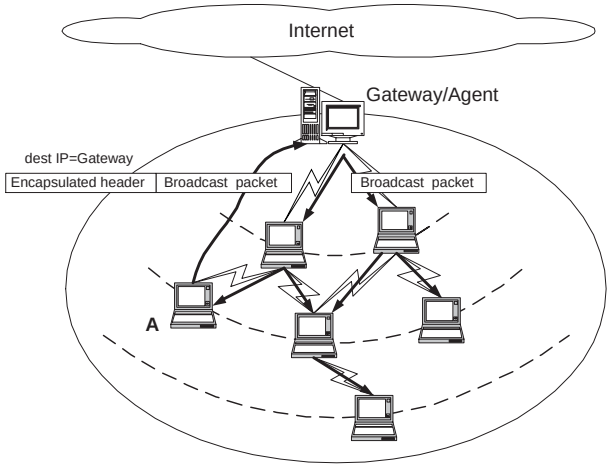


Fig. 5. Routing scenario of broadcasting.

with DSDV a parameter M, which reflects the service range of DSDV. I.e., a MH always collects/propagates routing information for MHs that are within M wireless hops from itself. As a result, hosts in different, but connected, MANETs may communicate with each other directly. The routing, tunneling, and encapsulating overheads can be reduced by such optimization. Note that it is mandatory that $M \geq N$ so that routing information leading to the local gateway is always known by a MH.

- *Inter-MANET communication in overlaid MANETs (with Mobile IP):* Contrary to the above scenario, when two MHs are resident in connected MANETs but away by more than M hops, their communications should be routed through the Internet. In the transmission from C to D in Fig. 4, assuming $M = 5$, host C will not be aware of any route (although existing) leading to D. In this case, its IP datagrams will be forwarded to the local gateway G3 (by DSDV), which will in turn forward the datagrams to D's HA (by IP routing), which will encapsulate the datagrams to D's current FA (by Mobile IP), which will forward the datagrams to D (by DSDV). As can be expected, the values of N and M should be properly tuned to reduce overheads and improve efficiency, which may be directed to an interesting research problem.
- *Broadcast:* Broadcasting is useful in many circumstances. In wireline communication, the scope of broadcast is well defined — a broadcast message is typically flooded to the physical range covered by a subnet. In wireless communication, due to the radio transmission property, the range that should be covered by a broadcast is usually not well defined. This is particularly true for ad hoc networks, where each MH has its own radio coverage. Note that if we directly adopt a TTL value to a broadcast packet, each mobile host's broadcast range will be distinct (depending on its current location).
  We propose to define the coverage range of a broadcast as the service range provided by the local gateway where the broadcast is issued. As a result, the range of a subnet matches with the range of a MANET. The detailed routing is conducted as follows. When a MH wants to send a broadcast datagram, it first encapsulates the packet as a unicast by identifying the gateway as the destination host. When the unicast packet is tunneled to the gateway, the gateway will decapsulate the packet and find that it is a broadcast packet. Then the gateway broadcasts this packet on behalf of the original source with a TTL = N. For example, Fig. 5 shows how A's broadcast datagram flows. Note that to detect duplicate broadcasts, each MH should maintain a list of broadcast IDs that it has received recently. The "source IP address" and the "IP identification" fields in the IP header can serve as a unique identity.

IV. Integration and Implementation Issues

Based on the proposed network architecture, we have developed a prototype which integrates Mobile IP and MANET together. Below, we report our integration and implementation experiences. The following modifications are required:

- *TTL in IP Packets:* Each IP datagram has a TTL field to control its lifetime on the Internet. In the original Mobile IP, each AGENT_ADVERTISEMENT should have TTL = 1. We dynamically tune TTL to control our AGENT_ADVERTISEMENT, AGEN_SOLICITATION, and broadcast packets.

- *Routing inside MANET:* Our implementation is based on the DSDV protocol [12]. Each host maintains a *forwarding table* containing a list of all available destinations together with the next hop leading to each destination. This forwarding table is used to update the kernel's routing table. Control packets are used to exchanges distance vectors between neighboring hosts. Each route entry is tagged with a sequence number originated by the destination host. These control packets have a destination address of 224.0.0.1 (all-systems multicast address) with TTL = 1 because they need not to be rebroadcast.
  In our protocol, several modifications are required. First, since we allow MANETs to overlap with each other, to avoid the amount of information being exchanged becoming too large, only route information that is within M hops is registered and propagated. Second, recall that M should be at least as large as N used by the local gateway; the value of N should be broadcast together with the gateway's control packets. Third, each gateway should identify itself as a gateway by associating a gateway bit in its control packets. Each MH should set its *default router* to be the host that leads to the gateway host with the least metric. Forth, if a MH also has a CoA, it has to advertise through DSDV's control packets its original IP address as well as its CoA. This is similar to having two IP addresses by a host. This can be easily achieved by providing two entries in the control packets. So the MH can be reached both by its permanent IP address directly (in the MANET sense) and by its CoA (in the Mobile IP sense).
- *Agent Advertisement:* In the original Mobile IP, AGENT_ADVERTISMENT has TTL = 1. Due to the multi-hop nature of MANETs, the TTL should be set to N, and the value is decreased by one each time it is rebroadcast. No rebroadcast is needed when TTL = 0. The destination field should be 255.255.255.255.
- *Agent Solicitation:* A MH can multicast AGENT_SOLICITATION to find a nearby mobile agent. The destination field should be the all-routers multicast address 224.0.0.2. We recommend that its TTL field be doubled each time when the solicitation process fails. Intuitively, doubling the TTL can reach approximately four times the hosts that can be reached in the previous round. In addition, since the value of TTL will be decreased as the packet travels more hops, the original TTL value should be recorded in the packet's payload so that when the gateway receives the packet, it can recover its distance to the requesting MH. By comparing this value to N, the gateway can decide whether it needs to enlarge its service range N or not.
- *ARP:* In the original Mobile IP, ARP should be disabled when a MH visits a foreign network. The MAC-to-IP address mapping is registered when AGENT_ADVERTISEMENT is received. Under our network architecture, to allow peer-to-peer communication inside a MANET, ARP still needs to be enabled in foreign networks. ARP requests and replies should be sent as usual. Since many nomadic hosts may exist in a MANET, the concept of subnet mask should not be used and packets of any destination should be relayed.
- *Broadcast:* We design a broadcast daemon to support the scenario in Fig. 5. Whenever a broadcast datagram with destination address = 255.255.255.255 and TTL = 1, source address = myself is intercepted, the daemon will encapsulate this packet as a unicast destined for the local gateway. On receiving the packet, the gateway will decapsulate and broadcast it with TTL = N. However, one potential problem is that the broadcast datagram may loop back to the source host. To resolve this problem, the broadcast daemon should also record the recent broadcast datagrams that it has encapsulated recently.
- *Destination address and TTL:* Recall that the M used by MANETs should be at least as large as N used by Mobile IP. By adjusting M and N, we can control the amount of traffic flowing into and out of a MANET. We recommend that $M = 2N$, which guarantees that intra-MANET communication can always be done directly without encapsulation (in the worst case, an intra-MANET packet may need to be sent to the gateway first and then forwarded to the destination). Also, inter-MANET communication between nearby MANETs are likely to be done without going through Mobile IP (and thus the encapsulation procedure).
- *Configuration of IEEE 802.11b NICs:* In our implementation, all wireless NICs are set to the peer-to-peer (ad hoc) mode. All mobile hosts use the same ESSID and the same channel number so as to communicate with each other. To increase channel reuse (and thus communication bandwidth), it is possible that FAs can use different channels. In most current products, a NIC will automatically scan the available channels only when its current connection is broken (i.e., active scanning). So a host may not be able to discover all its neighbors if they are operating at different channels. Under our framework, the network should function correctly, except that some

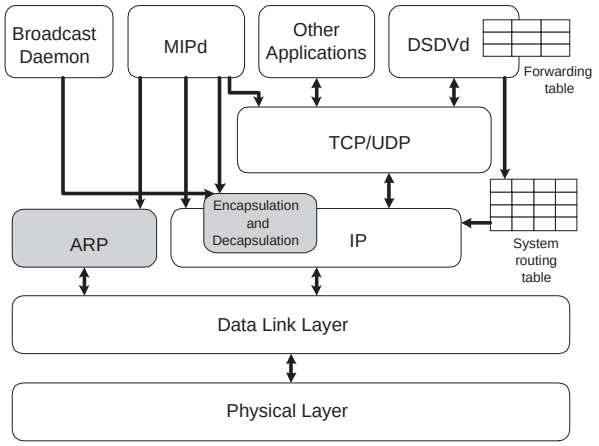


Fig. 6. System architecture of our implementation.

routes may not exist even if some hosts are physically neighbors.

Our system is developed based on Linux Redhat 2.2.16. Two daemons, namely *DSDVd* and *MIPd*, are implemented. Conceptually, both daemons are network-layer programs. However, they are actually implemented on the application layer and interact with system kernel through socket interfaces. The concept is shown in Fig. 6. *DSDVd* periodically multicast UDP packets to help maintain hosts' forwarding tables. Proper entries from the forwarding table are written into the kernel's routing table by system calls. In *MIPd*, we use RAW sockets for advertisement, encapsulation, and decapsulation, and normal sockets for registration. Proxy ARP is done by UNIX system calls. Also, the IP forwarding option at each mobile host must be turned on.

V. Conclusions

In this paper, we have investigated the related issues to integrate MANETs with Mobile IP. Hence, traditional access points can directly enjoy the flexibility of MANETs and widen their coverage ranges. In view of the worldwide explosive deployments of IEEE 802.11-based access points, such extension would help make our dream of ubiquitous broadband wireless access come true. Details of our prototyping and implementation experiences are reported. Some performance test results are available in [19]. The discussion in this paper is based on IP version 4; it will be interesting to investigate the related issues on IP version 6.

Acknowledgements

This work is co-sponsored by the MOE Program for Promoting Academic Excellence of Universities under grant numbers A-91-H-FA07-1-4 and 89-E-FA04-1-4.

References

[1] C.-C. Chiang, M. Gerla, and L. Zhang. Forwarding group multicast protocol (fgmp) for multihop, mobile wireless networks. *ACM-Baltzer Journal of Cluster Computing*, 1(2), 1998.

[2] J. J. Garcia-Luna-Aceves and E. L. Madruga. A multicast routing protocol for ad-hoc networks. In *INFOCOM*, 1999.

[3] Z. Haas and M. Pearlman. *ZRP: A Hybrid Framework for Routing in Ad Hoc Networks (a book chapter in Ad Hoc Networking, Ed. C. E. Perkins, Chapter 7)*. Addison-Wesley, 2000.

[4] M. Jiang, J. Li, and Y. Tay. Cluster based routing protocol (CBRP) functional specification (internet draft), 1998.

[5] D. B. Johnson, D. Maltz, and J. Broch. *DSR: The Dynamic Source Routing Protocol for Multihop Wireless Ad Hoc Networks (a book chapter in Ad Hoc Networking, Ed. C. E. Perkins, Chapter 5)*. Addison-Wesley, 2000.

[6] J. Macker and M. Corson. Mobile ad hoc networking and the IETF. *ACM Mobile Computing and Communications Review*, 2(1):9–14, Oct. 1998.

[7] S.-Y. Ni, Y.-C. Tseng, Y.-S. Chen, and J.-P. Sheu. The Broadcast Storm Problem in a Mobile Ad Hoc Network. In *MobiCom*, pages 151–162, 1999.

[8] E. Pagani and G. P. Rossi. Reliable broadcast in mobile multihop packet networks. In *MobiCom*, pages 34–42, 1997.

[9] E. Pagani and G. P. Rossi. Providing reliable and fault tolerant broadcast delivery in mobile ad-hoc networks. *Mobile Networks and Applications*, 4:175–192, 1999.

[10] C. Perkins. Mobile-IP, ad-hoc networking, and nomadicity. In *COMPSAC*, 1996.

[11] C. Perkins. Ad-hoc on-demand distance vector routing. In *MILCOM*, 1997.

[12] C. Perkins and P. Bhagwat. Highly Dynamic Destination-Sequenced Distance-Vector (DSDV) Routing for Mobile Computers. In *ACM SIGCOMM Symposium on Communications, Architectures and Protocols*, pages 234–244, Sep. 1994.

[13] C. Perkins and H. Lei. Ad hoc networking with mobile IP. In *EPMCC*, 1997.

[14] C. Perkins and K. Wang. Optimized smooth handoffs in mobile IP. In *IEEE Symp. on Computers and Communications*, 1999.

[15] C. E. Perkins. *Mobile IP Design Principles and Practices*. Addison-Wesley, 1997.

[16] C. E. Perkins and D. B. Johnson. Mobility support in IPv6. In *Mobile Computing and Networking*, pages 27–37, 1996.

[17] R. Ramjee, T. F. L. Porta, S. Thuel, K. Varadhan, and S. Y. Wang. HAWAII: A domain-based approach for supporting mobility in wide-area wireless networks. In *ICNP*, pages 283–292, 1999.

[18] E. Royer and C. Toh. A review of current routing protocols for ad-hoc mobile wireless networks. *IEEE Personal Communications*, Apr. 1999.

[19] C.-C. Sheng. Mobile IP and ad hoc networks: An integration and implementation experience. Master Thesis, National Chiao Tung University, Dept. of Computer Science and Information Engineering, 2002.

[20] Y.-C. Tseng, S.-L. Wu, W.-H. Liao, and C.-M. Chao. Location awareness in ad hoc wireless mobile networks. *IEEE Computer*, 34(6):46–52, June 2001.

[21] A. G. Valko. Cellular IP - a new approach to internet host mobility. In *ACM Computer Communication Review*, 1999.

論文名稱：Collision Analysis for a Multi-Bluetooth Picocells Environment

作者：T.-Y. Lin and Y.-C. Tseng

發表情況: ***IEEE Communications Letters***, Vol. 7, No. 10, Oct. 2003, pp. 475-477. (SCI, EI)

Collision Analysis for a Multi-Bluetooth Picocells Environment

Ting-Yu Lin and Yu-Chee Tseng
Department of Computer Science and Information Engineering
National Chiao-Tung University, Hsin-Chu, 300, Taiwan

Abstract—
Operating in the unlicensed 2.4GHz ISM band, a Bluetooth piconet will inevitably encounter the interference problem from other piconets. With a special channel model and packet formats, one research issue is how to predict the packet collision effect in a multi-piconet environment. In an earlier work [1], this problem is studied, but the result is still very limited, due to the assumptions that packets must be single-slot ones and that time slots of each piconet must be fully occupied by packets. A more general analysis is presented in this work by eliminating these constraints.

I. Introduction

As a promising WPAN technology, Bluetooths are expected to be used in many applications, such as wireless earphones, keyboards, wireless access points, etc [3]. Operating in the unlicensed 2.4GHz ISM band, multiple Bluetooth piconets are likely to coexist in a physical environment. With a frequency-hopping radio and without coordination among piconets, transmissions from different piconets will inevitably encounter the collision problem. In a previous work [1], the author investigates the co-channel interference between Bluetooth piconets and derives an upper bound on packet error rate. The analysis in [1] has two limitations. First, all packets are assumed to be single-slot ones. Second, it is assumed that each piconet is fully loaded, in the sense that packets are sent in a back-to-back manner. These constraints greatly limit the applicability of the result in [1].

Also focusing on the same problem, this paper derives a more general analysis model where all packet types (1-slot, 3-slot, and 5-slot) can coexist in the network, and the system is not necessarily fully-loaded. The latter is achieved by modeling idle slots as individual single slots with *no* traffic load. So the result greatly relax the constraints in [1].

II. Problem Statement

Bluetooth is a master-driven, time-division duplex (TDD), frequency-hopping (FH) wireless radio system [2]. The smallest networking unit is a *piconet*, which consists of one master and no more than seven active slaves. Each picocell channel is represented by a pseudo-random hopping sequence comprised of 79 or 23 frequencies. In Bluetooth, the hopping sequence is determined by the master's ID and clock value. The channel is divided into time slots, each corresponding to one random frequency. In the following discussion, we assume 79 frequencies.

In each piconet, the master and slaves take turns to exchange packets. While the master only transmits in even-numbered slots, slaves must reply in odd-numbered slots. Three packet sizes are available: 1-slot, 3-slot, and 5-slot. For a multi-slot packet, its frequency is fully determined by the first slot and remains unchanged throughout.

We consider N piconets coexisting in a physically closed environment. Since no coordination is possible between piconets, each piconet has $N-1$ potential competitors. In any time instance, if two piconets transmit with the same frequency, the corresponding two packets are considered damaged. Our goal is to derive an analytic model to evaluate the impact of collisions in such a multi-piconet environment.

We assume a uniform traffic in each piconet, and let λ_1, λ_3, and λ_5 be the arrival rates of 1-, 3-, and 5-slot packets per slot, respectively, to a piconet. Note that for a multi-slot packet, only the header slot counts as arrival. It is easy to see that $\lambda_1 + 3\lambda_3 + 5\lambda_5 \leq 1$. Further, we can regard the remaining vacant slots as "dummy" single-slot packets. Thus, the arrival rate of such dummy (1-slot) packets is $\lambda_0 = 1-(\lambda_1+3\lambda_3+5\lambda_5)$.

III. Collision Analysis in a Multi-Piconet Environment

Let's consider a piconet X and another competitor piconet Y, which is regarded as the unique source of interference to X. With the interference from Y, we first derive the success probability $P_S(i)$ of i-slot packets in X, where $i = 1, 3, 5$. We start by introducing the concept of "slot delimiter." Consider any slot in X. One or two slot delimiters in Y may cross X's slot. However, since we are considering continuous probability, the possibility of two crossing slot delimiters can be ignored, and thus we will deal with one crossing delimiter in the rest of the discussion. For example, for a 1-slot packet in X, it succeeds only if there is no interference from the two slots before and after the delimiter, so the success probability of X's packet could be 1, $\frac{78}{79}$, or $(\frac{78}{79})^2$, depending on whether Y transmits or not. Below, we denote the constant factor 78/79 by P_0.

Next, we further elaborate on slot delimiters. Depending on what packet(s) is divided by it, a delimiter is classified into ten types (refer to Fig. 1):

- B_1, B_2, B_5: the beginning of a 1-, 3-, and 5-slot packet, respectively.
- B_3, B_4: the beginnings of the second and third slots of a 3-slot packet, respectively.

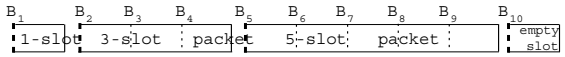


Fig. 1. Illustration of slot delimiters.

- B_6, B_7, B_8, B_9: the beginnings of the second, third, fourth, and fifth slots of a 5-slot packet, respectively.
- B_{10}: the beginning of a dummy slot.

It is easy to see that the rate of B_1 is λ_1 per slot; the rate of each of B_2, B_3, and B_4 is λ_3; the rate of each of B_5, B_6, B_7, B_8, and B_9 is λ_5; and the rate of B_{10} is λ_0. For ease of presentation, we denote the arrival rate of B_j by $\lambda(B_j)$, $j = 1..10$. Given any B_j, we also define $g(j)$ to be the number of slots that follows delimiter B_j and belong to the same packet. For example, $g(1) = 1$, $g(3) = 2$, $g(7) = 3$, and $g(10) = 1$.

Intuitively, when a packet in X is crossed by a delimiter of type $B_1/B_2/B_5$ in Y, there may exist two packets (of different frequencies) in both sides of the delimiter in Y which are potential sources of interference to X's packet. On the other hand, when the delimiter is of the other types, the interference source reduces to one.

Next, we formulate the success probability $P_S(i)$ of an i-slot packet, $i = 1, 3, 5$, in X, given the interference source Y. Toward this goal, we first introduce another probability function.

Definition 1: Given any i-slot packet in piconet X and any interference source piconet Y, define $L(k), k < i$, to be the probability that the packet of X experiences no interference from Y starting from the delimiter of Y crossing the $(i-k+1)$-th slot of the packet to the end of the packet, under the condition that the aforementioned delimiter is of type $B_1/B_2/B_5/B_{10}$. For $k \leq 0$ (in which case the above definition is not applicable), $L(k) = 1$.

Intuitively, $L(k)$ is the success probability of the last k slots of X's packet excluding the part before the first delimiter of Y crossing these k slots, given the delimiter type constraint. With this definition, we can find $P_S(i)$ by repeatedly cutting off some slots from the head of X's packet, until there is no remaining slot. Specifically, we establish $P_S(i)$ by $L()$ as follows:

$$P_S(i) = \sum_{j=1}^{10} \lambda(B_j) \cdot f(j) \cdot L(i - g(j)), \qquad (1)$$

where

$$f(j) = \begin{cases} (1-\lambda_0) \cdot P_0^2 + \lambda_0 \cdot P_0 & \text{if } j = 1, 2, 5 \\ (1-\lambda_0) \cdot P_0 + \lambda_0 & \text{if } j = 10 \\ P_0 & \text{otherwise} \end{cases}.$$

In the equation, we consider each type $B_j, j = 1..10$, of the first delimiter in Y crossing X's packet. The corresponding probability is $\lambda(B_j)$. Function $f(j)$ gives the probability that the packet(s) of Y on both sides of the first delimiter B_j does (do) not interfere with X's packet. It remains to consider the success probability of the last $i - g(j)$ slots of X's packet, excluding the part before the first delimiter of Y crossing these $i-g(j)$ slots (which must be of delimiter type $B_1/B_2/B_5/B_{10}$). This is reflected by the last factor $L(i - g(j))$.

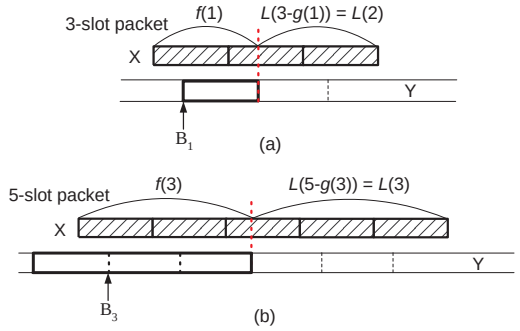


Fig. 2. Analysis of success probabilities for (a) 3-slot and (b) 5-slot packets.

For example, Fig. 2(a) illustrates a 3-slot packet in X. The first delimiter in Y crossing the 3-slot packet is of type B_1. The success probability of the first part in X is $f(1) = (1-\lambda_0) \cdot P_0^2 + \lambda_0 \cdot P_0$. Intuitively, if the packet of Y before the delimiter B_1 is a dummy packet (of probability λ_0), the success probability is simply P_0; otherwise, there are two packets which are potential interference sources, and the success probability is P_0^2. Then we can move on to consider the success probability of the remaining part of X after the second delimiter in Y, which is given by $L(2)$. Another example of a 5-slot packet is shown in Fig. 2(b). The first delimiter in Y crossing the 5-slot packet is of type B_3. So the success probability from the beginning of the packet up to the third delimiter in Y crossing the packet is $f(3)$. For the remaining part, the success probability is $L(3)$. So the success probability of the 5-slot packet is $f(3) \cdot L(3)$.

The remaining part of X's packet covered by $L(k)$ must start with a delimiter in Y of a restricted type of $B_1/B_2/B_5/B_{10}$. Since the packet in Y after the delimiter must be a complete packet, it can be solved recursively as follows ($k > 0$):

$$\begin{aligned} L(k) \;=\; & \frac{\lambda_0}{\lambda_0+\lambda_1+\lambda_3+\lambda_5} \cdot L(k - g(10)) + \\ & \frac{\lambda_1}{\lambda_0+\lambda_1+\lambda_3+\lambda_5} \cdot P_0 \cdot L(k - g(1)) + \\ & \frac{\lambda_3}{\lambda_0+\lambda_1+\lambda_3+\lambda_5} \cdot P_0 \cdot L(k - g(2)) + \\ & \frac{\lambda_5}{\lambda_0+\lambda_1+\lambda_3+\lambda_5} \cdot P_0 \cdot L(k - g(5)). \end{aligned} \qquad (2)$$

In each term, the first part is the probability of the corresponding packet type in Y. As to the boundary conditions, $L(k) = 1$, for $k \leq 0$.

Next, we consider an N-piconet environment. For each piconet X, there are $N-1$ piconets each serving as an interference source. Since these interferences are uncoordinated and independent, the success probability of an i-slot packet in X can be written as $P_S(i)^{N-1}$. So the network throughput of X is:

$$\begin{aligned} T \;=\; & \lambda_1 \cdot P_S(1)^{N-1} \cdot R_1 + 3 \cdot \lambda_3 \cdot P_S(3)^{N-1} \cdot R_3 + \\ & 5 \cdot \lambda_5 \cdot P_S(5)^{N-1} \cdot R_5, \end{aligned} \qquad (3)$$

where R_1, R_3, and R_5 are the data rates (bits/slot) of 1-, 3-, and 5-slot packets, respectively (for example, if

DH1/DH3/DH5 are used, $R_1 = 216$, $R_3 = 488$, and $R_5 = 542.4$). The aggregate network throughput of N piconets is $N \times T$.

IV. Simulation and Analysis Results

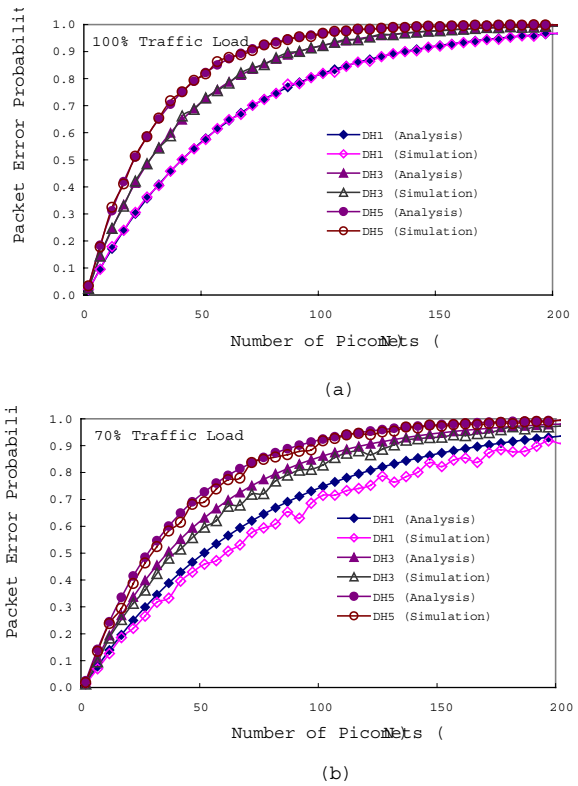


Fig. 3. Packet error probabilities under traffic loads of: (a) 100% and (b) 70%.

From the above discussion, it can be seen that the result in [1] is in fact a special case of our analysis when $\lambda_1 = 1$ and $\lambda_3 = \lambda_5 = 0$.

To verify our analysis, simulations are conducted. We investigate DH1/3/5 packets. Assuming $\lambda_1 = \lambda_3 = \lambda_5$, we inject traffic loads of 100% and 70% (the percentage of busy slots) to reflect heavy and medium loads, respectively. That is, $\lambda_1 + 3\lambda_3 + 5\lambda_5 = 1$ and 0.7. Fig. 3 plots the error probabilities of DH1/3/5 packets under different numbers of picocells. The packet error probability increases as the traffic load or the number of piconets grows. Small packets (DH1) suffer less collisions than large ones (DH5) due to shorter transmission durations. However, larger packets are much more bandwidth-efficient than smaller ones (e.g., a DH5 carries 542.4/216 times more bits per slot than a DH1 does). This observation leads us to conduct the next experiment by using network throughput as the metric.

Here we evaluate aggregate network throughput (i.e., $N \times T$). We show the case of 70% traffic load. In addition to equating $\lambda_1 = \lambda_3 = \lambda_5$, we also set $\lambda_1 : \lambda_3 : \lambda_5$ as $3 : 2 : 1$ to reflect the case of more shorter packets, and as $1 : 2 : 3$ to reflect the case of more longer packets. The results are shown in Fig. 4(a). The aggregate throughput saturates at a certain point as the number of piconets increases, and then drops sharply. Different from the earlier observation, the figure reveals that

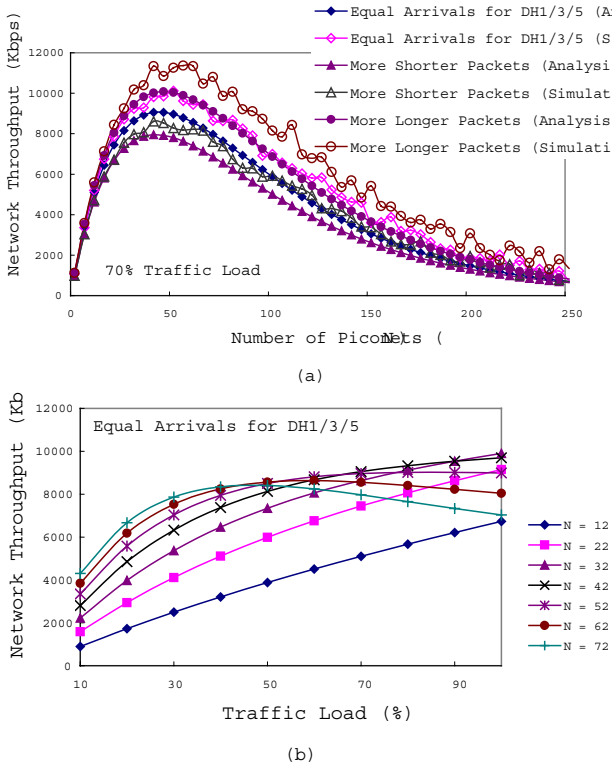


Fig. 4. Network throughput (a) under 70% traffic load, and (b) against traffic loads for various network sizes.

longer packets are more preferable in terms of throughput because the collision problem can be compensated by the benefit of bandwidth efficiency.

Finally, Fig. 4(b) plots the throughput against traffic loads by fixing the value of N. It indicates that throughput goes up steadily as traffic load increases when $N \leq 32$. However, for larger N's, throughputs saturate at certain points, due to more serious collisions. The results suggest that at most 42 piconets can be placed in a physical area.

References

[1] A. El-Hoiydi. Interference Between Bluetooth Networks - Upper Bound on the Packet Error Rate. *IEEE Communications Letters*, June 2001.

[2] http://www.bluetooth.com. Bluetooth SIG Specification v1.1. February 2001.

[3] K. Sairam, N. Gunasekaran, and S. Reddy. Bluetooth in Wireless Communication. *IEEE Communications Magazine*, June 2002.

論文名稱：An Adaptive Sniff Scheduling Scheme for Power Saving in Bluetooth

作者：T.-Y. Lin and Y.-C. Tseng

發表情況: ***IEEE Wireless Communications***, Vol. 9, No. 6, Dec. 2002, pp. 92-103. (SCI, EI)

An Adaptive Sniff Scheduling Scheme for Power Saving in Bluetooth

Ting-Yu Lin and Yu-Chee Tseng
Department of Computer Science and Information Engineering
National Chiao-Tung University
Hsin-Chu, 300, Taiwan
Email: {tylin, yctseng}@csie.nctu.edu.tw

Abstract

Bluetooth is expected to be an important basic constructing component for *Smart Homes*. In a smart home environment, a lot of devices will be portable and battery-operated, making *power saving* an essential issue. In this paper, we study the problem of managing the low-power *sniff mode* in Bluetooth, where a slave is allowed to be awake only periodically. One challenging problem is how to schedule each slave's sniffing period in a piconet so as to resolve the tradeoff between traffic requirement and power-saving requirement, to which we refer as the *sniff-scheduling problem*. We propose an adaptive protocol to dynamically adjust each slave's sniff parameters, with a goal of catching the varying, and even asymmetric, traffic patterns among the master and slaves. As compared to existing works, our work is unique in the following sense: First, our scheduling considers multiple slaves simultaneously. Existing works only consider one slave and different slaves are treated independently. Second, our scheduling is more accurate and dynamic in determining the sniff-related parameters based on slaves' traffic patterns. Most works are restricted to a naive exponential adjustment in sniff interval/sniff-attempt window. Third, our proposal includes the placement of sniff-attempt periods of sniffed slaves on the time axis when multiple slaves are involved. This issue is ignored by earlier works. Extensive simulation results are presented. Among the many observations, one interesting result is that with proper settings, our protocol can save significant power while achieving higher network throughput than a naive always-active, round-robin scheme.

Keywords

Bluetooth, mobile computing, personal-area networks (PANs), piconet, power saving, sniff mode, wireless communication, smart home.

I. Introduction

COMPUTING and communication anytime, anywhere is a global trend in today's development. Ubiquitous computing has been made possible by the advance of wireless communication technology and the availability of many light-weight, compact, portable computing devices. This area has attracted a lot of attention recently, and various types of network architectures have been proposed, such as wireless LAN, ad hoc network, sensor network, and personal-area network.

One emerging environment, which is gaining more and more attention, is the *Smart Home*. The basic idea behind Smart Homes is to provide various human-friendly services with the goal of facilitating human life. Typical home electronic appliances will not be considered clumsy any more. Instead, they are capable of coordinating with each other and adapting to surroundings. Such capabilities are achieved by equipping these appliances with embedded computing and communication devices. There are diverse aspects and technologies involving the development of Smart Homes. One promising technology supported by numerous organizations and companies is Bluetooth. With the design goal of compactness, low-cost, and low-power, Bluetooth is expected to be a promising basic constructing component for Smart Homes.

This paper focuses on Bluetooth [1], which is characterized by indoor, low-power, low-complexity, short-range radio wireless communications with a frequency-hopping, time-division-duplex channel model. A master-slaves configuration called a *piconet* is adopted. Readers can refer to [2], [3], [4] for more general details of the Bluetooth standard description.

Since low cost is one of the design goal of Bluetooth, a large number of wide-spread deployments of Bluetooth are expected. Within home environments, the deployment could consist of various portable devices. One essential issue for almost all kinds of portable devices is *power saving*. Mobile devices have to be supported by batteries, and without powers they become useless. Battery power is a limited resource, and it is expected that battery technology is not likely to progress as fast as computing and communication technologies do. Hence, how to lengthen the lifetime of batteries in portable devices is an important issue. Solutions for power saving can be generally categorized into several approaches.

- **Transmission Power Control:** In wireless communication, transmission power has strong impact on bit error rate, transmission rate, and inter-radio interference. These are typically contradicting factors. Power control to reduce interference for ad hoc networks is addressed in [5]. Dynamically adjusting transmission powers of mobile hosts in ad hoc networks to control network topology, or known as *topology control*, is addressed in [6]. How to increase network throughput by power adjustment for packet radio networks is addressed in [7].
- **Power-Aware Routing:** Power-aware routing protocols for ad hoc networks are discussed in [8], [9]. Several interesting power-related metrics are proposed in [9].
- **Management of Low-Power Modes:** More and more wireless devices can provide low-power modes. IEEE 802.11 has a power-saving mode in which a radio only needs to be awake periodically. HiperLAN allows the mobile host, which is in power-saving mode, to define its own active period [10]. As for active hosts, they can save power by turning off their equalizer according to the transmission bit rate. Bluetooth provides three low-power modes: *sniff*, *hold*, and *park* [1].

We study the management of low-power sniff mode in Bluetooth to conserve power, and thus this falls into the third category in the above classification. In the sniff mode, a slave's listening activity is reduced. Slaves only listen in specified time slots regularly spaced by sniff intervals. One challenging problem is how to schedule each slave's sniffing period in a piconet to balance the tradeoff between traffic requirement and power-saving requirement, to which we refer as the *sniff-scheduling problem*.

In this paper, an adaptive sniff scheduling protocol is proposed to periodically adjust the sniff parameters. An *Evaluator* is used by each master and its slaves to determine its traffic pattern and sniff-related parameters. A *Scheduler* is deployed in the master's side to schedule each slave's sniffing period. Since each slave's sniffing period can be regarded as an infinite sequence of time slots and multiple slaves are considered in this paper, we propose a concept called *Resource Pool (RP)* to manage the available/occupied time slots in the piconet. The master periodically checks the needs of its slaves by running the Evaluator, and allocates suitable slot resources from RP for them. On the other hand, slaves can exercise their own Evaluators and issue requests for slot resources as well. Two Scheduler policies are proposed: *LSIF (Longest Sniff Interval First)* and *SSIF (Shortest Sniff Interval First)*. Simulation results are presented to verify the effectiveness of our protocol.

As compared to existing works, our work is unique in the following sense: First, our scheduling scheme considers multiple slaves simultaneously. Existing works only consider one slave and different slaves are treated independently. Second, our scheduling is more accurate and dynamic in determining the sniff-related parameters based on slaves' traffic patterns. Most works are restricted to a naive exponential adjustment in sniff interval/sniff-attempt window. Third, our proposal includes the placement of sniff-attempt periods of sniffed slaves on the time axis when multiple slaves are involved. This issue is ignored by earlier works.

Related works include [11], [12], [13], [14]. In [12], [14], the polling priorities of slaves are determined based on their traffic loads; however, how to combine this with the sniff mode is not addressed. In [13], it is proposed to dynamically adjust the sniff parameters according to a slave's slot utilization. In [11], a learning function is proposed to determine the sniff interval. However, the sniff interval is adjusted in a per-interval basis and thus the overhead of control messages might be pretty high. Both [11], [13] suffer the problems that only one single slave is considered in an independent way, and that the placement of sniff-attempt windows is not addressed.

The rest of this paper is organized as follows. Preliminaries are in Section 2. Our sniff scheduling protocol is presented in Section 3. Section 4 proposes two policies for our Scheduler. Simulation results are provided in Section 5. Finally, Section 6 draws the conclusions.

II. Preliminaries

A. Bluetooth Technology Review

Bluetooth is a master-driven, time-division duplex, short-range radio wireless system. The smallest network unit is called a *piconet*, which has a mater-slaves configuration. A time slot in Bluetooth is 625μs. The master sends data to the slaves in even-numbered slots, while the slaves send data to the master in odd-numbered slots. A slave only transmits packets after the master polls or sends data to it.

According to the Bluetooth protocol stack [1], on top of RF is the Bluetooth Baseband, which controls the use of the radio. Four important operational modes are supported by the baseband: *active*, *sniff*, *hold*, and *park*. The active mode is most energy-consuming, where a bluetooth unit is turned on for most of the time. The sniff mode allows a slave to go to sleep and only wake up in specific time. In the hold mode, a slave can temporarily suspend supporting data packets on the current channel; the capacity can be made free for other things, such as scanning, paging, and inquiring. While in the hold mode, a unit can also attend other piconets. Prior to entering the hold mode, an agreement should be reached between the master and slave on the hold duration. When a slave does not want to participate in the piconet channel, but still wants to remain synchronized, it can enter the park mode. The parked slave has to wake up regularly to listen to the channel, for staying synchronized or checking broadcast messages.

On top of Baseband is the Link Manager (LM), which is responsible for link configuration and control, security functions, and power management. The corresponding protocol is called Link Manager Protocol (LMP). The Logical Link Control and Adaptation Protocol (L2CAP) provides connection-oriented and connectionless datagram services to upper-layer protocols. Two major functionalities of L2CAP are protocol multiplexing and segmentation and reassembly (SAR).

The Service Discovery Protocol (SDP) defines the means for users to discover which services are available in their neighborhood and the characteristics of these services. The RFCOMM protocol provides emulation of serial ports over L2CAP so as to support many legacy applications based on serial ports over Bluetooth without any modifications. Up to 60 serial ports can be emulated.

B. The Low-Power Sniff Mode

Our main focus, the power-saving issue of Bluetooth, is discussed in more details in this section. In the active mode, the Bluetooth unit is turned on for most of the time to participate in send/receive activities. An active slave listens in even slots for packets. If the slave is not addressed in the current packet, it may sleep until the next even slot that the master may transmit. From the type indication of the packet, the slave can derive the number of slots to be used by the master to transmit the current packet. The addressed slave will reply in the next odd slot after the master's transmission.

In the sniff mode, the slave's listening activities are reduced to save energy. For a sniffed slave, the time slots where the master can communicate to that slave is limited to some specific time slots. These so-called *sniff-attempt slots* arrive periodically, as illustrated in Fig. 1. In the Bluetooth specification, there are three parameters specified for such sniff activity: T_{sniff}, $N_{sniff\_attempt}$, and $N_{sniff\_timeout}$. Since Bluetooth specification separates even and odd slots for the master and slave transmissions, the values of these sniff parameters are all based on *slot pairs* (one even plus one odd slots). In every T_{sniff} slot pairs, the slave will wake up to listen to the master for $N_{sniff\_attempt}$ consecutive (even) slots for possible packets destined to it. After every reception of a packet with a matching address, the slave continues listening for $N_{sniff\_timeout}$ more slots or for the remaining of the $N_{sniff\_attempt}$ slot

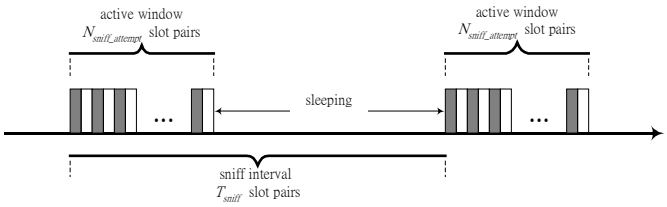


Fig. 1. Sniff interval and active window of Bluetooth (darkened parts are even slots).

pairs, whichever is greater. In this paper, we will call T_{sniff} the *sniff interval*, and $N_{sniff\_attempt}$ the *active window*.

The control packets exchanged between two communicating LMs via Link Manager Protocol (LMP) are termed as LMP_PDUs. There are four LMP_PDUs that involve in sniff-mode management, namely LMP_sniff_req, LMP_accepted, LMP_not_accepted, and LMP_unsniff_req. These PDUs are for making/rejecting/accepting requests and returning to normal active mode. To enter the sniff mode, a LMP_sniff_req request packet can be initiated by either a master or a slave carrying the proposed parameters. Upon receipt of the request, the receiver side can negotiate with the other side on the related sniff parameters by issuing another LMP_sniff_req request packet carrying the suggested parameters. If an agreement can be seen, a LMP_accepted packet is used to place the slave into sniff mode. Otherwise, a LMP_not_accepted packet is returned with a reason code for rejection. Also, note that the sniff parameters can be negotiated in a per master-slave basis.

The sniff mode can be ended by sending a LMP_unsniff_req packet. The counterpart must reply with a LMP_accepted packet. If this is requested by the slave, it will enter active mode after receiving LMP_accepted. If this is requested by the master, the slave will enter active mode immediately after receiving LMP_unsniff_req.

C. Problem Statement

Although the operations of sniff mode have been given in the Bluetooth specification, how to determine the sniff-related parameters is left as an open issue to the designers. One should dynamically determine a proper set of sniff parameters for a slave based on its traffic pattern so as to save as much of its power as possible without incurring too much delay in packet delivery. Further, multiple slaves could be in sniff mode at the same time. How to schedule their active windows on the time axis is a challenging problem since these windows arrive periodically and may extend, conceptually, to the infinity on the time axis. Overlapping of active windows is allowed, but is undesirable. Collectively, we call this the *sniff-scheduling problem*.

Finally, we are aware of the possibility of using hold or park mode for power-saving purpose. However, this paper only considers the sniff mode because it can serve various types of traffic with power saving in mind.

III. The Sniff Scheduling Protocol

In this section, we propose a protocol to exploit the low-power sniff mode of Bluetooth. Because of the master-driven, centrally controlled architecture of Bluetooth, we will maintain a *resource pool (RP)* at the master's side. The sniff parameters of each slave will be adjusted dynamically based on many factors such as the slave's traffic load, current backlog, previous utilization of sniff slots, and the availability of the resource pool. The ultimate goal is to save slaves' powers while keeping packet delays as small as possible. Note that because of the Bluetooth's separation of even and odd slots, all calculations will refer to slot pairs, unless stated otherwise.

Fig. 2 shows the architecture of our sniff-scheduling protocol in a piconet with K active slaves, $1 \leq K \leq 7$. On the master side, there are three main entities: *Evaluator*, *Scheduler*, and *RP*. The master periodically runs the Evaluator for each slave $k, 1 \leq k \leq K$, to evaluate its condition. If

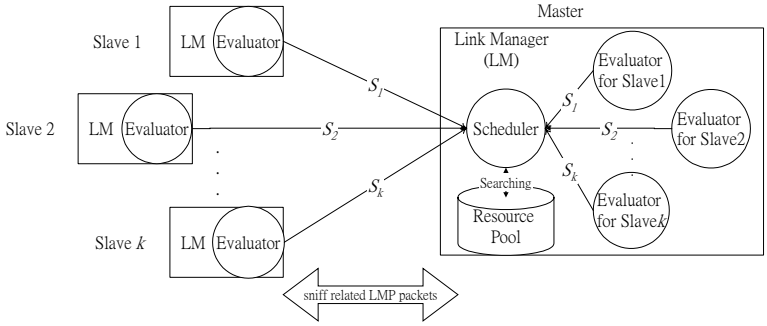


Fig. 2. The architecture of our sniff-scheduling protocol.

necessary, a value S_k, which is used to reflect the estimated traffic load of slave k, is generated and fed into the Scheduler to readjust this slave's sniff parameters. The Scheduler then searches the RP to schedule a new set of sniff parameters for the slave.

On the other hand, slaves also run their own Evaluators periodically. These distributed Evaluators will pass their desired S_k values to the master via a LMP_sniff_req packet. The Scheduler then searches the RP, and arranges new sniff parameters for them. In our protocol, when the Scheduler is unable to find suitable sniff parameters for a slave, a LMP_unsniff_req packet will be issued to invite it into an active mode. This usually happens when the traffic load of the slave is quite large. When the master finds possible to allocate a proper sniff scheduling for the slave, it may bring the slave back to the sniff mode again.

Since the low-power modes of Bluetooth are managed by the Link Manager (LM), our protocol should reside in the LM layer of each Bluetooth unit, monitoring the backlogs of the lower Baseband buffers and issuing proper sniff-scheduling packets. In this paper, we follow the same assumption as in [12] that the master keeps separate buffers in the Baseband layer for its slaves. Each buffer queues the data dedicated to the corresponding slave. Below, we discuss our design in more details.

A. Evaluator

The purpose of the Evaluator is to measure how efficient/inefficient a slave uses the sniff-attempt slots assigned to it and, if necessary, to trigger the Scheduler to readjust its sniff parameters.

The fundamental parameters are explained below. First, we define (T_k, N_k, O_k) as the current sniff parameters (sniff interval, active window size, and offset, respectively) associated with slave k. For each slave k, we have to measure its U_k. This is a ratio between (including) 0 and 1, indicating how many slot pairs of the sniff-attempt slots are used effectively for real data communication under the current setting for slave k. Also, we use B_k to denote the buffer backlog for slave k, i.e., the number of packets currently queued in the local Baseband buffer. By U_k and B_k, we derive a weighted value W_k to measure the current requirement of slave k:

$$W_k = \alpha U_k + (1 - \alpha) B_k / B_{max}, \tag{1}$$

where B_{max} is the maximum buffer space, and α is a constant between 0 and 1 to differentiate the importance of U_k and B_k. The resulting W_k will be tested against the condition $r_{lb} < W_k < r_{ub}$, where r_{lb} and r_{ub} are pre-defined tolerable lower bound and upper bound, respectively, of W_k. If this condition is violated, the master will be triggered to readjust slave k's current sniff parameters; otherwise, no readjustment is needed.

Based on W_k, our objective is to determine the desired slot occupancy S_k of slave k, which represents the expected ratio of the new N_k to the new T_k:

$$S_k = (N_k/T_k) \times W_k/\delta. \quad (2)$$

Intuitively, N_k/T_k is slave k's current slot occupancy. Multiplying this ratio with W_k gives the slot occupancy ratio that is expected to be assigned to this slave. The factor δ is a positive constant below 1 so as to enlarge the expected ratio to tolerate certain level of inaccuracy in our estimation.

Note that a minor logic flaw that we intentionally omit in the above discussion (for ease of presentation) is that when the slave is in the active mode, it has no sniff parameters. So the ratio N_k/T_k becomes meaningless. In this case, we simply replace this ratio by the recent slot occupancy of slave k (with all slots as the denominator). The rest is all the same.

B. Resource Pool

The available sniff-attempt slots that can be allocated to slaves are managed by the Resource Pool at the master side. One may regard the sniff-attempt slots of a slave as a periodical, infinite sequence. However, we need an efficient, finite data structure for the representation.

In this section, we propose to use a two-dimensional $d_1 \times d_2$ matrix M for the representation. The basic idea is as follows. We group (infinite) slot pairs appearing with period $d_1 \cdot d_2$ into one set. For $p = 0..d_2 \cdot d_2 - 1$, define

$$G_p = \{p + q \cdot d_1 \cdot d_2 \mid q \text{ is any non-negative integer}\}.$$

Each entry in matrix M is used to represent the availability of one such slots group, so we define, for $i = 0..d_1 - 1$, $j = 0..d_2 - 1$,

$$M[i,j] = \begin{cases} 0 & \text{if } G_{i \cdot d_2 + j} \text{ is free} \\ 1 & \text{if } G_{i \cdot d_2 + j} \text{ is busy} \end{cases}. \quad (3)$$

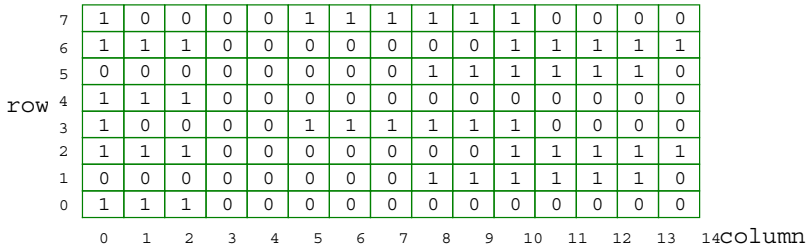
Note that variables d_1 and d_2 are adjustable parameters. The value of $d_1 \cdot d_2$ should be large enough to capture the behavior of those slaves which have very low traffic load and would like to spend very low energy on sniff attempt. Note that $d_1 \cdot d_2$ indicates the maximum allowed sniff interval. To facilitate exponential adjustment of sniff interval, we config d_1 to be power of 2, denoted as 2^u, where u is an non-negative integer. Besides, d_2 is replaced with T, which indicates the minimum allowable sniff interval. Different values of u and T will provide different level of flexibility, as to be shown later. Table I plots several examples of M with size 8×15.

The two-dimensional matrix M can provide us a lot of flexibility in managing periodical time slots. We can manipulate both sniff intervals and active windows of sniffing slaves easily. Two groups that are adjacent in M can be framed together to double the active window. Two groups spaced by a certain distance in the matrix can be grouped together too to divide the sniff interval by half. This can be extended to the combination of more groups easily. Reversibly, we may decrease the active window or increase the sniff interval of a slave to reduce its power consumption by partitioning groups. For example, given the state (a) as shown in Table I, if a slave's packet mean arrival rate is $\frac{16}{120}$, we show four possible ways (b, c, d, e) to arrange the slave's sniff-attempt slots. Case(b) means that the slave is awake every 120 slot pairs, each time lasting for 16 slot pairs. Case(c) means that it is awake every 60 slots pairs, each time lasting for 8 slot pairs. Case(d) means that it is awake every 30 slots pairs, each time lasting for 4 slot pairs. Case(e) means that it is awake every 15 slots pairs, each time lasting for 2 slot pairs.

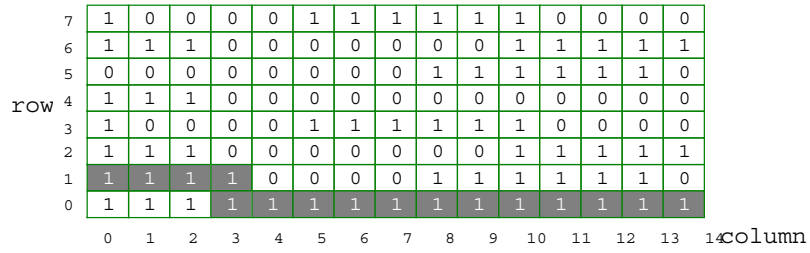
With such formulation, the resource management problem becomes one of allocating proper entries in the matrix M. In Section 4, we will propose two searching policies to this problem.

C. LMP_PDU Flows

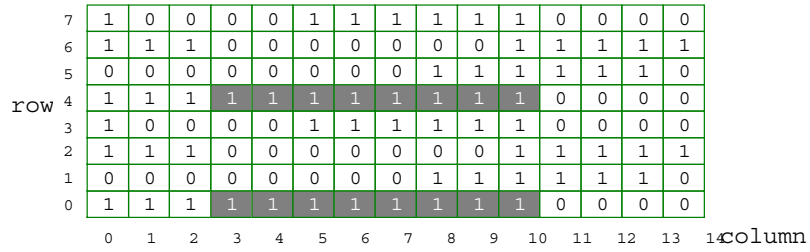
In the Bluetooth specification, both master and slaves can initiate the sniffing request. In our protocol, we also allow the sniffing request to be master- or slave-activated. This section discusses the



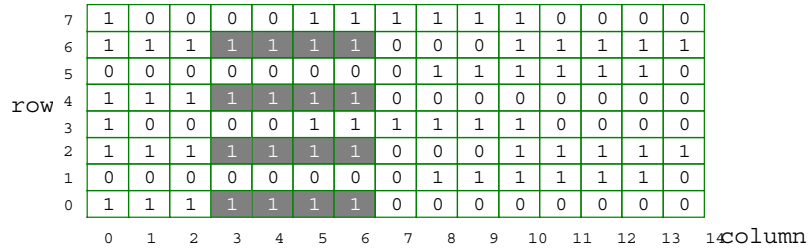
(a)



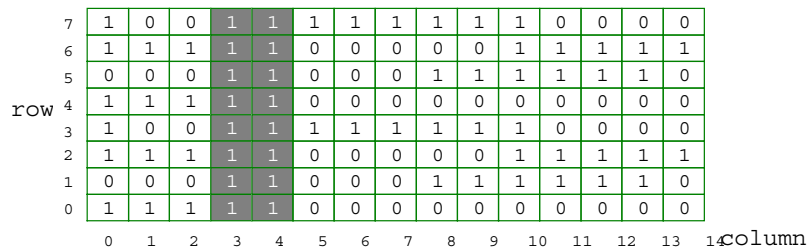
(b)



(c)



(d)



(e)

TABLE I

EXAMPLE: INITIAL STATE OF A 8×15 MATRIX M AND FEASIBLE ASSIGNMENTS IN MATRIX M (IN DARKNESS) FOR A SLOT OCCUPANCY OF $\frac{16}{120}$.

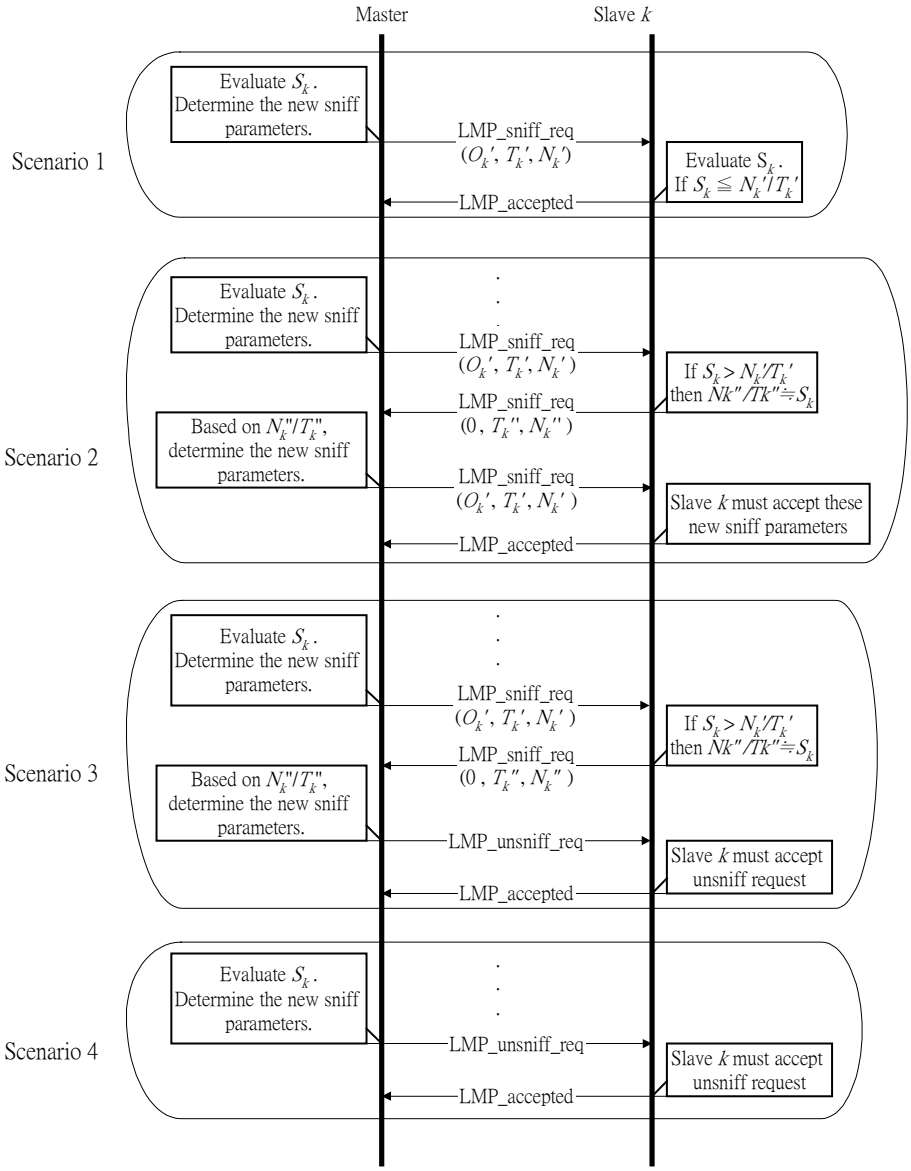


Fig. 3. Scenarios of master-activated sniff parameters negotiation.

related LMP_PDU exchanges. Recall the calculation of W_k for slave k. All the following discussion is triggered by violating the constraint $r_{lb} < W_k < r_{ub}$.

Fig. 3 shows four possible master-activated scenarios. The first one demonstrates that the master proposes a new set of sniff parameters (O_k', T_k', N_k') for slave k. In response, the slave runs its Evaluator to determine its local S_k. If the assigned slot occupancy derived from N_k'/T_k' is no less than S_k, a LMP_accepted can be returned.

The second scenario demonstrates that slave k disagrees on the assigned parameters. So a LMP_sniff_req is returned. However, note that since slave k does not know the status of the RP, we actually intend to return its estimated S_k to the Scheduler for an arrangement. Since S_k is a ratio between 0 and 1, we simply use the two fields T_{sniff} and $N_{sniff\_attempt}$ in LMP_sniff_req to carry two values T_k'' and N_k'', respectively, such that $N_k''/T_k'' \approx S_k$. The Scheduler then tries to allocate a new set of sniff parameters based on N_k''/T_k'' for slave k. In response, the slave issues a LMP_accepted.

The third scenario is similar to the second scenario, but the master fails in allocating a large-enough

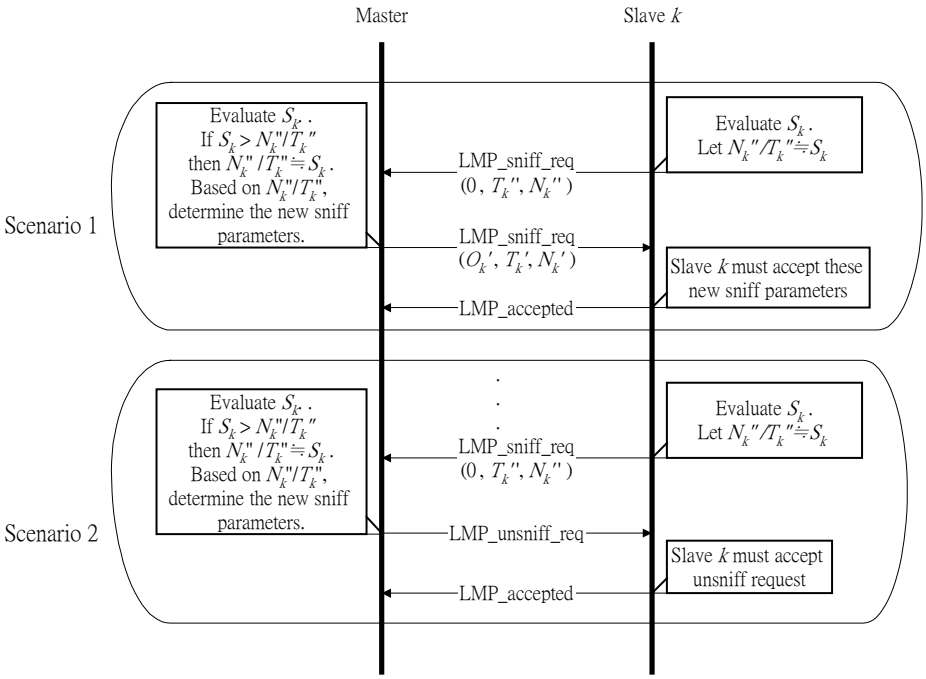


Fig. 4. Scenarios of slave-activated sniff parameters negotiation.

active window from its RP (probably because the matrix M is too crowded or too fragmented). In this case, the master will request the slave to un-sniff itself.

The fourth scenario is the case that from the estimated S_k, the master directly finds that it has difficulty in allocating a large-enough active window from its RP. So an un-sniff request is directly sent to the slave.

Fig. 4 shows the scenarios of slave-activated negotiation. This is similar to the aforementioned second and third scenarios. But this is triggered by finding violation of the condition $r_{lb} < W_k < r_{ub}$ at the slave's side. Again, since the slave does not know the status of RP, we use the two fields T_{sniff} and $N_{sniff\_attempt}$ in LMP_sniff_req to convey the desired S_k to the master. It then follows with a LMP_sniff_req containing the new sniff parameters or LMP_unsniff_req packet, depending on the crowdedness of the RP.

IV. Scheduling Policies for the Resource Pool

The job of the Scheduler is to take an input S_k and determine a suitable set of sniff parameters (denoted by O_k', T_k', and N_k' below) for slave k. These parameters in fact represent a set of slots groups in the matrix M. Before doing the allocation, the old occupancy by this slave on M should be released, which is an easy job. In the following, we propose two policies for searching M, called *LSIF* and *SSIF*.

A. LSIF (Longest Sniff Interval First)

In the LSIF policy, we search the matrix M starting from the longest sniff interval, which is $2^u \cdot T$. If the search fails, we divide the interval by two, which is $2^{u-1} \cdot T$, and do the search again. If the search also fails, we further use the interval $2^{u-2} \cdot T$ to do the search. The search stops once a satisfactory set of slots groups is found. This is repeated until the shortest interval T is tried, in which case we will bring the slave into an active mode, as have been discussed earlier.

Below, we discuss the detailed steps when we search the matrix M with a sniff interval $2^p \cdot T$, where $0 \leq p \leq u$. Since the matrix is of size $2^u \times T$, we will "fold" M into a smaller matrix of size $2^p \times T$.

Specifically, we partition M horizontally and evenly into 2^{u-p} pieces, each of size $2^p \times T$. Then we fold all pieces together by executing a bit-wise OR operator. The meaning of OR is to ensure that each piece of sub-matrices has a free entry. Let the folded matrix be M'. We then search M' sequentially for possible existence of q consecutive free entries (with value 0), where

$$q = \left\lfloor \frac{S_k \cdot 2^u \cdot T}{2^{u-p}} \right\rfloor = \lfloor S_k \cdot 2^p \cdot T \rfloor.$$

The reason that we adopt a floor function instead of a ceiling function here is that the desired slot occupancy has been enlarged in our calculation (by dividing by a $\delta < 1$). Once this search succeeds, we can return $T'_k = 2^p \cdot T$, $N'_k = q$, and $O'_k = i \cdot T + j$, where O'_k is the offset indicating the starting point, say $M'[i, j]$, of the q consecutive free entries in M'.

An example is in Table II, given $T = 15$, $u = 3$, $r_{lb} = 0.2$, $r_{ub} = 0.8$, and $\delta = 0.8$. Assuming that there are $K = 5$ slaves, in the beginning round 0 all slaves share 1/5 of slots groups. In round 1, the estimated W_2 of slave 2 decrease to 0.18. So we release its occupancy on M and allocate a new space for it. In the figure, the ratio "q/t" means that the target number of 0's on M' is q and the searched sniff interval is t. For example, in round 1, we succeed in ratio "2/60" (the underlined one), which means that we find two consecutive 0's when the searched sniff interval is 60. Similarly, in rounds 2 and 3, we succeed in ratios "3/120" and "6/30", respectively.

B. SSIF (Shortest Sniff Interval First)

The SSIF policy only differs from the LSIF policy in that it searches starting from smaller sniff intervals and gradually increasing the searched interval. Specifically, we will start from the shortest sniff interval T. If the search fails, we double the interval and repeat the search, until the longest interval $2^u \cdot T$ is tried. The intuition is that although the slot occupancy may remain the same, with a smaller sniff interval, the buffers for this slave may experience less chances of overflow. Hence, SSIF has potential to improve the network throughput. However, the cost is put on the energy, since the slave needs to wake up more frequently. An example of this policy is in Table III (with all inputs the same as in Table II).

One minor detail that we intentionally omitted in the above discussion is that when the slave's traffic load is very low, it is possible that the value of q is always less than one throughout the searching. In this case, we simply take the sniff interval $2^p \cdot T$ such that $S_k \cdot 2^p \cdot T$ is closest to 1 to do the search again by enforcing $q = 1$.

V. Simulation Results

We have developed a simulator to verify the effectiveness of our protocol. The goal is to observe the interaction between the two seemingly contradicting factors: *network throughput* and *power consumption*. We simulated a single piconet with six Bluetooth units (one master and five slaves). The programming environment was GNU C++ on UNIX SunOS 5.7. No mobility was modeled (i.e., no device joining or leaving the piconet during the simulation process). The power consumption of the master was not a concern since we assume that it has plug-in electricity. Each slave might switch between active and sniff modes, and we didn't consider other modes, such as hold and park. When switching modes or changing sniff parameters, hosts send control packets as described in Fig. 3 and Fig. 4. For each slave, there is a separate buffer queue at the master side. Besides, physical transmission problems such as fading and interference were not taken into account.

Our power model is derived based on experiences in Lucent WaveLAN cards and Bluetooth [13], which is summarized below. It takes half an unit of power for a Bluetooth device to receive a one-slot packet, and one unit to transmit a one-slot packet. Voice traffic is not simulated (but we can simply reserve 1's in matrix M spaced by regular distances to model such traffic). When a slave hears packets dedicated to others, a less amount of power is required since it can turn off its receiver after

Round0 $N_1/T_1 = N_2/T_2 = N_3/T_3 = N_4/T_4 = N_5/T_5 = 3/15$

S1	S1	S1	S2	S2	S2	S3	S3	S3	S4	S4	S4	S5	S5	S5
S1	S1	S1	S2	S2	S2	S3	S3	S3	S4	S4	S4	S5	S5	S5
S1	S1	S1	S2	S2	S2	S3	S3	S3	S4	S4	S4	S5	S5	S5
S1	S1	S1	S2	S2	S2	S3	S3	S3	S4	S4	S4	S5	S5	S5
S1	S1	S1	S2	S2	S2	S3	S3	S3	S4	S4	S4	S5	S5	S5
S1	S1	S1	S2	S2	S2	S3	S3	S3	S4	S4	S4	S5	S5	S5
S1	S1	S1	S2	S2	S2	S3	S3	S3	S4	S4	S4	S5	S5	S5
S1	S1	S1	S2	S2	S2	S3	S3	S3	S4	S4	S4	S5	S5	S5

Round1 $W_2 = 0.18 < r_{lb} \Rightarrow S_2 = [(3/15)*0.18]/0.8 = 0.045$
$0.045 = 5/120 = \underline{2/60} = 1/30 = 0/15 \Rightarrow (O_k', T_k', N_k') = (3, 60, 2)$

S1	S1	S1				S3	S3	S3	S4	S4	S4	S5	S5	S5
S1	S1	S1				S3	S3	S3	S4	S4	S4	S5	S5	S5
S1	S1	S1				S3	S3	S3	S4	S4	S4	S5	S5	S5
S1	S1	S1	S2	S2		S3	S3	S3	S4	S4	S4	S5	S5	S5
S1	S1	S1				S3	S3	S3	S4	S4	S4	S5	S5	S5
S1	S1	S1				S3	S3	S3	S4	S4	S4	S5	S5	S5
S1	S1	S1				S3	S3	S3	S4	S4	S4	S5	S5	S5
S1	S1	S1	S2	S2		S3	S3	S3	S4	S4	S4	S5	S5	S5

Round2 $W_3 = 0.11 < r_{lb} \Rightarrow S_3 = [(3/15)*0.11]/0.8 = 0.028$
$0.028 = \underline{3/120} = 1/60 = 0/30 = 0/15 \Rightarrow (O_k', T_k', N_k') = (5, 120, 3)$

S1	S1	S1							S4	S4	S4	S5	S5	S5
S1	S1	S1							S4	S4	S4	S5	S5	S5
S1	S1	S1							S4	S4	S4	S5	S5	S5
S1	S1	S1	S2	S2					S4	S4	S4	S5	S5	S5
S1	S1	S1							S4	S4	S4	S5	S5	S5
S1	S1	S1							S4	S4	S4	S5	S5	S5
S1	S1	S1							S4	S4	S4	S5	S5	S5
S1	S1	S1	S2	S2	S3	S3	S3		S4	S4	S4	S5	S5	S5

Round3 $W_4 = 0.9 > r_{ub} \Rightarrow S_4 = [(3/15)*0.9]/0.8 = 0.23$
$0.23 = 27/120 = 13/60 = \underline{6/30} = 3/15 \Rightarrow (O_k', T_k', N_k') = (18, 30, 6)$

S1	S1	S1	S4	S4	S4	S4	S4	S4				S5	S5	S5
S1	S1	S1										S5	S5	S5
S1	S1	S1	S4	S4	S4	S4	S4	S4				S5	S5	S5
S1	S1	S1	S2	S2								S5	S5	S5
S1	S1	S1	S4	S4	S4	S4	S4	S4				S5	S5	S5
S1	S1	S1										S5	S5	S5
S1	S1	S1	S4	S4	S4	S4	S4	S4				S5	S5	S5
S1	S1	S1	S2	S2	S3	S3	S3					S5	S5	S5

TABLE II
SEARCHING EXAMPLE OF LSIF.

monitoring the packet header. In this case, it only consumes 1/6 of receiving power, which is 0.083 units. Bluetooth also defines some short packets, such as ACK and NULL (to respond to a poll with no data). We assume the power consumption for delivering such packets to be 1/6 of the transmission power, which is 0.167 units.

In addition to our LSIF and SSIF policies, three other approaches are simulated for comparison purpose. The first one is called Always-Active (AA), where we enforce all slaves to always stay in the active mode. The master polls its slaves using a round robin policy. Note that, while naive, AA is used here only as a referential point. The second approach is called Always-Sniff-with-Varying-Sniff-Interval (AS_VSI), where we enforce slaves to always stay in the sniff mode, but the active window must remain as a constant. When a slave's slot utilization $\leq r_{lb}$, its sniff interval will be doubled. On the other hand, its sniff interval will be cut in half if the slot utilization $\geq r_{ub}$. The smallest sniff interval is T, while the largest possible is $2^u \cdot T$. The third approach is called Always-Sniff-with-Varying-Active-Window (AS_VAW), where we enforce the sniff interval to be constant. The active window will be doubled when slot utilization $\geq r_{ub}$, but cut in half when $\leq r_{lb}$. The lower and upper bounds of active window size is 1 and $T/5$, respectively.

Below, we divide our presentation into two parts. Section V-A shows the results under fixed traffic

<u>Round0</u> $N_1/T_1 = N_2/T_2 = N_3/T_3 = N_4/T_4 = N_5/T_5 = 3/15$

S1	S1	S1	S2	S2	S2	S3	S3	S3	S4	S4	S4	S5	S5	S5
S1	S1	S1	S2	S2	S2	S3	S3	S3	S4	S4	S4	S5	S5	S5
S1	S1	S1	S2	S2	S2	S3	S3	S3	S4	S4	S4	S5	S5	S5
S1	S1	S1	S2	S2	S2	S3	S3	S3	S4	S4	S4	S5	S5	S5
S1	S1	S1	S2	S2	S2	S3	S3	S3	S4	S4	S4	S5	S5	S5
S1	S1	S1	S2	S2	S2	S3	S3	S3	S4	S4	S4	S5	S5	S5
S1	S1	S1	S2	S2	S2	S3	S3	S3	S4	S4	S4	S5	S5	S5
S1	S1	S1	S2	S2	S2	S3	S3	S3	S4	S4	S4	S5	S5	S5

<u>Round1</u> $W_2 = 0.18 < r_{lb}$ => $S_2 = [(3/15)*0.18]/0.8 = 0.045$
$0.045 = 5/120 = 2/60 = \underline{1/30} = 0/15$ => $(O_k', T_k', N_k') = (3, 30, 1)$

S1	S1	S1				S3	S3	S3	S4	S4	S4	S5	S5	S5
S1	S1	S1	S2			S3	S3	S3	S4	S4	S4	S5	S5	S5
S1	S1	S1				S3	S3	S3	S4	S4	S4	S5	S5	S5
S1	S1	S1	S2			S3	S3	S3	S4	S4	S4	S5	S5	S5
S1	S1	S1				S3	S3	S3	S4	S4	S4	S5	S5	S5
S1	S1	S1	S2			S3	S3	S3	S4	S4	S4	S5	S5	S5
S1	S1	S1				S3	S3	S3	S4	S4	S4	S5	S5	S5
S1	S1	S1	S2			S3	S3	S3	S4	S4	S4	S5	S5	S5

<u>Round2</u> $W_3 = 0.11 < r_{lb}$ => $S_3 = [(3/15)*0.11]/0.8 = 0.028$
$0.028 = 3/120 = \underline{1/60} = 0/30 = 0/15$ => $(O_k', T_k', N_k') = (4, 60, 1)$

S1	S1	S1							S4	S4	S4	S5	S5	S5
S1	S1	S1	S2						S4	S4	S4	S5	S5	S5
S1	S1	S1							S4	S4	S4	S5	S5	S5
S1	S1	S1	S2	S3					S4	S4	S4	S5	S5	S5
S1	S1	S1							S4	S4	S4	S5	S5	S5
S1	S1	S1	S2						S4	S4	S4	S5	S5	S5
S1	S1	S1							S4	S4	S4	S5	S5	S5
S1	S1	S1	S2	S3					S4	S4	S4	S5	S5	S5

<u>Round3</u> $W_4 = 0.9 > r_{ub}$ => $S_4 = [(3/15)*0.9]/0.8 = 0.23$
$0.23 = 27/120 = 13/60 = 6/30 = \underline{3/15}$ => $(O_k', T_k', N_k') = (5, 15, 3)$

S1	S1	S1			S4	S4	S4					S5	S5	S5
S1	S1	S1	S2		S4	S4	S4					S5	S5	S5
S1	S1	S1			S4	S4	S4					S5	S5	S5
S1	S1	S1	S2	S3	S4	S4	S4					S5	S5	S5
S1	S1	S1			S4	S4	S4					S5	S5	S5
S1	S1	S1	S2		S4	S4	S4					S5	S5	S5
S1	S1	S1			S4	S4	S4					S5	S5	S5
S1	S1	S1	S2	S3	S4	S4	S4					S5	S5	S5

TABLE III
Searching example of SSIF.

patterns, while Section V-B does under varying traffic patterns. The latter is intended to model real system traffics and demonstrate the flexibility of our protocol in catching such dynamics. Through the presentation, we shall provide a guideline for choosing proper parameters for our protocol. In our simulation, we always adopt $r_{lb} = 0.3$ and $r_{ub} = 0.7$. Each simulation run lasts for 100,000 slot pairs.

A. Fixed Traffic Load

In this part, we assume a Poisson process with packet arrival rate $\lambda = 0.2$ (packets per time slot) for each buffer (at both the master and slave sides). Fig. 5 illustrates the power consumption and network throughput against buffer size B_{max}. We observe that, with enough buffer space (≥ 50), all five approaches can achieve a high throughput close to 0.99. This is because fixed traffic loads are easy to catch. As opposed to AA, all other four schemes consume significantly less powers.

These observations motivate us to derive the following simulations, where traffics are non-uniform, so as to reveal the strength of our proposals.

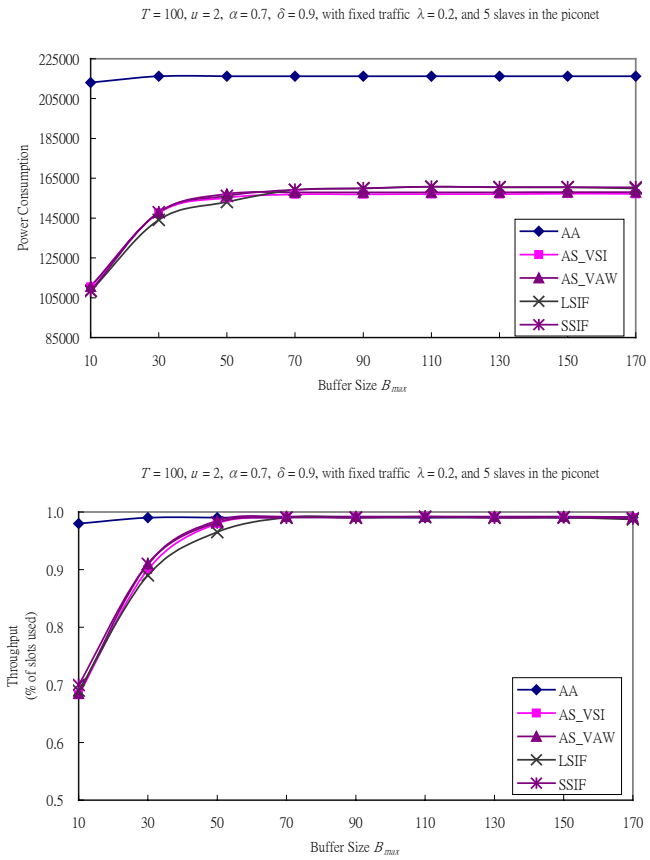


Fig. 5. Effect of B_{max} under fixed traffic load.

B. *Varying Traffic Load*

In this part, we adopt the variable traffic model similar to that proposed in [14]. Packets still arrive by the Poisson process, but with different rates. Two types of traffic patterns will be explored. The first one is denoted as TypeI(λ_M-λ_S), which means that the arrival rate at the master's side is λ_M, and the arrival rate at the slave's side is λ_S. The second one is denoted as TypeII(λ_A-λ_B), which means that there are two kinds of arrival rates, λ_A and λ_B, for both the master and the slave. The master and the slave change states between rates λ_A and λ_B independently, and the transition probability from one rate to the other is 0.01 in both directions.

We first investigate the effect of weight α on our LSIF and SSIF schemes. Fig. 6(a) plots the network throughput against α, where five slaves with traffic patterns TypeI(0.2-0.2), TypeI(0.19-0.01), TypeI(0.01-0.19), TypeII(0.19-0.01), and TypeII(0.19-0.01) are simulated. It indicates that the throughput can be consistently improved as α increases, until $\alpha \leq 0.7$. At $\alpha = 0.0$, LSIF and SSIF give the worst throughput among all. The reason is that the evaluating metric is all based on buffer information when $\alpha = 0.0$, which is unfair. As a result, the assigned sniff-attempt slots are not able to handle future traffic well, thus degrading the performance. After $\alpha > 0.7$, the throughput starts to degrade as α grows. However, even when $\alpha = 1.0$, our LSIF and SSIF still outperform the other three schemes. So a value between 0.6 and 0.7 for α could be the best choice.

The above simulation in fact reveals two interesting phenomena. First, the slot utilization factor alone can not predict the traffic well. One should take both slot utilization and buffer backlog information to predict future traffic. Second, and much to our surprise, our LSIF and SSIF schemes can even provide a better network throughput than a naive always-active, round-robin scheme as α is properly set. The reason is that, in the AA case, the master wastes much time polling slaves with no backlogs,

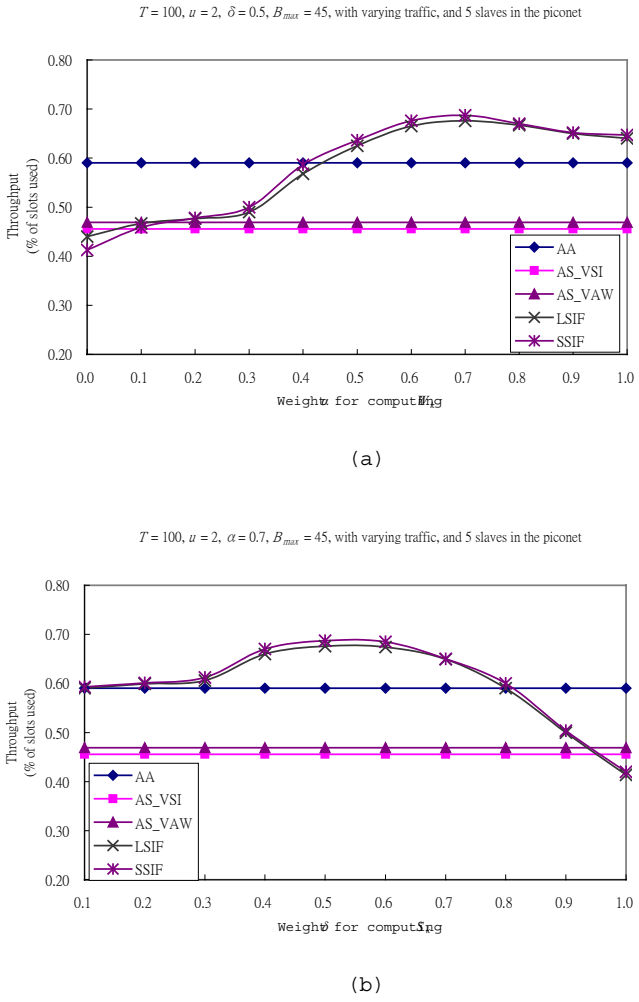


Fig. 6. Effect of (a) α and (b) δ on LSIF and SSIF under varying traffic load.

resulting in reduced throughput. This indicates a prospective direction that one can save power and improve network throughput at the same time (which are contradicting factors by intuition).

Fig. 6(b) illustrates the impact of weight δ on LSIF and SSIF, with $\alpha = 0.7$ and the same traffic pattern as above. It shows that before $\delta \leq 0.6$, the throughput can be improved as δ grows. Recall that δ is a factor to enlarge the expected sniff-attempt slots so as to tolerate certain level of inaccuracy in our estimation. With a too small value (such as $\delta = 0.1$), LSIF and SSIF will degenerate into the AA scheme, since slaves can hardly obtain such a large slot occupancy. As a result, all slaves may remain active for most of the time. This also violates our goal of conserving power. After $\delta > 0.6$, the throughput starts to decline as δ increases. After $\delta > 0.8$, our throughput falls behind that of the AA scheme. This is because our prediction is too conservative to catch the dynamics of variable traffics. So a reasonable value for δ would be between 0.5 and 0.7.

Also with the same traffic pattern, in Fig. 7(a), we study network throughput with different values of u, which controls the largest possible sniff interval. For both LSIF and SSIF, it shows that the throughput remains at around the same level when $u = 0, 1$ and 2, but decreases as u is larger. With $u = 0$, there is only one sniff interval available, which is T. With $u = 1$ and 2, there are two and three sniff intervals available, respectively. A larger sniff interval means that the slave needs to switch modes less frequently, which is more favorable. Based on these considerations, one may choose a proper value

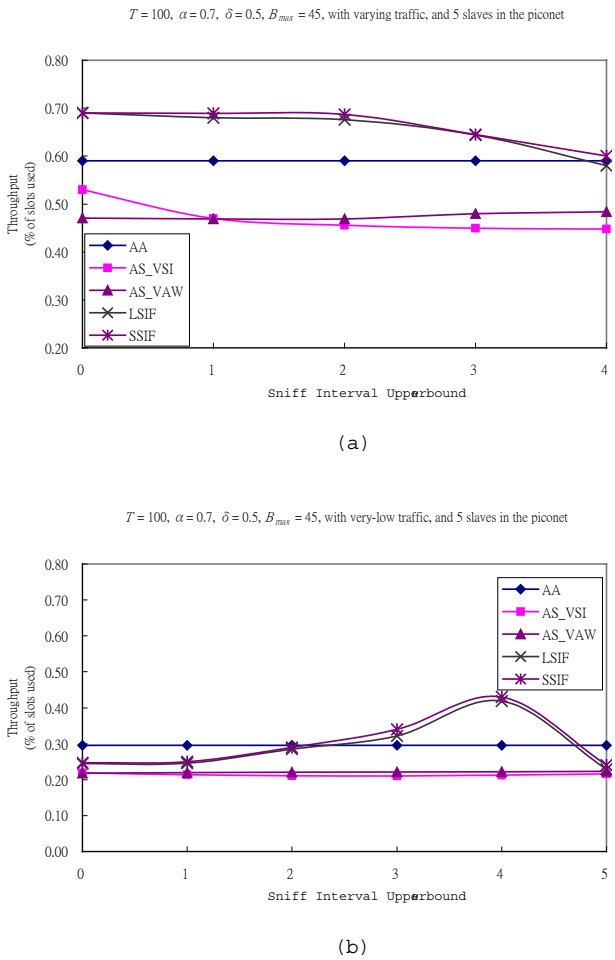


Fig. 7. Effect of u on LSIF and SSIF under (a) varying traffic and (b) very-low traffic load.

of u to use.

One may note that in the previous simulation, the advantage of using our two-dimensional matrix M is not well justified. When $u = 0$, M degenerates to a one-dimensional matrix. So why should one need a two-dimensional M remains a question. The reason is that we have adopted $T = 100$. Since the lowest traffic load that we may inject for each entity is 0.01, A sniff interval of $T = 100$ and active window of 1 can properly catch such traffic without much wastage. To justify this point, we have simulated hosts with very low traffic loads. We have conducted another experiment with five slaves having the following traffic patterns: TypeI(0.2-0.2), TypeI(0.001-0.001), TypeI(0.002-0.002), TypeII(0.002-0.005), and TypeII(0.002-0.005). The result is in Fig. 7(b). It indicates that the throughput can be improved as u increases, until $u \leq 4$. Before $u < 2$, the throughput of LSIF and SSIF is worse than AA. This phenomenon is due to slot wastage caused by sniff intervals that are too short. (For example, to handle a low arrival rate of 0.002, the Scheduler should reserve one slot out of every 500 slots in average. When u is set too small, say $u < 2$, the Scheduler will reserve too much resource for such slaves. Therefore, both network throughput and power consumption might get hurt in this case.) As mentioned earlier, the value of $2^u \cdot T$ should be large enough to capture the behavior of those slaves which have very-low traffic load. In Fig. 7(b), a value of $u = 4$ will perform the best. This justifies that our two-dimensional matrix representation of M is flexible enough to schedule sniffing slots for hosts with both high and very low traffic loads.

In Fig. 8(a), we investigate the effect of T, which represents the smallest possible sniff interval,

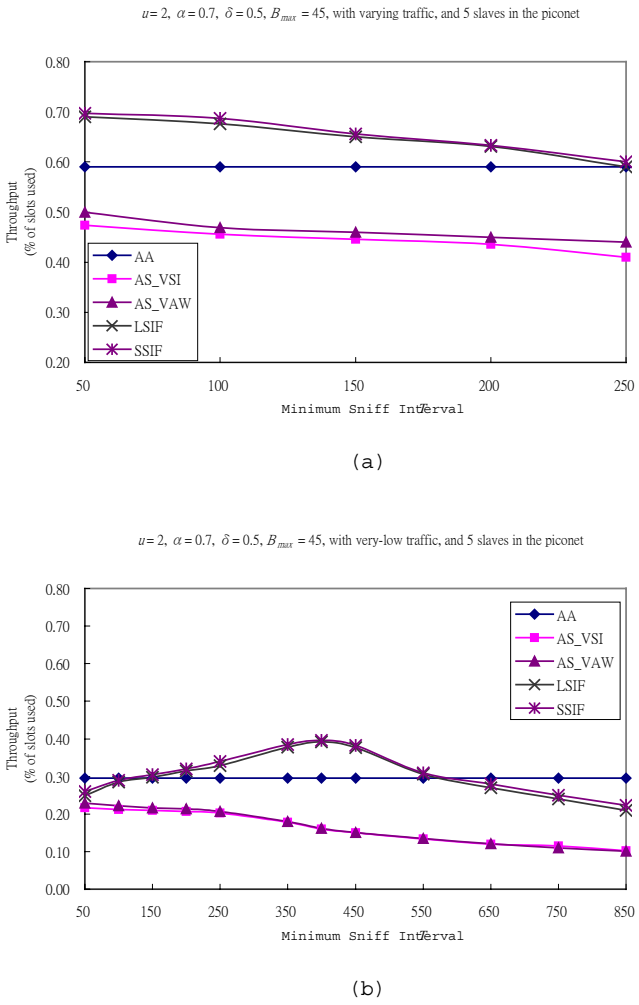


Fig. 8. Effect of T on LSIF and SSIF under (a) varying traffic and (b) very-low traffic load.

under the traffic patterns of TypeI(0.2-0.2), TypeI(0.19-0.01), TypeI(0.01-0.19), TypeII(0.19-0.01), and TypeII(0.19-0.01). It shows that the throughput of LSIF and SSIF will slightly decrease as T grows, so a T between 50 and 100 will be proper. The reason for the degradation in throughput is that we activate the Evaluator based on the value of T. A larger T will trigger the protocol to re-evaluate its traffic load less frequently. The inaccuracy in load estimation will cause slot wastage. We believe that this problem can be fixed by using a different rule to trigger our Evaluators, which will be directed to future research.

Another experiment to investigate the impact of T is in Fig. 8(b), where we try low traffic patterns: TypeI(0.2-0.2), TypeI(0.001-0.001), TypeI(0.002-0.002), TypeII(0.002-0.005), and TypeII(0.002-0.005). Similar to the earlier observations, lower traffic loads require larger $2^u \cdot T$. That is why we see continuous improvement before $T \leq 400$. After T is too large, our Evaluators will react to traffic changes too slowly, causing degradation in throughput. The results from Fig. 8(a) and Fig. 8(b) suggest a T between 100 and 200 to be chosen.

In Fig. 9, we set our parameters as recommended above and look at the impact of buffer spaces, B_{max}. We observe both power consumption and network throughput against different buffer spaces. It shows that the throughput climbs as B_{max} increases, up to the point $B_{max} = 50$. After $B_{max} \geq 50$, the throughput remains almost the same. So a buffer space between 30 and 50 will be proper. For

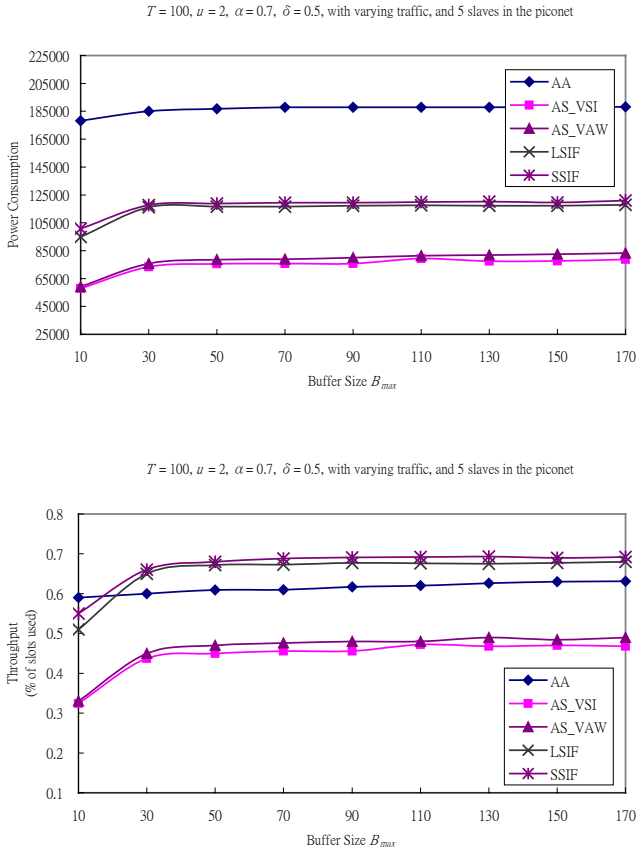


Fig. 9. Effect of B_{max} under varying traffic load

LSIF and SSIF, with $B_{max} \approx 30$, they increase system throughput by around 16.7% as compared to AA, while reducing power consumption by around 37.8% as compared to AA.

VI. Conclusions

We have proposed an adaptive and efficient protocol for managing the low-power sniff mode in Bluetooth. Two essential parts in our protocol are the Evaluator and the Scheduler, which are responsible for measuring how well slaves utilize their sniff-attempt slots and for arranging the sniff parameters for slaves in the sniff mode. A new representation based on a two-dimensional matrix is proposed to maintain slaves' sniff-attempt slots, which are conceptually infinite sequences of periodical slots. Two searching strategies, LSIF (Longest Sniff Interval First) and SSIF (Shortest Sniff Interval First), are proposed to look for available sniff-attempt slots in the two-dimensional matrix. Our simulation results have indicated that, with proper settings and buffer spaces, the protocol can potentially improve network throughput, while reducing power consumption, compared to a naive always-active, round-robin protocol.

Acknowledgments

Y.-C. Tseng would like to thank the Lee and MTI Center for Networking Research at NCTU, the Ministry of Education (contract numbers 89-H-FA07-1-4 and 89-E-FA04-1-4), and the National Science Council, Taiwan (contract number NSC90-2213-E009-154) for financially supporting this research.

References

[1] Bluetooth SIG http://www.bluetooth.com, "Bluetooth Specification v1.1," February, 2001.

[2] Jaap C. Haartsen, "The Bluetooth Radio System," *IEEE Personal Communications*, February 2000.
[3] Jaap C. Haartsen and Sven Mattisson, "Bluetooth - A New Low-Power Radio Interface Providing Short-Range Connectivity," *Proceedings of the IEEE*, vol. 88, October 2000.
[4] R. Bruno, M. Conti, and E. Gregori, "WLAN Technologies for Mobile Ad Hoc Networks," *IEEE Proceedings of the 34th Hawaii International Conference on System Sciences*, 2001.
[5] Shih-Lin Wu, Yu-Chee Tseng, and Jang-Ping Sheu, "Intelligent Medium Access for Mobile Ad Hoc Networks with Busy Tones and Power Control," *IEEE Journal on Selected Areas in Communications*, vol. 18, pp. 1647–1657, September, 2000.
[6] Ram Ramanathan and Regina Rosales-Hain, "Topology Control of Multihop Wireless Networks Using Transmit Power Adjustment," *IEEE INFOCOM*, pp. 404–413, 2000.
[7] Chi-Fu Huang, Yu-Chee Tseng, Shih-Lin Wu, and Jang-Ping Sheu, "Increasing the Throughput of Multihop Packet Radio Networks with Power Adjustment," *International Conference on Computer Cummunications and Networks*, 2001.
[8] Jung hee Ryu, Sanghwa Song, and Dong-Ho Cho, "A Power-Saving Multicast Routing Scheme in 2-tir Hierarchical Mobile Ad-Hoc Networks," *IEEE Vehicular Technology Conference*, vol. 4, pp. 1974–1978, 2000.
[9] Suresh Singh, Mike Woo, and C. S. Raghavendra, "Power-Aware Routing in Mobile Ad Hoc Networks," *International Conference on Mobile Computing and Networking*, pp. 181–190, 1998.
[10] Hagen Woesner, Jean-Pierre Ebert, Morten Schlager, and Adam Wolisz, "Power-Saving Mechanisms in Emerging Standards for Wireless LANs: The MAC Level Perspective," *IEEE Personal Communications*, pp. 40–48, June 1998.
[11] Indraneel Chakrabory, Abhishek Kashyap, Apurva Kumar, Anupam Rastogi, Huzur Saran, and Rajeev Shorey, "MAC Scheduling Policies with Reduced Power Consumption and Bounded Packet Delays for Centrally Controlled TDD Wireless Networks," *IEEE International Conference on Communications*, 2001.
[12] Abhishek Das, Abhishek Ghose, Ashu Razdan, Huzur Saran, and Rajeev Shorey, "Enhancing Performance of Asynchronous Data Traffic over the Bluetooth Wireless Ad-hoc Network," *IEEE INFOCOM*, 2001.
[13] Sumit Garg, Manish Kalia, and Rajeev Shorey, "MAC Scheduling Policies for Power Optimization in Bluetooth: A Master Driven TDD Wireless System," *IEEE Vehicular Technology Conference*, 2000.
[14] Manish Kalia, Deepak Bansal, and Rajeev Shorey, "Data Scheduling and SAR for Bluetooth MAC," *IEEE Vehicular Technology Conference*, 2000.

Biographies

Ting-Yu Lin (tylin@csie.nctu.edu.tw) received her B.S. degree in Computer Science from the National Chiao-Tung University, Taiwan, in 1996. She is currently a Ph.D. candidate at the Department of Computer Science and Information Engineering, National Chiao-Tung University, Taiwan. Her research interests include wireless communication, mobile computing, personal-area networks, and energy conservation.

Yu-Chee Tseng (yctseng@csie.nctu.edu.tw) is currently a Full Professor at the Department of Computer Science and Information Engineering, National Chiao-Tung University, Taiwan. Dr. Tseng has served as a Program Committee Member in several international conferences and as a Guest Editor in several journals, including IEEE Trans. on Computers, Wireless Communications and Mobile Computing, Wireless Networks, and Journal of Internet Technology. His research interests include wireless communication, network security, parallel and distributed computing, and computer architecture. Dr. Tseng is a member of the IEEE Computer Society.

論文名稱：Power-Saving Protocols for IEEE 802.11-Based Multi-Hop Ad Hoc Networks

作者：Y.-C. Tseng, C.-S. Hsu, and T.-Y. Hsieh

發表情況: ***IEEE INFOCOM***, 2002

Power-Saving Protocols for IEEE 802.11-Based Multi-Hop Ad Hoc Networks

Yu-Chee Tseng, Chih-Shun Hsu, Ten-Yueng Hsieh

Abstract—***Power-saving* is a critical issue for almost all kinds of portable devices. In this paper, we consider the design of power-saving protocols for *mobile ad hoc networks (MANETs)* that allow mobile hosts to switch to a low-power *sleep* mode. The MANETs being considered in this paper are characterized by unpredictable mobility, multi-hop communication, and no clock synchronization mechanism. In particular, the last characteristic would complicate the problem since a host has to predict when another host will wake up to receive packets. We propose three power management protocols, namely *dominating-awake-interval*, *periodically-fully-awake-interval*, and *quorum-based* protocols, which are directly applicable to IEEE 802.11-based MANETs. As far as we know, the power management problem for multi-hop MANETs has not been seriously addressed in the literature. Existing standards, such as IEEE 802.11, HIPERLAN, and bluetooth, all assume that the network is fully connected or there is a clock synchronization mechanism. Extensive simulation results are presented to verify the effectiveness of the proposed protocols.**

Keywords— **HIPERLAN, IEEE 802.11, mobile ad hoc network (MANET), power management, power saving, wireless communication.**

I. Introduction

COMPUTING and communication anytime, anywhere is a global trend in today's development. Ubiquitous computing has been made possible by the advance of wireless communication technology and the availability of many lightweight, compact, portable computing devices. Among the various network architectures, the design of *mobile ad hoc network (MANET)* has attracted a lot of attention recently. A MANET is one consisting of a set of mobile hosts which can communicate with one another and roam around at their will. No base stations are supported in such an environment, and mobile hosts may have to communicate with each other in a *multi-hop* fashion. Applications of MANETs occur in situations like battlefields, major disaster areas, and outdoor assemblies. It is also a prospective candidate to solve the "last-mile" problem for broadband Internet service providers [1].

One critical issue for almost all kinds of portable devices supported by battery powers is *power saving*. Without power, any mobile device will become useless. Battery power is a limited resource, and it is expected that battery technology is not likely to progress as fast as computing and communication technologies do. Hence, how to lengthen the lifetime of batteries is an important issue, especially for MANET, which is all supported by batteries.

Y. C. Tseng and T. Y. Hsieh are with the Department of Computer Science and Information Engineering, National Chiao Tung University, Hsin-Chu, Taiwan. E-mail: yctseng, tyhsieh@csie.nctu.edu.tw .

C. S. Hsu is with the Department of Computer Science and Information Engineering, National Central University, Chung-Li, Taiwan. E-mail: cshsu@axp1.csie.ncu.edu.tw .

This work is supported by the National Science Council of the Republic of China under Grant #NSC90-2213-E-009-049 and #NSC90-2213-E-009-154, and the Ministry of Education, the Republic of China, under grant 90-H-FA07-1-4(Learning Technology).

The authors would also like to thank the Lee and MTI Center for Networking Research at NCTU, Taiwan, for sponsoring this research.

Solutions addressing the power-saving issue in MANETs can generally be categorized as follows:

- *Transmission Power Control:* In wireless communication, transmission power has strong impact on bit error rate, transmission rate, and inter-radio interference. These are typically contradicting factors. In [2], power control is adopted to reduce interference and improve throughput on the MAC layer. How to determine transmission power of each mobile host so as to determine the best network topology, or known as *topology control*, is addressed in [3], [4], [5]. How to increase network throughput by power adjustment for packet radio networks is addressed in [6].
- *Power-Aware Routing:* Power-aware routing protocols have been proposed based on various power cost functions [7], [8], [9], [10], [11]. In [7], when a mobile host's battery level is below a certain threshold, it will not forward packets for other hosts. In [10], five different metrics based on battery power consumption are proposed. Reference [11] considers both hosts' lifetime and a distance power metric. A hybrid environment consisting of battery-powered and outlet-plugged hosts is considered in [8]. Two distributed heuristic clustering approaches for multicasting are proposed in [9] to minimizing the transmission power.
- *Low-Power Mode:* More and more wireless devices can support low-power sleep modes. IEEE 802.11 [12] has a power-saving mode in which a radio only needs to be awake periodically. HyperLAN allows a mobile host in power-saving mode to define its own active period. An active host may save powers by turning off its equalizer according to the transmission bit rate. Comparisons are presented in [13] to study the power-saving mechanisms of IEEE 802.11 and HIPERLAN in ad hoc networks. Bluetooth [14] provides three different low-power modes: *sniff*, *hold*, and *park*. Other references include [15], [16], [17], [18], [19], [20], [21].

This paper studies the management of power-saving (PS) modes for IEEE 802.11-based MANETs and thus falls into the last category of the above classification. We consider MANETs which are characterized by multi-hop communication, unpredictable mobility, no plug-in power, and no clock synchronization mechanism. In particular, the last characteristic would complicate the problem since a host has to predict when another host will wake up to receive packets. Thus, the protocol must be asynchronous. As far as we know, the power-management problem for multi-hop MANETs has not been addressed seriously in the literature. Existing standards, such as IEEE 802.11 and HYPERLAN, do support PS modes, but assume that the MANET is fully connected. Bluetooth also has low-power modes, but is based on a master-slave architecture, so time synchronization is trivial. The works [18], [19] address the power-saving problem, but assume the existence of access points. A lot of works have focused on multi-hop MANETs on issues such as power-aware

routing, topology control, and transmission power control (as classified above), but how to design PS mode is left as an open problem.

Two major challenges that one would encounter when designing power-saving protocols are: *clock synchronization* and the *neighbor discovery*. Clock synchronization in a multi-hop MANET is difficult since there is no central control and packet delays may vary due to unpredictable mobility and radio interference. PS modes are typically supported by letting low-power hosts wake up only in specific time. Without precise clocks, a host may not be able to know when other PS hosts will wake up to receive packets. Further, a host may not be aware of a PS host at its neighborhood since a PS host will reduce its transmitting and receiving activities. Such incorrect neighbor information may be detrimental to most current routing protocols because the route discovery procedure may incorrectly report that there is no route even when routes actually exist with some PS hosts in the middle. These problems will be discussed in more details in Section II.

In this paper, we propose three asynchronous power management protocols for multi-hop MANETs, namely *dominating-awake-interval*, *periodically-fully-awake-interval*, and *quorum-based* protocols. We target ourselves at IEEE 802.11-based LAN cards. The basic idea is twofold. First, we enforce PS hosts send more beacon packets than the original IEEE 802.11 standard does. Second and most importantly, we carefully arrange the wake-up and sleep patterns of PS hosts such that any two neighboring hosts are guaranteed to detect each other in finite time even under PS mode.

Based on our power-saving protocols, we then show how to perform unicast and broadcast in an environment with PS hosts. Simulation results are presented, which show that our protocols can save lots of powers when the traffic load is not high.

The rest of this paper is organized as follows. Preliminaries are given in Section 2. In Section 3, we present our power-saving protocols. Unicast and broadcast protocols based on our power-saving mechanisms are in Section 4. Simulation results are presented in Section 5. Section 6 concludes this paper.

II. Preliminaries

In this section, we start with a general review on power-saving works, followed by detailed design of PS mode in IEEE 802.11. Then we motivate our work by pointing out some problems connecting to PS mode in multi-hop MANETs.

A. Reviews of Power Mode Management Protocols

Several power management protocols have been proposed for MANET in [15], [16], [20], [21]. The *PAMAS* (Power Aware Multi-Access protocol with Signalling) [15] protocol allows a host to power its radio off when it has no packet to transmit/receive or any of its neighbors is receiving packets, but a separate signalling channel to query neighboring hosts' states is needed. Reference [16] provides several sleep patterns and allows mobile hosts to select their sleep patterns based on their battery status and quality of service, but a special hardware, called *Remote Activated Switch (RAS)*, is required which can receive wakeup signals even when the mobile host has entered a sleep state. A connected-dominated-set-based power-saving protocol is proposed in [20]. Some hosts must serve as *coordinators*, which are chosen according to their remaining battery energies and the numbers of neighbors they can connect to. In the network, only coordinators need to keep awake; other hosts can enter the sleeping mode. Coordinators are responsible of relaying packets for neighboring hosts. With a similar idea, a grid-based energy-saving routing protocol is proposed in [21]. With the help of GPS, the area is partitioned in to small subareas called grids, in each of which only one host needs to remain active to relay packets for other hosts in the same grid.

A page-and-answer protocol is proposed in [18] for wireless networks with base stations. A base station will keep on sending paging messages whenever there are buffered packets. Each mobile host powers up periodically. However, there is no time synchronization between the base station and mobile hosts. On reception of paging messages, mobile hosts return acknowledgements, which will trigger the base station to stop paging and begin transmitting buffered packets. After receiving the buffered packets, mobile hosts return to power-saving mode, and the process repeats. When the system is too heavily loaded, the base station may spend most of its time in transmitting buffered packets, instead of paging messages. This may result in long packet delays for power-saving hosts. A theoretical analysis of [18] is in [22]. Several software power-control issues for portable computers are discussed in [17]. How to combine power management and power control for wireless cards is addressed in [19].

B. Power-Saving Modes in IEEE 802.11

IEEE 802.11 [12] supports two power modes: *active* and *power-saving (PS)*. The protocols for infrastructure networks and ad hoc networks are different. Under an infrastructure network, there is an access point (AP) to monitor the mode of each mobile host. A host in the active mode is fully powered and thus may transmit and receive at any time. On the contrary, a host in the PS mode only wakes up periodically to check for possible incoming packets from the AP. A host always notifies its AP when changing modes. Periodically, the AP transmits *beacon frames* spaced by a fixed *beacon interval*. A PS host should monitor these frames. In each beacon frame, a *traffic indication map (TIM)* will be delivered, which contains ID's of those PS hosts with buffered unicast packets in the AP. A PS host, on hearing its ID, should stay awake for the remaining beacon interval. Under the contention period (i.e., DCF), a awake PS host can issue a PS-POLL to the AP to retrieve the buffered packets. While under the contention-free period (i.e., PCF), a PS host will wait for the AP to poll it. Spaced by a fixed number of beacon intervals, the AP will send *delivery TIMs (DTIMs)* within beacon frames to indicate that there are buffered broadcast packets. Immediately after DTIMs, the buffered broadcast packets will be sent.

Under an ad hoc network, PS hosts also wake up periodically. The short interval that PS hosts wake up is called the *ATIM window*. It is assumed that hosts are fully connected and all synchronized, so the ATIM windows of all PS hosts will start at about the same time. In the beginning of each ATIM window, each mobile host will contend to send a beacon frame. Any successful beacon serves as the purpose of synchronizing mobile

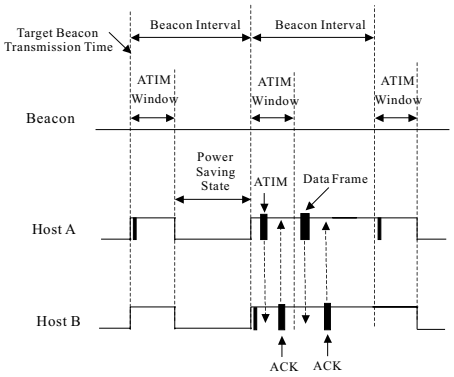


Fig. 1. An example of unicasting in an ad hoc networks with PS hosts.

hosts' clocks. This beacon also inhibits other hosts from sending their beacons. To avoid collisions among beacons, a host should wait a random number of slots between 0 and $2 \times CW_{min} - 1$ before sending out its beacon.

After the beacon, a host with buffered unicast packets can send a direct ATIM frame to each of its intended receivers in PS mode. ATIM frames are also transmitted by contention based on the DCF access procedure. After transmitting an ATIM frame, the mobile host shall remain awake for the entire remaining period. On reception of the ATIM frame, the PS host should reply with an ACK and remains active for the remaining period. The buffered unicast packets should be sent based on the normal DCF access procedure after the ATIM window finishes. If the sender doesn't receive an ACK, it should retry in the next ATIM window. As for buffered broadcast packets, the ATIM frames need not be acknowledged. Broadcast packets then can be sent based on contention after the ATIM window finishes. If a mobile host is unable to transmit its $ATIM$ frame in the current ATIM window or has extra buffered packets, it should retransmit ATIMs in the next ATIM window. To protect PS hosts, only RTS, CTS, ACK, Beacon, and ATIM frames can be transmitted during the ATIM window.

Figure 1 shows an example, where host A wants to transmit a packet to host B. During the ATIM window, an ATIM frame is sent from A to B. In response, B will reply with an ACK. After the ATIM window finishes, A can try to send out its data packet.

C. Problem Statement

The PS mode of IEEE 802.11 is designed for a single-hop (or fully connected) ad hoc network. When applied to a multi-hop ad hoc network, three problems may arise. All these will pose a demand of redesigning the PS mode for multihop MANET.

A) Clock Synchronization: Since IEEE 802.11 assumes that mobile hosts are fully connected, the transmission of a beacon frame can be used to synchronize all hosts' beacon intervals. So the ATIM windows of all hosts can appear at around the same time without much difficulty. In a multi-hop MANET, clock synchronization is a difficult job because communication delays and mobility are all unpredictable, especially when the network scale is large. Even if perfect clock synchronization is available, two temporarily partitioned sub-networks may independently enter PS mode and thus have different ATIM timing. With the clock-drifting problem, the ATIM windows of different hosts are not guaranteed to be synchronous. Thus, the ATIM window has to be re-designed.

B) Neighbor Discovery: In a wireless and mobile environment, a host can only be aware by other hosts if it transmits a signal that is heard by the others. For a host in the PS mode, not only is its chance to transmit reduced, but also its chance to hear others' signals. As reviewed above, a PS host must compete with other hosts to transmit its beacon. A host will cancel its beacon frame once it hears other's beacon frame. This may run into a dilemma that hosts are likely to have inaccurate neighborhood information when there are PS hosts. Thus, many existing routing protocols that depend on neighbor information may be impeded.

C) Network Partitioning: The above inaccurate neighbor information may lead to long packet delays or even network-partitioning problem. PS hosts with unsynchronized ATIM windows may wake up at different times and may be partitioned into several groups. These conceptually partitioned groups are actually connected. Thus, many existing routing protocols may fail to work in their route discovery process unless all hosts are awaken at the time of the searching process.

III. Power-Saving Protocols for MANET

In this section, we present three asynchronous power-saving protocols that allow mobile hosts to enter PS mode in a multihop MANET. According to the above discussion, we derive several guidelines in our design:

- *More Beacons:* To prevent the inaccurate-neighbor problem, a mobile host in PS mode should insist more on sending beacons. Specifically, a PS host should not inhibit its beacon in the ATIM window even if it has heard others' beacons. This will allow others to be aware of its existence. For this reason, our protocols will allow multiple beacons in a ATIM window.
- *Overlapping Awake Intervals:* Our protocols do not count on clock synchronization, To resolve this problem, the wake-up patterns of two PS hosts must overlap with each other no matter how much time their clocks drift away.
- *Wake-up Prediction:* When a host hears another PS host's beacon, it should be able to derive that PS host's wake-up pattern based on their time difference. This will allow the former to send buffered packets to the later in the future. Note that such prediction is not equal to clock synchronization since the former does not try to adjust its clock.

Based on the above guidelines, we propose three power-saving protocols, each with a different wake-up pattern for PS hosts. PS hosts' wake-up patterns do not need to be synchronous. For each PS host, it divides its time axis into a number of fixed-length intervals called *beacon intervals*. In each beacon interval, there are three windows called *active window*, *beacon window*, and *MTIM window*. During the active window, the PS host should turn on its receiver to listen to any packet and take proper actions as usual. The beacon window is for the PS host to send its beacon, while the MTIM window is for other hosts to send their MTIM frames to the PS host. Our MTIM frames serve the similar purpose as ATIM frames in IEEE 802.11; here we use MTIM to emphasize that the network is a mutil-hop MANET. Excluding these three windows, a PS host with no packet to send or receive may go to the sleep mode. Figure 2(a)

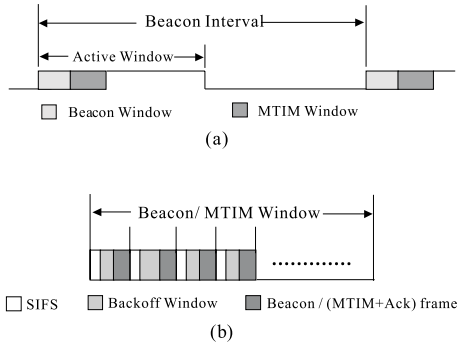


Fig. 2. Structure of a beacon interval: (a) active, beacon, and MTIM windows and (b) access procedure.

shows an example structure of a beacon interval.

The following notations are used throughout this paper:

- BI: length of a beacon interval
- AW: length of an active window
- BW: length of a beacon window
- MW: length of an MTIM window

We should comment at this point that the structure of a beacon interval may vary for different protocols (to be elaborated later). The illustration in Figure 2(a) is only one of the several possibilities. In the beacon window (resp., MTIM window), hosts can send beacons (resp., MTIM frames) following the DCF access procedure. Each transmission must be led by a SIFS followed by a random delay ranging between 0 and $2 \times CW_{min} - 1$ slots. This is illustrated in Figure 2(b).

A. Protocol 1: Dominating-Awake-Interval

The basic idea of this approach is to impose a PS host to stay awake sufficiently long so as to ensure that neighboring hosts can know each other and, if desire, deliver buffered packets. By "dominating-awake", we mean that a PS host should stay awake for at least about half of BI in each beacon interval. This guarantees any PS host's beacon window to overlap with any neighboring PS host's active window, and vice versa.

This protocol is formally derived as follows. When a host decides to enter the PS mode, it divides its time axis into fixed-length beacon intervals, each of length BI. Within each beacon, the lengths of all three windows (i.e., AW, BW, and MW) are constants. To satisfy the "dominating-awake" property, we enforce that $AW \geq BI/2 + BW$. The sequence of beacon intervals are alternatively labeled as *odd* and *even* intervals. Odd and even intervals have different structures as defined below (see the illustration in Figure 3).

- Each odd beacon interval starts with an active window. The active window is led by a beacon window followed by an MTIM window.
- Each even beacon interval also starts with an active window, but the active window is terminated by an MTIM window followed by a beacon window.

It is not hard to see that by imposing the active window occupying at least half of each beacon interval, we can guarantee that two hosts' active windows always have some overlapping. However, why we have different structures for odd and even beacon intervals remains obscure. Let's consider Figure 4,

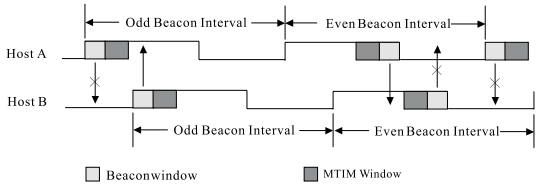


Fig. 3. Structures of odd and even intervals in the Dominating-Awake-Interval protocol.

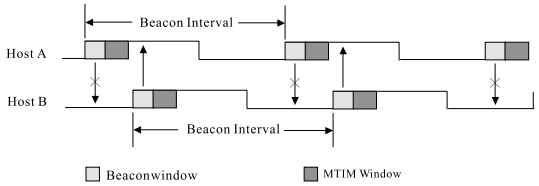


Fig. 4. An example where host B will always miss A's beacons.

where beacon windows always appear at the beginning of beacon intervals. In this case, host A can hear host B's beacons, but B always misses A's beacons. On the contrary, as Figure 3 shows, A can hear B's beacons at odd intervals, and B can hear A's beacons at even intervals.

Earlier we imposed the condition $AW \geq BI/2 + BW$. The following theorem provides a formal proof on the correctness of this protocol.

Theorem 1: The Dominating-Awake-Interval protocol guarantees that when $AW \geq BI/2 + BW$, a PS host's entire beacon window always overlaps with any neighboring PS host's active window in every other beacon interval, no matter how much time their clocks drift away.

Proof: The detail of the proof is in [23] ■

The above proof guarantees that a PS host is able to receive all its neighbors' beacon frames in every two beacon intervals, if there is no collision in receiving the latter's beacons. Since the response time for neighbor discovery is pretty short, this protocol is suitable for highly mobile environments.

B. Protocol 2: Periodically-Fully-Awake-Interval

The previous protocol requires PS hosts keep active more than half of the time, and thus is not energy-efficient. To reduce the active time, in this protocol we design two types of beacon intervals: *low-power intervals* and *fully-awake intervals*. In a low-power interval, the length of the active window is reduced to the minimum, while in a fully-awake interval, the length of the active window is extended to the maximum. Since fully-awake intervals need a lot of powers, they only appear periodically and are interleaved by low-power intervals. So the energy required can be reduced significantly.

Formally, when a host decides to enter the PS mode, it divides its time axis into fixed-length beacon intervals of length BI. The beacon intervals are classified as *low-power* and *fully-awake* intervals. The fully-awake intervals arrive periodically every T intervals, and the rest of the intervals are low-power intervals. The structures of these beacon intervals are defined as follows.

- Each low-power interval starts with an active window, which contains a beacon window followed by a MTIM window, such

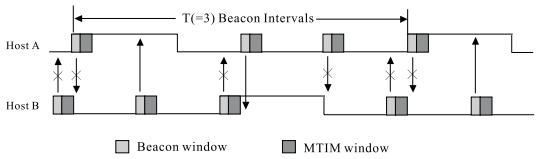


Fig. 5. An example of the Periodically-Fully-Awake-Interval protocol with fully-awake intervals arrive every $T = 3$ beacon intervals.

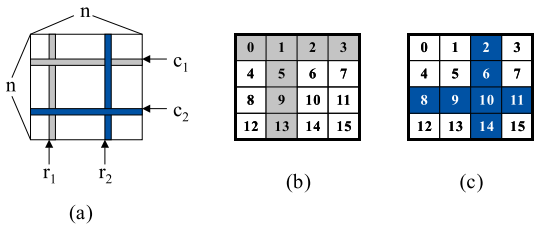


Fig. 6. Examples of the Quorum-Based Protocol (a)intersections of two PS hosts' quorum intervals, (b)host A's quorum intervals, and (c)host B's quorum intervals

that $AW = BW + MW$. In the rest of the time, the host can go to the sleep mode.

- Each fully-awake interval also starts with a beacon window followed by a MTIM window. However, the host must remain awake in the rest of the time, i.e., $AW = BI$.

Intuitively, the low-power intervals is for a PS host to send out its beacons to inform others its existence. The fully-awake intervals are for a PS host to discover who are in its neighborhood. It is not hard to see that a fully-awake interval always has overlapping with any host's beacon windows, no matter how much time their clocks drift away. By collecting other hosts' beacons, the host can predict when its neighboring hosts will wake up. Figure 5 shows an example with $T = 3$ intervals. So hosts A's and B's beacons always have chances to reach the other's active windows.

Theorem 2: The Periodically-Fully-Awake-Interval protocol guarantees that a PS host's beacon windows overlap with any neighbor's fully-awake intervals in every T beacon intervals, no matter how much time their clocks drift away.

Compared to the previous Dominating-Awake-Interval protocol, which requires a PS host to stay awake more than half of the time, this protocol can save more power as long as $T > 2$. However, the response time to get aware of a newly appearing host could be as long as T beacon intervals. So this protocol is more appropriate for slowly mobile environments. One way to reduce the response time is to decrease the value of T to fit one's need.

C. Protocol 3: Quorum-Based

In the previous two protocols, a PS host has to contend to send a beacon in each beacon interval. In this section, we propose a protocol based on the concept of *quorum*, where a PS host only needs to send beacons in $O(1/n)$ of the all beacon intervals. Thus, when transmission takes more powers than reception, this protocol may be more energy-efficient. The concept of quorums has been used widely in distributed system design (e.g., to guarantee mutual exclusion [24], [25], [26], [27]). A quorum is a set of identities from which one has to obtain permission to perform some action [24]. Typically, two quorum sets always have nonempty intersection so as to guarantee the atomicity of a transaction. Here we adopt the concept of quorum to design PS hosts' wakeup patterns so as to guarantee a PS host's beacons can always be heard by others' active windows. This is why our protocol is named so.

The quorum structure of our protocol is as follows. The sequence of beacon intervals is divided into sets starting from the first interval such that each continuous n^2 beacon intervals are called a *group*, where n is a global parameter. In each group, the n^2 intervals are arranged as a 2-dimensional $n \times n$ array in a row-major manner. On the $n \times n$ array, a host can arbitrarily pick one column and one row of entries and these $2n - 1$ intervals are called *quorum intervals*. The remaining $n^2 - 2n + 1$ intervals are called *non-quorum intervals*.

Before proceeding, let's make some observation from the quorum structure. Given two PS hosts that are perfectly time-synchronized, it is not hard to see that their quorum intervals always have at least two intersecting beacon intervals(see the illustration in Figure 6(a)). This is due to the fact that a column and a row in a matrix always have an intersection. Thus, two PS host may hear each other on the intersecting intervals. However, the above reasoning is not completely true since we do not assume that hosts are time-synchronized (the formal proof is in Theorem 3). For example, in Figure 6(b) and (c), host A selects intervals on row 0 and column 1 as its quorum intervals from a 4×4 matrix, while host B selects intervals on row 2 and column 2 as its quorum intervals. When perfectly synchronized, intervals 2 and 9 are the intersections.

The structures of quorum and non-quorum intervals are formally defined below.

- Each quorum interval starts with a beacon window followed by an MTIM window. After that, the host must remain awake for the rest of the interval, i.e., $AW = BI$.
- Each non-quorum interval starts with an MTIM window. After that, the host may go to the sleep mode, i.e., we let $AW = MW$.

Theorem 3: The Quorum-Based protocol guarantees that a PS host always has at least two entire beacon windows that are fully covered by another PS host's active windows in every n^2 beacon intervals.

Proof: The detail of the proof is in [28] ■

The Quorum-Based protocol has advantage in that it only transmits in $O(1/n)$ of the beacon intervals(on the contrary, the earlier two protocols have to transmit a beacon in every interval). In addition, it also keeps awake in $O(1/n)$ of the time. As long as $n \geq 4$, this amount of awaking time is less than 50%. So this protocol is more energy-efficient when transmission cost is high. The backside is that a PS host may learn its vicinity at lower speed.

D. Summary

Table I summarizes the characteristics of the three proposed power-saving protocols. "Number of beacons" indicates the average number of beacons that a host need to transmit in each beacon interval, "Active ratio" indicates the ratio of time that a PS host needs to stay awake while in the PS mode, and

"Neighbor sensitivity " indicates the average time that a PS host takes to hear a neighbor's beacon. As Table I shows, the Quorum-Based protocol spends least power in transmitting beacons. The Periodically-Fully-Awake-Interval and the Quorum-based protocols' active ratios can be quite small as long as T and n, respectively are large enough. The Dominated-Awake-Interval protocol is most sensitive to neighbor changes, while the Quorum-based protocol is least sensitive.

IV. Communication Protocols for Power-Saving Hosts

This section discusses how a host sends packets to a neighboring PS host. Since the PS host is not always active, the sending host has to predict when the PS host will wake up, i.e., when the latter's $MTIM$ windows will arrive. To achieve this, each beacon packet has to carry the clock value of the sending host so as for other hosts to calculate their time differences. Table II summarizes when $MTIM$ windows arrive in the proposed protocols.

After correctly predicting the receiving side's $MTIM$ windows, the sending side can contend to send $MTIM$ packets to notify the receiver, after which the buffered data packet can be sent. Below, we discuss how unicast and broadcast are achieved.

A. Unicast

This is similar to the procedure in IEEE 802.11's PS mode. During the receiver's $MTIM$ window, the sender contends to send its $MTIM$ packet to the receiver. The receiver, on receiving the MTIM packet, will reply an ACK after SIFS and stay awake in the remaining of the beacon interval. After the MTIM window, the sender will contend to send the buffered packet to the receiver based on the DCF procedure.

B. Broadcast

The situation is more complicated for broadcasting since the sender may have to deal with multiple asynchronous neighbors. To reduce the number of transmissions, we need to divide these asynchronous neighbors into groups and notify them separately in multiple runs. The steps are described below. Note that here the broadcast is not designed to be 100% reliable at the MAC layer (reliable broadcast may be supported at a higher layer).

When a source host S intends to broadcast a packet, it first checks the arrival time of the MTIM windows of all its neighbors. Then S picks the host, say Y, whose first MTIM window arrives earliest. Based on Y's first MTIM window, S further picks those neighbors whose MTIM windows have overlapping with Y's first MTIM window. These hosts, including Y, are groups together and S will try to notify them in one MTIM frame (note that such MTIM frames need not be acknowledged due to the unreliable assumption). After this notification, S considers the rest of the neighbors that have not been notified yet in the previous MTIM and repeats the same procedure again to initiate another MTIM frame. The process is repeated until all its neighbors have been notified.

A neighbor, on receiving a MTIM carrying a broadcast indication, should remain awake until a broadcast packet is received or a timeout value expires (here we recommend a timeout value of two beacon intervals be used, but this can also be a adjustable parameter during system configuration). The source S, after notifying all neighbors, can contend to send its buffered broadcast packet after the last neighbor's MTIM window passes. Broadcast packets should be sent based on the DCF procedure too.

V. Simulation Experiments

To evaluate the performance of the proposed power-saving protocols, we have developed a simulator using C. In the simulations, we assume that the transmission radius is 250 meters and the transmission rate is 2M bits/sec. The MAC part basically follows the IEEE 802.11 standard [12], except the power management part. We intend to model one central host with the possibility of multiple mobile hosts approaching or leaving it (in the experiment, we use four neighbors.) We use an "on-off" model to simulate the mobility of the surrounding hosts. Specially, in every 5 seconds, a surrounding host chooses to enter an "on" or an "off" state. An "on" state indicates that the host is within the central host's transmission range, while an "off" state indicates that it is out of the range. The choice is based on a probability distribution. Three parameters are tunable in our simulations:

- Traffic load: a Poisson distribution for unicast/broadcast with rate between 5 $\sim$ 30 packets/sec.
- "On" probability: a uniform distribution between 50% $\sim$ 100%.
- Beacon interval: length of one beacon interval between 100 ms $\sim$ 500 ms.

Each simulation lasts for 100 seconds. Each result is obtained from the average of 1000 simulation runs. For simplicity, we assume that all hosts are in the PS mode. To make comparison, we also simulate an "always-active" scheme in which all hosts are active all the time.

Three performance metrics are used to evaluate our power-saving protocols:

- power consumption: the average power consumption per mobile host throughout one simulation run.
- power efficiency: the average power consumption for each successful packet transmission.
- neighbor discovery time: average time to discover a newly approaching neighbor.

The power model in [29] is adopted, which is obtained by real experiments on Lucent WaveLAN cards. Table III summarizes the power consumption parameters used in our simulations. Sending/receiving a unicast/broadcast packet has a cost $P_{base} + P_{byte} \times L$, where P_{base} is the power consumption independent of packet length, P_{byte} is the power consumption per byte, and L is the packet length. When sending a packet of the same size, unicast consumes more power than broadcast because it needs to send and receive extra control frames (RTS, CTS, and ACK). The last two entries indicate the consumption when a host has no send/receive activity and is in the active mode and PS mode, respectively. As can be seen, staying in active mode is much more energy-consuming. The traffic-related parameters are in Table IV.

In the following subsections, we show how beacon interval, mobility, and traffic load affect the performance of the proposed power-saving protocols. For simplicity, the Dominating-Awake-Interval protocol is denoted as D, the Periodically-Fully-Awake-

TABLE I

CHARACTERISTICS OF THE PROPOSED POWER-SAVING PROTOCOLS

Protocol	Number of beacons	Active ratio	Neighbor sensitivity
Dominated-Awake	1	$1/2 + BW/BI$	BI
Periodically-Fully-Awake	1	$1/T$	$T \times BI/2$
Quorum-Based	$(2n-1)/n^2$	$(2n-1)/n^2$	$(n^2/4) \times BI$

TABLE II

TIMING OF $MTIM$ WINDOWS OF THE PROPOSED PROTOCOLS.

Protocol	$MTIM$ window's timing
Dominated-Awake	$[(2m+1) \times BI + BW, (2m+1) \times BI + BW + MW]$ (odd int.) $[2m \times BI + BI/2 - MW, 2m \times BI + BI/2]$ (even int.)
Periodically-Fully-Awake	$[m \times BI + BW, m \times BI + BW + MW]$
Quorum-Based	$[m \times BI + BW, m \times BI + BW + MW]$ (quorum int.) $[m \times BI, m \times BI + MW]$ (non-quorum int.)

TABLE III

POWER CONSUMPTION PARAMETERS USED IN THE SIMULATION

Unicast send	$454 + 1.9 \times L\ \mu W$
Broadcast send	$266 + 1.9 \times L\ \mu W$
Unicast receive	$356 + 0.5 \times L\ \mu W$
Broadcast receive	$56 + 0.5 \times L\ \mu W$
Idle	843 $\mu W/ms$
Doze	27 $\mu W/ms$

TABLE IV

TRAFFIC-RELATED PARAMETERS USED IN THE SIMULATION

Unicast packet size	2048 bytes
Broadcast packet size	256 bytes
Beacon window size	8 ms
MTIM window size	16 ms

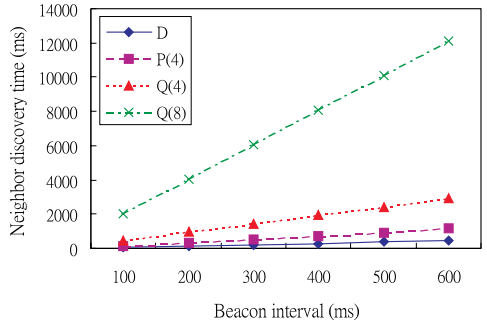


Fig. 7. Neighbor discovery time vs. beacon interval length. (traffic load = 10 packets, "on" probability = 80%)

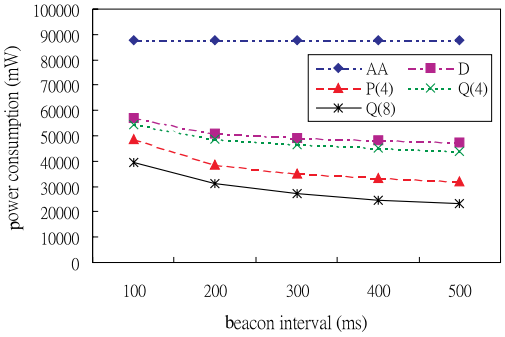


Fig. 8. Power consumption for unicast vs. beacon interval length. (traffic load = 10 packets/sec, "on" probability = 80%)

Interval protocol with parameter T is denoted as $P(T)$, the Quorum-Based protocol with parameter n is denoted as $Q(n)$, and the "always active" scheme is denoted as AA.

A. Impact of Beacon Interval Length

The length of beacon intervals has impact on hosts' sensitivity to environmental changes and power consumptions. However, these are contradicting factors. To observe its impact, we vary the beacon interval length between 100 ms to 500 ms. As Figure 7 shows, longer beacon intervals only slightly increase the neighbor discovery time for schemes D and $P(4)$, but have more significant impact on schemes $Q(4)$ and $Q(8)$. Overall, scheme D has the shortest neighbor discovery time, which is subsequently followed by $P(4), Q(4)$, and $Q(8)$.

Figure 8 and Figure 9 show the power consumption at various beacon interval lengths for unicast and broadcast, respectively. In both cases, longer beacon intervals do incur less power consumption. From the curves, we would suggest the beacon interval be set at around 300 ms, since a larger interval will only contribute to a little more saving in power consumption. Overall, scheme $Q(8)$ has the smallest power consumption, which is subsequently followed by $P(4)$, $Q(4)$, D, and AA.

Comparing the above, we would conclude that $P(4)$ is the best choice since it minimizes neighbor discovery time as well as power consumption. Scheme D is useful in highly mobile environment when power consumption is a less important issue, while scheme $Q(8)$ is very power-efficient when being applied to a low-mobility environment.

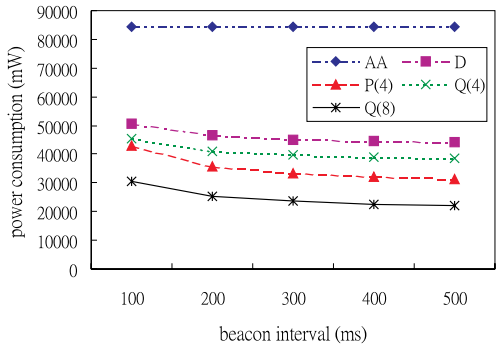


Fig. 9. Power consumption for broadcast vs. beacon interval length. (traffic load = 10 packets/sec, "on" probability = 80%)

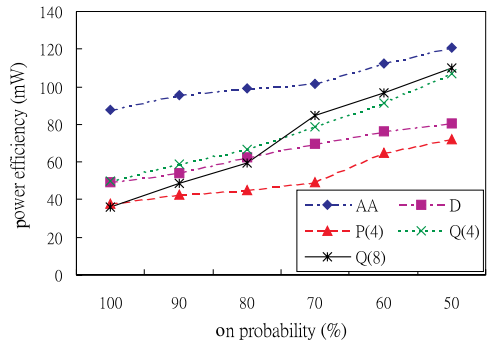


Fig. 10. Power efficiency for unicast vs. "on" probability. (traffic load = 10 packets/sec, beacon interval = 300 ms)

B. Impact of Mobility

We use an "on-off" model to simulate the mobility of the surrounding hosts. Intuitively, a lower "on" probability implies higher mobility. When the mobility is high, a PS host may not be able to keep accurate neighborhood information, leading to a higher chance of sending useless "orphan" packets and thus wasting powers.

Figure 10 shows the impact of "on" probability on power efficiency. When the "on" probability is less than 70%, the power efficiency will increase sharply for all schemes. This is because inaccuracy neighborhood information mislead a host keeping on polling a missing host by sending many $MTIM$ packets. The efficiency of $Q(8)$ and $P(4)$ are the best at higher "on" probability. However, at lower "on" probability, $P(4)$ will outperform $Q(8)$ (because $Q(8)$'s sensitivity to neighborhood change will reduce).

The simulation result for broadcasting is show in Figure 11. The major difference is that the power efficiency is quite independent of the "on" probability. The reason is that broadcast is unreliable; based on our assumption, a broadcast packet is counted as successful as long as some neighbors are there to receive the packet. Thus, there are less useless transmissions. The trend is similar — $Q(8)$ performs the best, which is subsequently followed by $P(4), Q(4), D$, and AA.

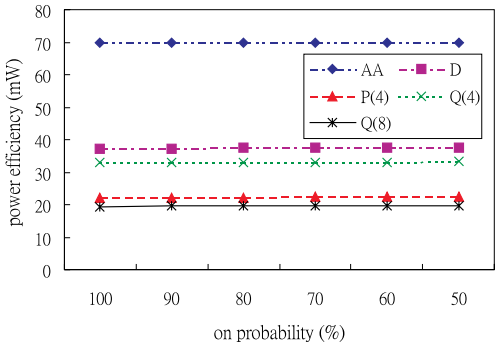


Fig. 11. Power efficiency for broadcast vs. "on" probability. (traffic load = 10 packets/sec, beacon interval = 300 ms)

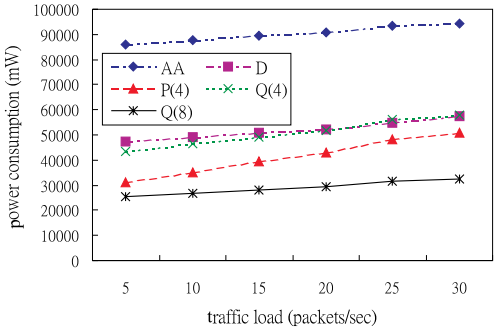


Fig. 12. Power consumption for unicast vs. traffic load. ("on" probability = 80%, beacon interval = 300 ms)

C. Impact of Traffic Load

In this experiment, we vary the traffic load to observe the effect. Figure 12 shows the power consumption for unicast traffic. A higher traffic load incurs higher power consumption, which is reasonable since hosts have less chance to sleep. Figure 13 shows the power consumption for broadcast traffic. The increase of power consumption due to increase of traffic load is almost unnoticeable because the broadcast packets being injected are quite small.

To observe from a different angle, Figure 14 and Figure 15 show the power efficiency at different traffic loads. A higher load makes transmitting a packet less costly for both unicast and broadcast traffics because multiple packets may be transmitted in one beacon interval. At lower traffic load, the idle time for hosts increases, thus wasting more power. Again, $P(4)$ and $Q(8)$ are most power-efficient, which are followed subsequently by $Q(4), D$, and AA.

VI. Conclusions

In this paper, we have addressed the power management problem in a MANET which is characterized by unpredictable mobility, multi-hop communication, and no clock synchronization. We have pointed out two important issues, the *neighbor discovery* problem and the *network-partitioning* problem, which may occur in such an environment if one directly adopts the power-saving (PS) mode defined in the IEEE 802.11 protocol. As far

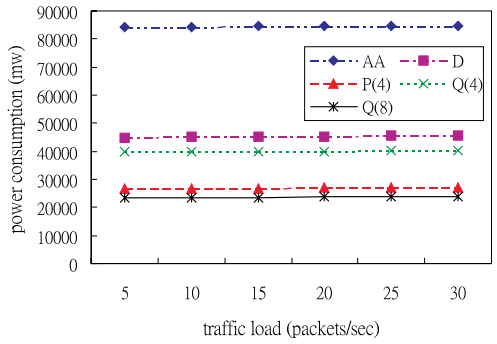


Fig. 13. Power consumption for broadcast vs. traffic load. ("on" probability = 80%, beacon interval = 300 ms)

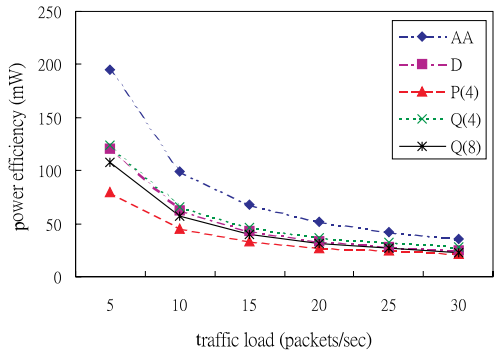


Fig. 14. Power efficiency for unicast vs. traffic load. ("on" probability = 80%, beacon interval = 300 ms)

as we know, the power-saving issues, particularly for multi-hop MANETs, have not been addressed seriously in the literature. In this paper, we have proposed three power-saving protocols for IEEE 802.11-based, multi-hop, unsynchronized MANETs. The protocols can each guarantee an upper bound on packet delay if there is no collision in the beacon window (but collision is inevitable in any random-access network). Simulation results have shown that our power-saving protocols can save lots of power with reasonable neighbor discovery time. Among the three proposed protocols, the Dominating-Awake-Interval protocol is most power-consuming but has the lowest neighbor discovery time, while the Quorum-based protocol is most power-saving but has the longest neighbor discovery time. They are appropriate for highly mobile and lowly mobile environments, respectively. The Periodical-Fully-Awake-Interval protocol can balance both power consumption and neighbor discovery time, and thus may be used in most typical environments. We believe that the proposed protocols can be applied to current IEEE 802.11 wireless LAN cards easily with little modification.

REFERENCES

[1] Nokia, "Wireless Broadband Access–Nokia Rooftop Solution," *Nokia Network References, http://www.wbs.nokia.com/solution/index.html*, 2001.

[2] S. L. Wu, Y. C. Tseng, and J. P. Sheu, "Intelligent Medium Access for Mobile Ad Hoc Networks with BusyTones and Power Control," *IEEE Journal on Selected Areas in Communications*, vol. 18, pp. 1647–1657, Sep 2000.

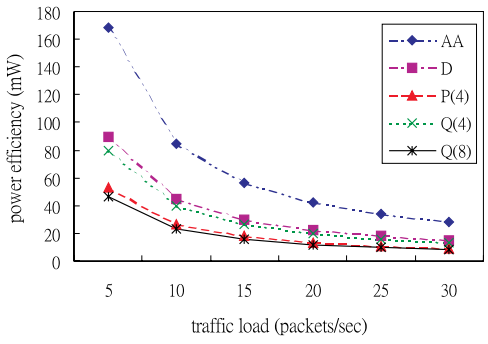


Fig. 15. Power efficiency for broadcast vs. traffic load. ("on" probability = 80%, beacon interval = 300 ms)

[3] L. Hu, "Topology Control for Multihop Packet Radio Networks," *IEEE Transactions on Communications*, vol. 41, pp. 1474–1481, Oct 1993.

[4] R. Ramanathan and R. Rosales-Hain, "Topology Control of Multihop Wireless Networks using Transmit Power Adjustment," *IEEE INFOCOM*, pp. 404–413, 2000.

[5] R. Wattenhofer, L. Li, P. Bahl, and Y. M. Wang, "Distributed Topology Control for Power Efficient Operation in Multihop Wireless Ad Hoc Networks," *IEEE INFOCOM*, pp. 1388–1397, 2001.

[6] C. F. Huang, Y. C. Tseng, S. L. Wu, and J. P. Sheu, "Increasing the Throughput of Multihop Packet Radio Networks with Power Adjustment," *International Conference on Computer, Cummunication, and Networks*, 2001.

[7] J. Gomez, A. T. Campbell, M. Naghshineh, and C. Bisdikian, "A Distributed Contention Control Mechanism for Power Saving in random-access Ad-Hoc Wireless Local Area Networks," *Proc. of IEEE International Workshop on Mobile Multimedia Communications*, pp. 114–123, 1999.

[8] J. H. Ryu and D. H. Cho, "A New Routing Scheme Concerning Power-Saving in Mobile Ad-Hoc Networks," *Proc. of IEEE International Conference on Communications*, vol. 3, pp. 1719–1722, 2000.

[9] J. H. Ryu, S. Song, and D. H. Cho, "A Power-Saving Multicast Routing Scheme in 2-tir Hierarchical Mobile Ad-Hoc Networks," *Proc. of IEEE Vehicular Technology Conference*, vol. 4, pp. 1974–1978, 2000.

[10] S. Singh, M. Woo, and C. S. Raghavendra, "Power-Aware Routing in Mobile Ad Hoc Networks," *Proc. of the International Conference on Mobile Computing and Networking*, pp. 181–190, 1998.

[11] I. Stojmenovic and X. Lin, "Power-aware Localized Routing in Wireless Networks," *Proc. of IEEE International Parallel and Distributed Processing Symposium*, pp. 371–376, 2000.

[12] LAN MAN Standards Committee of the IEEE Computer Society, "IEEE Std 802.11-1999, Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications," *IEEE*, 1999.

[13] H. Woesner, J. P. Ebert, M. Schlager, and A. Wolisz, "Power-Saving Mechanisms in Emerging Standards for Wireless LANs: The MAC Level Perspective," *IEEE Persinal Communications*, pp. 40–48, Jun 1998.

[14] J. C. Haartsen, "The Bluetooth Radio System," *IEEE Persinal Communications*, pp. 28–36, Feb 2000.

[15] S. Singh and C. S. Raghavendra, "Power Efficient MAC Protocol for Multihop Radio Networks," *Proc. of IEEE International Personal, Indoor and Mobile Radio Communications Conference*, pp. 153–157, 1998.

[16] C. F. Chiasserini and R. R. Rao, "A Distributed Power Management Policy for Wireless Ad Hoc Networks," *IEEE Wireless Communication and Networking Conference*, pp. 1209–1213, 2000.

[17] J. R. Lorch and A. J. Smith, "Software Strategies for Portable Computer Energy Management," *IEEE Personal Communications*, pp. 60–73, Jun 1998.

[18] A. K. Salkintzis and C. Chamzas, "An In-Band Power-Saving Protocol for Mobile Data Networks," *IEEE Transactions on Communications*, vol. 46, pp. 1194–1205, Sep 1998.

[19] T. Simunic, H. Vikalo, P. Glynn, and G. D. Micheli, "Energy Efficient Design of Portable Wireless Systems," *Proc. of the International Symposium on Low Power Electronics and Design*, pp. 49–54, 2000.

[20] B. Chen, K. Jamieson, H. Balakrishnan, and R. Morris, "Span: An Energy-Efficient Coordination Algorithm for Topology Maintenance in Ad Hoc Wireless Networks," *Proc. of the International Conference on Mobile Computing and Networking*, pp. 85–96, 2001.

[21] Y. Xu, J. Heidemann, and D. Estrin, "Geography-informed Energy Conservation for Ad Hoc Routing," *Proc. of the International Conference on Mobile Computing and Networking*, pp. 70–84, 2001.
[22] A. K. Salkintzis and C. Chamzas, "Performance Analysis of a Downlink MAC Protocol with Power-Saving Support," *IEEE Transactions on Vehicular Technology*, vol. 49, pp. 1029–1040, May 2000.
[23] C. S. Hsu, "The Correctness of the Dominating-Awake-Interval Protocol," *http://axp3.csie.ncu.edu.tw/∼ cshsu/proof1.html*, 2001.
[24] H. Garcia-Molina and D. Barbara, "How to Assign Votes in a Distributed Systems," *Journal of the ACM*, vol. 32, pp. 841–860, Oct 1985.
[25] D. Agrawal and A. El Abbadi, "An Efficient and Fault-Tolerance Algorithm for Distributed Multual Exclusion," *ACM Transactions on Computer Systems*, vol. 9, pp. 1–20, Feb 1991.
[26] A. Kumar, "Hierarchical Quorum Consensus: A New Algorithm for Managing Replicated Data," *IEEE Transactions on Computers*, vol. 40, pp. 996–1004, Sep 1991.
[27] Y. C. Kuo and S. T. Huang, "A Geometric Approach for Constructing Coteries and k-Coteries," *IEEE Transactions on Parallel and Distributed Systems*, vol. 8, pp. 402–411, Apr 1997.
[28] C. S. Hsu, "The Correctness of the Quorum-Based Protocol," *http://axp3.csie.ncu.edu.tw/∼ cshsu/proof3.html*, 2001.
[29] L. M. Feeney and M. Nilsson, "Investigating the Energy Consumption of Wireless Network Interface in an Ad Hoc Networking Environment," *IEEE INFOCOM*, pp. 1548–1557, 2001.

論文名稱：Fully Power-aware and Location-aware Protocols for Wireless Multi-Hop Ad Hoc Networks

作者：Y.-C. Tseng and T.-Y. Hsie

發表情況: ***ICCCN***, 2002

Fully Power-Aware and Location-Aware Protocols for Wireless Multi-hop Ad Hoc Networks*

Yu-Chee Tseng and Ten-Yueng Hsieh

Abstract

A *mobile ad hoc network (MANET)* is one consisting of a set of mobile hosts which can operate independently without infrastructure base stations. Power saving is a critical issue for MANET since most mobile hosts will be operated by battery powers. In this paper, we address the power-saving issue for IEEE 802.11-based MANETs from several protocol layers, including physical, MAC, and network layers. Our solution is *fully power-aware* and *location-aware* in the sense that it exploits location information of mobile hosts to achieve energy conservation on all these protocol layers. In comparison, existing protocols only exploit location information in limited layers (e.g., power control is covered in [9, 18, 27], power mode management in [3, 15, 30], power-aware MAC in [2, 19, 29], and power-aware routing in [5, 20, 23]). Similar to cellular networks, our approach is based on partitioning the network area into squares/hexagons, thus leading to a powerful energy and mobility management capability.

1 Introduction

Computing and communication anytime, anywhere is a global trend in today's development. Ubiquitous computing has been made possible by the advance of wireless communication technology and the availability of many light-weight, compact, portable computing devices. Among the various network architecture, the design of *mobile ad hoc network (MANET)* has attracted a lot of attention recently. A MANET is one consisting of a set of mobile hosts which can communicate with one another and roam around at their will. No base stations are supported in such an environment, and mobile hosts may have to communicate with each other in a *multi-hop* fashion. Applications of MANETs occur in situations like battlefields, major disaster areas, and outdoor assemblies. A working group called "manet" has been formed by the Internet Engineering Task Force (IETF) to study the related issues and stimulate research in MANET.

One critical issue for almost all kinds of portable devices supported battery energy is *power saving*. Without power, any mobile device will become useless. Battery energy is a limited resource, and it is expected that battery technology is not likely to progress as fast as computing and communication technologies do. Hence, how to extend the lifetime of batteries is an important issue, especially for MANET, which consists of mobile hosts only.

This paper investigates the design of power-conserving and location-aware protocols for MANETs. A protocol is called *power-aware* if it is designed with power saving in mind. A protocol's behavior does have significant impact on power consumption [2, 4, 30]. For example, the Lucent ORiNOCO WLAN PC Card consumes 0.05, 0.9, and 1.4 watts/sec under the doze, receive, and send modes, respectively[1]. Further, a MANET protocol is called *location-aware* if it tries to exploit the physical positions of mobile hosts. Typically, the location of a mobile host can be collected by attaching a GPS receiver to it. We note that applying location information in MANETs is a very natural way to improve MANET's performance since the connectivity of mobile hosts can be reflected in a 3D physical area, as opposed to wireline networks, where hosts are connected by multitudes of cables. Indeed, a lot of location-aware protocols for MANETs have been proposed [14, 16, 25].

Our solution is *fully power-aware* and *location-aware* in the sense that it exploits location information of mobile hosts to achieve energy conservation on all protocol layers including physical, MAC, and network layers. In comparison, existing protocols only exploit location information in limited layers (e.g., power control is covered in [9, 27, 18], power mode management in [3, 15, 30], power-aware MAC in [2, 19, 29], and power-aware routing in [5, 20, 23]). Similar to cellular networks, our approach is based on partitioning the network area into squares/hexagons, thus leading to strong energy and mobility management capability. To choose routing paths, we define the collective energy level of each square/hexagon and choose routes accordingly. To save more powers, a power mode management mechanism is proposed to reduce the number of awake hosts while keeping the connectivity of the network. To reduce unnecessary collisions (and thus save powers), we separate inter- from intra-square/hexagon communications in the MAC

*This work is co-sponsored by the Lee and MTI Center for Networking Research at the National Chiao Tung University, and the Program of Excellence Research, Ministry of Education, Taiwan (contract no. 89-E-FA04-4, 89-E-FA04-1-4, and 89-H-FA07-1-4). Part of this work was done while Y.-C. Tseng was supported by the Institute of Information Science, Academia Sinica, Taiwan, as a visiting scholar. Authors are with the Dept. of Computer Science and Infor. Engr., National Chiao Tung Univ.,Taiwan; Email:yctseng@csie.nctu.edu.tw, tyhsieh@csie.nctu.edu.tw.

[1] Data collected from http://www.agere.com, dated September 13, 2000.

layer. Finally, power control is adopted in the physical layer based on hosts' locations.

The rest of this paper is organized as follows. Section 2 reviews related works. The proposed protocols are presented in section 3. Section 4 concludes this paper.

2 Review of Power-Aware and Location-Aware Protocols

Earlier we have defined how a protocol is power-/location-aware or not. Based on the ISO model, we can further look at this issue from different protocol layers In the following, we review existing layer-1 to 3 protocols in a top-down manner. Table 1 summarizes our categorization.

Routing is typically addressed on layer 3. In [5], it is suggested that a mobile host with battery level below a certain threshold do not participate in routing activities so as to extend the lifetime of low-energy mobile hosts. Reference [20] proposes five different metrics, which can judge the power cost to route a packet. In [23], several power-aware routing policies are compared and a routing policy that can protect lower-energy mobile hosts is proposed. All these protocols are power-aware, but not location-aware. Location-aware, but non-power-aware, routing protocols also exist [10, 12, 14, 16]. The routing protocol proposed in [21] concerns both hosts' energies as well as the distances between neighboring hosts. Location information is used in [31] to choose routing paths with lowest power costs.

Power mode management is typically addressed on layer 2. The standard IEEE 802.11 [17] allows a host to enter a power-saving (PS) mode to reduce power consumption. The Bluetooth standard supports four power modes: active, hold, park, and sniff [8]. In [24], it is pointed out that in a *multi-hop* ad hoc networks, three problems, namely clock synchronization, neighboring discovery, and network partitioning, may arise. Several power-mode management protocols are then proposed to resolve these problems. Power management concerning transmission delays and network throughput is addressed in [3, 15]. GPS is adopted in [30] to partition the network into small squares called *grids*. In each grid, a node will be elected as the access point, which should stay awake, and the other nodes can go to sleep. Reference [2] proposes to construct a *virtual backbone*, which is a minimum spanning tree that can connect all nodes. An internal node in the spanning tree has to stay awake all the time, while a leaf node can switch to a sleep mode. One control and one data channels are used in [19], and the latter is turned on only when necessary.

Since MAC is closely related to the underlying physical layer, their power issue are sometimes addressed together. The work in [29] discusses a busy-tone-based protocol, which tries to conduct power control on RTS, CTS, and data packets. Through power control, interference is reduced and network throughput is improved. In a packet radio network, [28] discusses how to adjust hosts' transmission powers to improve network topology based on locations of stations. Topology control is addressed in [9, 18, 27], where centralized/distributed rules are used by hosts to adjust their transmission powers with certain target topologies in mind. Power control, together with routing, is addressed in [26].

Table 1: Categorization of protocols based on power/location awareness.

Protocol	routing	power mode	MAC	power cnt'l
[5, 20, 23]	PA			
[10, 12, 14, 16]	LA			
[21, 31]	PA+LA			
[3, 15, 24]		PA		
[30]		PA+LA		
[2, 19]		PA	PA	
[29]			PA	PA
[28]			PA+LA	PA+LA
[9, 18, 27]				LA+PA
[26]	LA+PA			LA+PA
ours	LA+PA	LA+PA	LA+PA	LA+PA

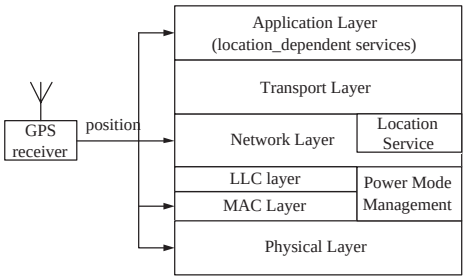


Figure 1: The proposed fully power-aware and location-aware protocol architecture.

3 A Fully Power-aware and Location-aware Protocol

3.1 Basic Idea

As reviewed above, most existing works have addressed the power-saving issue in MANET on some, but not all, of layers 1, 2, and 3. Our goal is to propose an integrated protocol that is power-aware and location-aware in routing, power mode management, medium access, and physical layer.

Our work can be considered as an integration and extension of [14, 30]. Supposing that each host has a GPS receiver, [14] proposes to partition the network area into squares called *grids*. This routing protocol is location-aware in route discovery, packet relay, and route maintenance. The related routing cost is significantly reduced, and route lifetime is greatly extended. However, the power issue is not addressed. Based on the similar partitioning concept, [30] suggests that only few hosts in each grid need to stay awake, and the remaining hosts can go to a sleep mode. Power conservation is thus achieved.

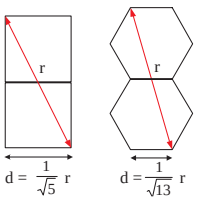


Figure 2: The relation of transmission distance, r, and grid side length, d, to guarantee inter-leader communication.

This paper extends the idea in [14, 30] and shows how location information can be utilized for full-scale power conservation in layers 1, 2, and 3. Fig. 1 illustrates such concept. The work is characterized by the following features:

- By partitioning the physical area into grids, the collective energy in each sub-area is calculated and routes are likely to pass those with higher collective energies. By so doing, routes are more stable and suffer less breakage probability.
- Also by having grids, only one higher-energy host needs to stay awake in each sub-area. Other hosts can go to sleep without worrying missing packets.
- In the medium access part, we separate intra- from inter-grid communications into different times to improve channel utilization and reduce unnecessary contention. By so doing, transmission power control can be easily achieved.

3.2 Network Layer

Similar to cellular networks, the geographic area of the MANET is partitioned into virtual cells called *grids*. From its current location, each mobile is able to tell which grid it is currently resident. In each grid, a leader will be elected based on energy concerns. Only the leader has to stay awake, and the other hosts without sending/receiving activities may go to sleep. The leader will be responsible for routing, relaying packets, and maintaining the correct operation of other hosts in low-power mode. The detailed leader election protocol will be discussed in 3.3.

The shape of a grid can be a square or a hexagon. Each grid is denoted by (i, j) based on a 90°- or 60°-xy-coordinate system for the square and hexagon cases, respectively. Let d be the side length of grids and r be the transmission distance of each radio. Since the leader of each grid is responsible of relaying packets to neighboring grids' leaders, the farthest distance between two neighboring leaders must satisfy $d \geq \frac{r}{\sqrt{5}}$ and $d \geq \frac{r}{\sqrt{13}}$, for square- and hexagon-shape grids, respectively. This is illustrated in Fig. 2.

Below, we discuss two possible ways to conduct packet routing: *connection-oriented* and *connectionless*. Then we discuss the important location service to support location discovery.

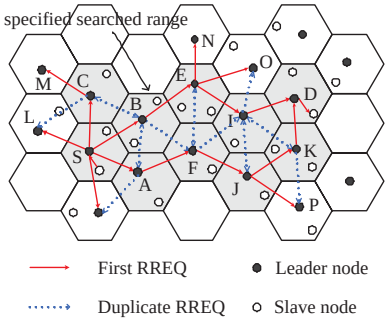


Figure 3: A route search example for connection-oriented routing.

3.2.1 Connection-Oriented (Stateful) Routing

For connection-oriented routing, routing tables have to be constructed before data packets are sent. We adopt *on-demand* routing. Our design has two features. First, locations of the source and the destination are used to confine the searching range to reduce the discovery cost and to save energy of irrelevant hosts. Second, routes will be selected based on energy concerns.

For each host i, its remaining energy at time t is denoted by c_i^t, which is a value between 0 and 1. A larger value represents more energy, and 1 means a fully charged battery. For each grid (i, j), its *collective energy* at time t is written as $C_{i,j}^t$, which is the sum of all battery energies of the hosts within it, i.e., $C_{i,j}^t = \sum_{k \in (i,j)} c_k^t$.

To discover a route from a source S to a destination D, S sends a RREQ to its grid leader, if S itself is not the leader. The leader then broadcasts the RREQ to its neighboring grid leaders. Note that leaders of non-adjacent grids may hear the RREQ. This causes no problem, and the following steps will still be executed. On receiving the RREQ, each leader will check whether D belongs to the local grid or not. If so, the leader will notify D, if it is sleeping. Otherwise, the leader increases the hop count by one, appends a record to the RREQ packet, and rebroadcasts the RREQ further to its neighboring leaders. The appended record includes the leader's grid id (i, j) and the current $C_{i,j}^t$. To avoid endless looping, a duplicate RREQ should not be rebroadcast. Also, using S and D, the search range should be confined. There are several alternatives to define the search range [14]. Fig. 3 illustrates an example of RREQ propagation.

Each received RREQ by D represents a route. If multiple routes exist, D may choose a proper one for use. We propose a heuristic that balances both power and throughput. First, D sorts all potential routes based on their route energies, where the *route energy* of a route R (consisting of a sequence of grids) is defined to be $\sum_{(i,j) \in R} C_{i,j}^t$. Those routes with route energies ranked at the bottom 50% are deleted. And then from the remaining routes, D picks the one with the least hop count by unicasting a RREP to S traveling in the reverse direction of the route.

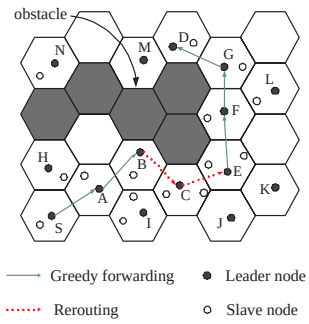


Figure 4: A greedy forwarding example for connectionless routing.

3.2.2 Connectionless (Stateless) Routing

For connectionless routing, no routing information needs to be constructed by leaders. To send a packet, a host simply relays to the neighbor that is closest to the destination. In case of hitting a dead end (local minimum), a *left-hand* or *right-hand* rerouting rule is used to route around the obstacle. This is typically called *greedy forwarding*. A number of such protocols have been proposed [1, 12, 22, 31]. However, they do not rely on partitioning the physical area into grids.

Our protocol works similarly greedy forwarding. Each leader periodically broadcasts its location and collective power with neighboring leaders. Any packet is sent to the local leader first. The greedy forwarding rule is used until either the destination or a local minimum is reached. In greedy forwarding, if multiple choices exist, the one with the best energy value will be selected. Fig. 4 gives an example.

3.2.3 Location Service for Location Tracking

The above routing protocols all rely on pre-knowledge about the current (or approximated) location of the destination. However, since mobile hosts may move at any time, tracking their locates presents an important issue. This is similar to the *location-tracking problem* in PCS networks. However, it is more challenging here since no central database should be used. We call this *location service problem*.

In the literature, a number of distributed location service solutions have been proposed [6, 7, 13, 16]. Fortunately, all these match well with our network architecture. Specifically, they all count on partitioning the network area into square grids, which are called *level-0* squares. Then level-i squares are constructed recursively by each grouping four level-$(i-1)$ grids together. Our approach directly supports such location service solutions.

3.3 Power Mode Management

Our protocol relies on electing a leader in each grid which should always remain awake. The leader's responsibility includes: (i) propagating RREQ, RREP, and data packets,

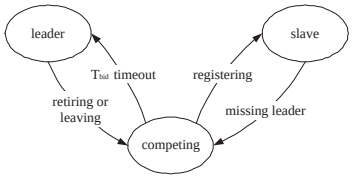


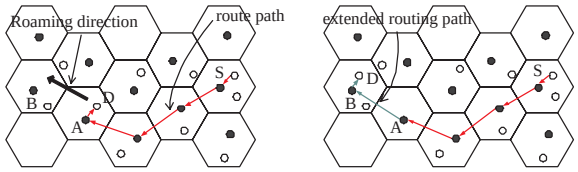
Figure 5: State transition for mobile hosts' power mode.

(ii) maintaining the local grid's routing table (under the connection-oriented case), (iii) keeping track of the other members in the local grid, and (iv) buffering packets for sleeping hosts. Fig. 5 shows the state transition of mobile hosts' power mode. A host can be in one of three states: *leader*, *competing*, and *slave*. Under the first two states, the host should remain active, while under the last state, it may go to sleep if its does not intend to send/receive. Also, to be power-aware, we rank a host's energy c_i^t into n levels, such that level j satisfies $\frac{j-1}{n} < c_i^t \leq \frac{j}{n}$, $j = 1..n$.

Whenever a host i is unaware of any leader in its grid, it should enter the competing state. It should contend to become the leader by broadcasting a $BID(g, loc, c_i^t, T_{bid})$ packet, where g is the grid identity, loc is i's current location, and T_{bid} a time interval. When a host j, which is also in the competing state, receives i's BID, it compares its energy rank against the received one. If j's energy is ranked better, it broadcasts a BID. If this is a tie, j further compares its current location against the i's. If j is closer to the center of the grid, it broadcasts a BID. Otherwise, j simply keeps silent, which means that j consents to i's bid. The loser j can enter the slave state. Host i, after detecting no BID for time interval T_{bid} after sending its BID, assumes itself as the leader and enters the leader state.

To announce its leadership, a leader should periodically broadcast a $HELLO(g, id, T_{hello})$ packet, where g is its grid identity, id its host identity, and T_{hello} the time interval between two consecutive $HELLO$s. On the other hand, a slave should periodically wake up to register with its leader by sending a $REG(g, id, c_i^t)$ packets carrying its own identity. Each leader should maintain a table to keep track of its local slaves. The registration period, denoted by T_{reg}, should be a multiple of T_{hello}. To reduce power consumption, we recommend that T_{reg} be considerably larger than T_{hello}. A slave who does not register for a timeout period $T_{reg\_timeout}$ is regarded as leaving this grid. We recommend that $T_{reg\_timeout}$ be 3$\sim$5 times of T_{reg}.

A slave, on newly entering a grid, should wait for the next $HELLO$ from the corresponding leader, in which case it can reply a REG to register. It is possible that the slave has not heard a BID for long time, in which case it may compete by sending a BID. When this is heard by a leader, it simply replies a $HELLO$ to discourage the slave. By so doing, we intend to keep the leader's leadership even if the bidder has better location/energy rank to avoid continuous changes of leaders. That is, a leader only returns to the competing state when it finds necessary to do so.



(a) before D move to new grid of leader B (b) after D move to new grid of leader B

Figure 6: Route maintenance example when a destination D moves into a new grid.

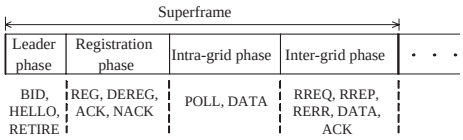


Figure 7: The structure of a superframe.

A leader, when leaving its current grid or becoming unwilling to serve as a leader due to low energy rank, can inform its slaves so by sending a $RETIRE(g, id, RT)$ packet in replace of $HELLO$, where RT the leader's current routing table. In this case, it returns to the competing state. Other slaves, on hearing the $RETIRE$, will enter the competing state for electing a new leader. The transmission of RT is for the next leader to carry on the routing job quickly without breaking current routes. This interesting "handover" process helps one leader shifts its job to another easily.

A leader should also help maintain its current routes. When a leader which is the only host in its grid leaves, all current routes passing this grid become broken. In this case, the leader should send a $RERR$ to neighboring leaders. This serves as a purpose to erase the erroneous route entries. The packet may be propagated farther by the receiving hosts.

On a slave i which is also a source or a destination of a route leaves its current grid, should deregister with its previous leader by sending a $DEREG(g, i, ng)$, where g and ng are its old and new grid identies. The leader of g will add a pointer to ng in its routing table so that afterward when data packets for i arrive, they can be correctly forwarded without need of reestablishing the route. Fig. 6 shows such an scenario.

3.4 MAC Layer

Below, we discuss how hosts contend to access the medium. Our design has the following features. First, we separate intra- from inter-grid communications. The former is supported by the point coordination function (PCF) of IEEE 802.11, while the latter supported by the distributed coordination function (DCF) of IEEE 802.11. Second, a host can switch to a low-power sleep mode without having to worry losing packets. Third, QoS communication can be supported based on our reservation and scheduling strategies.

The channel is partitioned into a sequence of *superframes*. Each superframe is of length T_{hello} and has four phases: leader phase, registration phase, intra-grid phase, and inter-grid phase, as shown in Fig. 7. Superframes among different grids are synchronized (this is made possible by the availability of clocks from GPS [11]). The leader phase is for each leader to send its $HELLO$ packet. Since inter-grid interference may exist, each leader should wait for a random backoff time before sending out its $HELLO$. The contention shouldn't be serious since a grid has only few neighbors.

In the registration phase, slaves may send REG and $DEREG$ packets. In addition, to support QoS communication, a host i in grid g can send a request packet $REQ(g, i, s)$ to its leader, where s is the number of superframes that it will stay in the doze mode. On the other hand, the leader can accept/reject the request by replying an ACK/NAK. A NAK can be replied if the leader would like to keep the slave awake so that it can forward buffered packets to the slave. A special case is $s = 0$, where the slave can notify the leader that it has packets to be sent, in which case the leader should poll this slave in the subsequent intra-grid phase.

In the intra-grid phase, the leader polls its slaves in a round robin manner. The polling list may contain slaves with buffered packets in the leader's side and those who have sent REQ with $s = 0$. The packet exchange is supported by the PCF of IEEE 802.11. Specifically, POLLs are sent to slaves one by one. Data payload can be combined with a POLL. On being polled, a slave can return a data packet targeted at the leader or another slave in the same grid. Note that PIFS is used to separated these frames. By such polling mechanism, QoS communication is achieved.

Finally, in the inter-grid phase, leaders can exchange packets. This is supported by the DCF of IEEE 802.11, so hosts contend to send packets based on CSMA/CA using RTS/CTS dialogues. DIFS is used to separate these frames. Note that intra-grid phases and inter-grid phases of different grids do not have to be perfectly synchronized. Since DIFS is longer than PIFS, intra-grid phases will have higher priority. The whole network can coordinate to pick a preferred length of intra-grid phase. However, supported by different IFSs, the intra-grid phase can be reduced or extended in length as desired by individual grids without problem.

3.5 Physical Layer

Generally speaking, power control is desired for two reasons: saving energy and increasing channel reuse (due to less interference). Since our protocol separates intra-grid from inter-grid communications, hosts can apply different transmission powers for them. Recall d and r. For intra-grid communications, it is sufficient to reduce the transmission range within $\sqrt{2}d$ and $2d$ for the square and hexagon

cases, respectively. For inter-grid communications, the transmission range depends on the inter-leader distance and radio environment. This can be supported by the receive signal strengths of RTS/CTS as suggested in [29]. One exception is the route request RREQ packets, which should be sent with the largest possible power so as to discover routes with less hop counts.

4 Conclusions

We have proposed a new power-aware and location-aware architecture for efficient communications in MANETs. Based on a grid concept, we have shown how to conserve energies in all aspects of routing, power mode management, medium access control, and transmission power control by exploiting hosts' physical positions. Our solution is an integrated one compared to existing solutions, which only address power saving in some of these aspects. Lifetimes of mobile hosts can be extended significantly. Network throughput, surprisingly, is not hurt by such conservation. These results can greatly improve the usefulness of MANETs. Simulation results are not presented due to space limit, but can be found in http://www.csie.nctu.edu.tw/∼yctseng.

References

[1] P. Bose, P. Morin, I. Stojmenovic, and J. Urrutia. Routing with guaranteed delivery in ad hoc wireless networks. *Wireless Networks*, Nov. 2001.

[2] B. Chen, K. Jamieson, H. Balakrishnan, and R. Morris. Span: An Energy-Efficient Coordination Algorithm for Topology Maintenace in Ad Hoc Wireless Networks. *ACM MOBICOM*, pages 85–96, 2001.

[3] C. F. Chiasserini and R. R. Rao. A Distributed Power Management Policy for Wireless Ad Hoc Networks. *IEEE Wireless Communication and Networking Conference*, pages 1209–1213, 2000.

[4] L. M. Feeney and M. Nilsson. Investigating the Energy Consumption of Wireless Network Interface in an Ad Hoc Networking Environment. *IEEE INFOCOM*, pages 1548–1557, 2001.

[5] J. Gomez, A. T. Campbell, M. Naghshineh, and C. Bisdikian. A Distributed Contention Control Mechanism for Power Saving in random-access Ad-Hoc Wireless Local Area Networks. *Proc. of IEEE Int'l Workshop on Mobile Multimedia Communications*, pages 114–123, 1999.

[6] Z. Hass and B. Liang. Ad Hoc Mobility Management with Uniform Quorum System. *IEEE/ACM Trans. Networking*, pages 228–240, Apr. 1999.

[7] P. H. Hsiao. Geographical Region Summary Service for Geographical Routing. *Mobile Computing and Communication Review*, pages 25–39, Oct. 2001.

[8] http://www.bluetooth.com. Bluetooth SIG Bluetooth Specification v1.1. February, 2001.

[9] L. Hu. Topology Control for Multihop Packet Radio Networks. *IEEE Trans. on Communications*, 41:1474–1481, Oct. 1993.

[10] R. Jain, A. Puri, and R. Sengupta. Geographical Routing Using Partial Information for Wireless Ad Hoc Networks. *IEEE Personal Communications*, pages 48–57, Feb. 2001.

[11] E. Kapalan. Understanding GPS. *Artech House*, 1996.

[12] B. Karp and H. T. Kung. GPSR: Greedy Perimeter Stateless Routing for Wireless Networks. *ACM MOBICOM*, pages 243–254, 2000.

[13] J. Li, J. Jannotti, D. Couto, D. R. Karger, and R. Morris. A Scalable Location Service for Geographic Ad hoc Routing. *ACM MOBICOM*, pages 120–130, 2000.

[14] W.-L. Liao, Y.-C. Tseng, and J.-P. Sheu. GRID: A Fully Location-Aware Routing Protocol for Mobile Ad Hoc Networks. *Telecommunication Systems*, 18:37–60, Sep. 2001.

[15] J. R. Lorch and A. J. Smith. Software Strategies for Portable Computer Energy Management. *IEEE Personal Communications*, pages 60–73, Jun. 1998.

[16] M. Mauve and J. Widmer. A Survey on Position-Based Routing in Mobile Ad Hoc Networks. *IEEE Netwroks*, pages 30–39, Nov./Dec. 2001.

[17] L. M. S. C. of the IEEE Computer Society. IEEE Std 802.11-1999, Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications. *IEEE*, 1999.

[18] R. Ramanathan and R. Rosales-Hain. Topology Control of Multihop Wireless Networks using Transmit Power Adjustment. *IEEE INFOCOM*, pages 404–413, 2000.

[19] S. Singh and C. S. Raghavendra. PAMAS: Power aware multi-access protocol with signalling for ad hoc networks. *Computer Communication Review*, pages 5–26, 1998.

[20] S. Singh, M. Woo, and C.-S. Raghavendra. Power-Aware Routing in Mobile Ad Hoc Networks. *ACM MOBICOM*, pages 181–190, 1998.

[21] I. Stojmenovic and X. Lin. Power-aware Localized Routing in Wireless Networks. *IEEE Int'l Parallel and Distributed Processing Symp.*, pages 371–376, 2000.

[22] H. Takagi and L. K. and. Optimal Transmission Ranges for Ranges for Randomly Distributed Packet Radio Terminals. *IEEE Trans. Communication*, pages 246–257, Mar. 1984.

[23] C.-K. Toh. Maximum battery Life Routing to Support Ubiquitous Mobile Computing in Wireless Ad Hoc Networks. *IEEE Communications Magazine*, 39:138–147, Jun. 2001.

[24] Y.-C. Tseng, C.-S. Hsu, and T.-Y. Hsieh. Power-saving Protocols for IEEE 802.11-Based Multi-Hop Ad Hoc Networks. *IEEE INFOCOM*, 2002.

[25] Y.-C. Tseng, S.-L. Wu, W.-H. Liao, and C.-M. Chao. Location Awareness in Ad Hoc Wireless Mobile Networks. *IEEE Computers*, 34(6):46–52, Jun. 2001.

[26] K. Tsudaka, M. Kawahara, A. Matsumoto, and H. Okada. Power Control for Multihop Wireless Ad-hoc Network. *IEEE GLOBECOM*, pages 2819–2824, 2001.

[27] R. Wattenhofer, L. Li, P. Bahl, and T.-M. Wang. Distributed Topology Control for Power Efficient Operation in Multihop Wireless Ad Hoc Networks. *IEEE INFOCOM*, pages 1388–1397, 2001.

[28] C.-F. Wu, Y.-C. Tseng, S.-L. Wu, and J.-P. Sheu. Increasing the throughput of multihop packet radio networks with power adjustment. *IEEE ICCCN*, pages 220–225, 2001.

[29] S.-L. Wu, Y.-C. Tseng, and J.-P. Sheu. Intelligent Medium Access for Mobile Ad Hoc Networks with BusyTones and Power Control. *IEEE J. on Selected Areas in Communications*, 18:1647–1657, Sep. 2000.

[30] Y. Xu, J. Heidemann, and D. Estrin. Georaphy-informed Energy Conservation for Ad Hoc Routing. *ACM MOBICOM*, pages 70–84, 2001.

[31] Y. Xue and B. Li. A Location-aided Power-aware Routing Protocol in Mobile Ad Hoc networks. *IEEE GLOBECOM*, pages 2837–2841, 2001.

論文名稱：Power-Saving Protocols for IEEE 802.11-Based Multi-Hop Ad Hoc Networks

作者：Y.-C. Tseng, C.-S. Hsu, and T.-Y. Hsieh

發表情況: ***Computer Networks***, Elsevier Science Pub., Vol. 43, No. 3, Oct. 2003, pp. 317-337. (SCIE, EI)

Power-Saving Protocols for IEEE 802.11-Based Multi-Hop Ad Hoc Networks

Yu-Chee Tseng[†], Chih-Shun Hsu[‡] and Ten-Yueng Hsieh[†]

[†]Department of Computer Science and Information Engineering

National Chiao Tung University

Hsin-Chu, 30050, Taiwan

[‡]Department of Computer Science and Information Engineering

National Central University

Chung-Li, 32054, Taiwan

Email:yctseng@csie.nctu.edu.tw (corresponding author)

Abstract

Power-saving is a critical issue for almost all kinds of portable devices. In this paper, we consider the design of power-saving protocols for *mobile ad hoc networks (MANETs)* that allow mobile hosts to switch to a low-power *sleep* mode. The MANETs being considered in this paper are characterized by unpredictable mobility, multi-hop communication, and no clock synchronization mechanism. In particular, the last characteristic would complicate the problem since a host has to predict when another host will wake up to receive packets. We propose three power management protocols, namely *dominating-awake-interval*, *periodically-fully-awake-interval*, and *quorum-based* protocols, which are directly applicable to IEEE 802.11-based MANETs. As far as we know, the power management problem for multi-hop MANETs has not been seriously addressed in the literature. Existing standards, such as IEEE 802.11, HIPERLAN, and bluetooth, all assume that the network is fully connected or there is a clock synchronization mechanism. Extensive simulation results are presented to verify the effectiveness of the proposed protocols.

Keywords: HIPERLAN, IEEE 802.11, mobile ad hoc network (MANET), power management, power saving, wireless communication.

1 Introduction

Computing and communication anytime, anywhere is a global trend in today's development. Ubiquitous computing has been made possible by the advance of wireless communication technology and the availability of many light-weight, compact, portable computing devices. Among the various network architectures, the design of *mobile ad hoc network (MANET)* has attracted a lot of attention recently. A MANET is one consisting of a set of mobile hosts which can communicate with one another and roam around at their will. No base stations are supported in such an environment, and mobile hosts may have to communicate with each other in a *multi-hop* fashion. Applications of MANETs occur in situations like battlefields, major disaster areas, and outdoor assemblies. It is also a prospective candidate to solve the "last-mile" problem for broadband Internet service providers [1].

One critical issue for almost all kinds of portable devices supported by battery powers is *power saving*. Without power, any mobile device will become useless. Battery power is a limited resource, and it is expected that battery technology is not likely to progress as fast as computing and communication technologies do. Hence, how to lengthen the lifetime of batteries is an important issue, especially for MANET, which is supported by batteries only.

Solutions addressing the power-saving issue in MANETs can generally be categorized as follows:

- *Transmission Power Control:* In wireless communication, transmission power has strong impact on bit error rate, transmission rate, and inter-radio interference. These are typically contradicting factors. In [2], power control is adopted to reduce interference and improve throughput on the MAC layer. How to determine transmission power of each mobile host so as to determine the best network topology, or known as *topology control*, is addressed in [3], [4], [5]. How to increase network throughput by power adjustment for packet radio networks is addressed in [6].

- *Power-Aware Routing:* Power-aware routing protocols have been proposed based on various power cost functions [7], [8], [9], [10], [11]. In [7], when a mobile host's battery level is below a certain threshold, it will not forward packets for other hosts. In [10], five different metrics based on battery power consumption are proposed. Reference [11] considers both hosts' lifetime and a distance power metric. A hybrid environment consisting of battery-powered and outlet-plugged hosts is considered in [8]. Two distributed heuristic clustering approaches for

multicasting are proposed in [9] to minimizing the transmission power.

- *Low-Power Mode:* More and more wireless devices can support low-power sleep modes. IEEE 802.11 [12] has a power-saving mode in which a radio only needs to be awake periodically. HyperLAN allows a mobile host in power-saving mode to define its own active period. An active host may save powers by turning off its equalizer according to the transmission bit rate. Comparisons are presented in [13] to study the power-saving mechanisms of IEEE 802.11 and HIPERLAN in ad hoc networks. Bluetooth [14] provides three different low-power modes: *sniff*, *hold*, and *park*. Other references include [15], [16], [17], [18], [19], [20], [21].

This paper studies the management of power-saving (PS) modes for IEEE 802.11-based MANETs and thus falls into the last category of the above classification. We consider MANETs which are characterized by multi-hop communication, unpredictable mobility, no plug-in power, and no clock synchronization mechanism. In particular, the last characteristic would complicate the problem since a host has to predict when another host will wake up to receive packets. Thus, the protocol must be asynchronous. As far as we know, the power-management problem for multi-hop MANETs has not been addressed seriously in the literature. Existing standards, such as IEEE 802.11 and HYPERLAN, do support PS modes, but assume that the MANET is fully connected. Bluetooth also has low-power modes, but is based on a master-slave architecture, so time synchronization is trivial. The works [18], [19] address the power-saving problem, but assume the existence of access points. A lot of works have focused on multi-hop MANETs on issues such as power-aware routing, topology control, and transmission power control (as classified above), but how to design PS mode is left as an open problem.

Two major challenges that one would encounter when designing power-saving protocols are: *clock synchronization* and the *neighbor discovery*. Clock synchronization in a multi-hop MANET is difficult since there is no central control and packet delays may vary due to unpredictable mobility and radio interference. PS modes are typically supported by letting low-power hosts wake up only in specific time. Without precise clocks, a host may not be able to know when other PS hosts will wake up to receive packets. Further, a host may not be aware of a PS host at its neighborhood since a PS host will reduce its transmitting and receiving activities. Such incorrect neighbor information may be detrimental to most current routing protocols because the route discovery procedure may incorrectly report that there is no route even when routes actually exist with some PS hosts in the middle. These

problems will be discussed in more details in Section 2.

In this paper, we propose three asynchronous power management protocols for multi-hop MANETs, namely *dominating-awake-interval*, *periodically-fully-awake-interval*, and *quorum-based* protocols. We target ourselves at IEEE 802.11-based LAN cards. The basic idea is twofold. First, we enforce PS hosts sending more beacon packets than the original IEEE 802.11 standard does. Second and most importantly, we carefully arrange the wake-up and sleep patterns of PS hosts such that any two neighboring hosts are guaranteed to detect each other in finite time even under PS mode.

Based on our power-saving protocols, we then show how to perform unicast and broadcast in an environment with PS hosts. Simulation results are presented, which show that our protocols can save lots of powers when the traffic load is not high.

The rest of this paper is organized as follows. Preliminaries are given in Section 2. In Section 3, we present our power-saving protocols. Unicast, broadcast and routing protocols based on our power-saving mechanisms are in Section 4. Simulation results are presented in Section 5. Section 6 concludes this paper.

2 Preliminaries

In this section, we start with a general review on power-saving works, followed by detailed design of PS mode in IEEE 802.11. Then we motivate our work by pointing out some problems connecting to PS mode in multi-hop MANETs.

2.1 Reviews of Power Mode Management Protocols

Several power management protocols have been proposed for MANET in [15], [16], [20], [21]. The *PAMAS* (Power Aware Multi-Access protocol with Signalling) [15] protocol allows a host to power its radio off when it has no packet to transmit/receive or any of its neighbors is receiving packets, but a separate signalling channel to query neighboring hosts' states is needed. Reference [16] provides several sleep patterns and allows mobile hosts to select their sleep patterns based on their battery status and quality of service, but a special hardware, called *Remote Activated Switch (RAS)*, is required which can receive wakeup signals even when the mobile host has entered a sleep state. A connected-dominated-set-based power-saving protocol is proposed in [20]. Some hosts must serve

as *coordinators*, which are chosen according to their remaining battery energies and the numbers of neighbors they can connect to. In the network, only coordinators need to keep awake; other hosts can enter the sleeping mode. Coordinators are responsible of relaying packets for neighboring hosts. With a similar idea, a grid-based energy-saving routing protocol is proposed in [21]. With the help of GPS, the area is partitioned in to small sub-areas called grids, in each of which only one host needs to remain active to relay packets for other hosts in the same grid.

A page-and-answer protocol is proposed in [18] for wireless networks with base stations. A base station will keep on sending paging messages whenever there are buffered packets. Each mobile host powers up periodically. However, there is no time synchronization between the base station and mobile hosts. On reception of paging messages, mobile hosts return acknowledgements, which will trigger the base station to stop paging and begin transmitting buffered packets. After receiving the buffered packets, mobile hosts return to power-saving mode, and the process repeats. When the system is too heavily loaded, the base station may spend most of its time in transmitting buffered packets, instead of paging messages. This may result in long packet delays for power-saving hosts. A theoretical analysis of [18] is in [22]. Several software power-control issues for portable computers are discussed in [17]. How to combine power management and power control for wireless cards is addressed in [19].

2.2 Power-Saving Modes in IEEE 802.11

IEEE 802.11 [12] supports two power modes: *active* and *power-saving (PS)* modes. The protocols for infrastructure networks and ad hoc networks are different. Under an infrastructure network, there is an access point (AP) to monitor the mode of each mobile host. A host in the active mode is fully powered and thus may transmit and receive at any time. On the contrary, a host in the PS mode only wakes up periodically to check for possible incoming packets from the AP. A host always notifies its AP when changing modes. Periodically, the AP transmits *beacon frames* spaced by a fixed *beacon interval*. A PS host should monitor these frames. In each beacon frame, a *traffic indication map (TIM)* will be delivered, which contains ID's of those PS hosts with buffered unicast packets in the AP. A PS host, on hearing its ID, should stay awake for the remaining beacon interval. Under the contention period (i.e., DCF), an awake PS host can issue a PS-POLL to the AP to retrieve the buffered packets. While under the contention-free period (i.e., PCF), a PS host will wait for the AP to poll it. Spaced by

a fixed number of beacon intervals, the AP will send *delivery TIMs (DTIMs)* within beacon frames to indicate that there are buffered broadcast packets. Immediately after DTIMs, the buffered broadcast packets will be sent.

Under an ad hoc network, PS hosts also wake up periodically. ATIM frames, instead of TIM frames, are sent. The ATIM frame is a subtype of the management frame, which contains frame control, duration, destination address, source address, BSSID, sequence control, and FCS fields. However, the frame body field of the ATIM frame is null. The short interval that PS hosts wake up to transmit/receive ATIM frames to/from other hosts is called the *ATIM window*. It is assumed that hosts are fully connected and all synchronized, so the ATIM windows of all PS hosts will start at about the same time. In the beginning of each ATIM window, each host will contend to send a beacon frame. Any successful beacon serves as the purpose of synchronizing hosts' clocks. This beacon also inhibits other hosts from sending their beacons. To avoid collisions among beacons, a host should wait a random number of slots between 0 and $2 \times CW_{min} - 1$ before sending out its beacon.

After the beacon, a host with buffered unicast packets can send a *directed ATIM* to each of its intended receivers in PS mode during the ATIM window to inform the receivers that there are buffered packets for them. A directed ATIM should be acknowledged immediately. If no acknowledgement is received, the ATIM shall be retransmitted using the DCF access procedure. A station transmitting an ATIM shall remain awake for the entire current beacon interval. If a station has buffered mutlicast frames, it shall transmit an appropriate addressed *multicast ATIM*. Multicast ATIM frames shall not be acknowledged. Immediately following the ATIM window, a station shall begin transmitting buffered multicast frames. Following the transmission of multicast frames, unicast packets can then be sent based on the DCF access procedure. If a mobile host is unable to transmit its ATIM frame in the current ATIM window or has extra buffered packets, it should retransmit ATIMs in the next ATIM window. To protect PS hosts, only RTS, CTS, ACK, Beacon, and ATIM frames can be transmitted during the ATIM window.

Figure 1 shows an example, where host A wants to transmit a packet to host B. During the ATIM window, an ATIM frame is sent from A to B. In response, B will reply with an ACK. After the ATIM window finishes, A can try to send out its data packet.

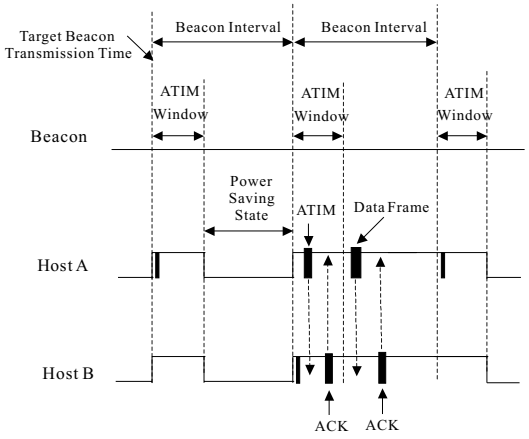


Figure 1: An example of unicasting in an ad hoc networks with PS hosts.

2.3 Problem Statement

The PS mode of IEEE 802.11 is designed for a single-hop (or fully connected) ad hoc network. When applied to a multi-hop ad hoc network, three problems may arise. All these will pose a demand of redesigning the PS mode for multihop MANET.

A) Clock Synchronization: Since IEEE 802.11 assumes that mobile hosts are fully connected, the transmission of a beacon frame can be used to synchronize all hosts' beacon intervals. So the ATIM windows of all hosts can appear at around the same time without much difficulty. In a multi-hop MANET, clock synchronization is a difficult job because communication delays and mobility are all unpredictable, especially when the network scale is large. Even if perfect clock synchronization is available, two temporarily partitioned sub-networks may independently enter PS mode and thus have different ATIM timing. With the clock-drifting problem, the ATIM windows of different hosts are not guaranteed to be synchronous. Thus, the ATIM window has to be re-designed.

B) Neighbor Discovery: In a wireless and mobile environment, a host can only be aware by other hosts if it transmits a signal that is heard by the others. For a host in the PS mode, both its chances to transmit and to hear others' signals are reduced. As reviewed above, a PS host must compete with other hosts to transmit its beacon. A host will cancel its beacon frame once it hears other's beacon frame. This may run into a dilemma that hosts are likely to have inaccurate neighborhood information when there are PS hosts. Thus, many existing routing protocols that depend on neighbor information may be impeded.

C) Network Partitioning: The above inaccurate neighbor information may lead to long packet delays or even network-partitioning problem. PS hosts with unsynchronized ATIM windows may wake up at different times and may be partitioned into several groups. These conceptually partitioned groups are actually connected. Thus, many existing routing protocols may fail to work in their route discovery process unless all hosts are awaken at the time of the searching process.

3 Power-Saving Protocols for MANET

In this section, we present three asynchronous power-saving protocols that allow mobile hosts to enter PS mode in a multi-hop MANET. According to the above discussion, we derive several guidelines in our design:

- *More Beacons:* To prevent the inaccurate-neighbor problem, a mobile host in PS mode should insist more on sending beacons. Specifically, a PS host should not inhibit its beacon in the ATIM window even if it has heard others' beacons. This will allow others to be aware of its existence. For this reason, our protocols will allow multiple beacons in an ATIM window.

- *Overlapping Awake Intervals:* Our protocols do not count on clock synchronization. To resolve this problem, the wake-up patterns of two PS hosts must overlap with each other no matter how much time their clocks drift away.

- *Wake-up Prediction:* When a host hears another PS host's beacon, it should be able to derive that PS host's wake-up pattern based on their time difference. This will allow the former to send buffered packets to the later in the future. Note that such prediction is not equal to clock synchronization since the former does not try to adjust its clock.

Based on the above guidelines, we propose three power-saving protocols, each with a different wake-up pattern for PS hosts. PS hosts' wake-up patterns do not need to be synchronous. For each PS host, it divides its time axis into a number of fixed-length intervals called *beacon intervals*. In each beacon interval, there are three windows called *active window*, *beacon window*, and *MTIM window*. During the active window, the PS host should turn on its receiver to listen to any packet and take proper actions as usual. The beacon window is for the PS host to send its beacon, while the MTIM window is for other hosts to send their MTIM frames to the PS host. Our MTIM frames serve the

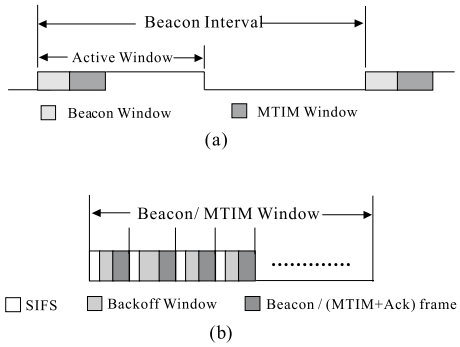


Figure 2: Structure of a beacon interval: (a) active, beacon, and MTIM windows and (b) access procedure.

similar purpose as ATIM frames in IEEE 802.11; here we use MTIM to emphasize that the network is a mutil-hop MANET. Excluding these three windows, a PS host with no packet to send or receive may go to the sleep mode. Figure 2(a) shows an example structure of a beacon interval.

The following notations are used throughout this paper:

- BI: length of a beacon interval.
- AW: length of an active window.
- BW: length of a beacon window.
- MW: length of an MTIM window, where $MW > BW$.

We should comment at this point that the structure of a beacon interval may vary for different protocols (to be elaborated later). The illustration in Figure 2(a) is only one of the several possibilities. In the beacon window (resp., MTIM window), hosts can send beacons (resp., MTIM frames) following the DCF access procedure. Each transmission must be led by a SIFS followed by a random delay ranging between 0 and $2 \times CW_{min} - 1$ slots. This is illustrated in Figure 2(b).

3.1 Protocol 1: Dominating-Awake-Interval

The basic idea of this approach is to impose a PS host to stay awake sufficiently long so as to ensure that neighboring hosts can know each other and, if desire, deliver buffered packets. By "dominating-awake", we mean that a PS host should stay awake for at least about half of BI in each beacon

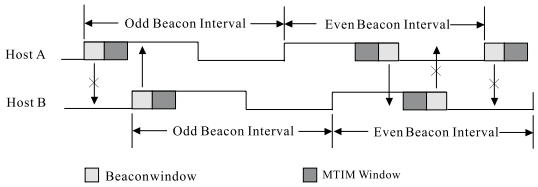


Figure 3: Structures of odd and even intervals in the Dominating-Awake-Interval protocol.

interval. This guarantees any PS host's beacon window to overlap with any neighboring PS host's active window, and vice versa.

This protocol is formally derived as follows. When a host decides to enter the PS mode, it divides its time axis into fixed-length beacon intervals, each of length BI. Within each beacon, the lengths of all three windows (i.e., AW, BW, and MW) are constants. To satisfy the "dominating-awake" property, we enforce that $AW \geq BI/2 + BW$. The sequence of beacon intervals is alternatively labeled as *odd* and *even* intervals. Odd and even intervals have different structures as defined below (see the illustration in Fig. 3).

- Each odd beacon interval starts with an active window. The active window is led by a beacon window and followed by an MTIM window.
- Each even beacon interval also starts with an active window, but the active window is terminated by an MTIM window followed by a beacon window.

It is not hard to see that by imposing the active window occupying at least half of each beacon interval, we can guarantee that two hosts' active windows always have some overlapping. However, why we have different structures for odd and even beacon intervals remains obscure. Let's consider Fig. 4, where beacon windows always appear at the beginning of beacon intervals. In this case, host A can hear host B's beacons, but B always misses A's beacons. On the contrary, as Figure 3 shows, A can hear B's beacons at odd intervals, and B can hear A's beacons at even intervals.

Earlier we imposed the condition $AW \geq BI/2+BW$. The following theorem provides a formal proof on the correctness of this protocol (proof available in Appendix A).

Theorem 1 *The Dominating-Awake-Interval protocol guarantees that when $AW \geq BI/2 + BW$, a PS host's entire beacon window always overlaps with any neighboring PS host's active window in every other beacon interval, no matter how much time their clocks drift away.*

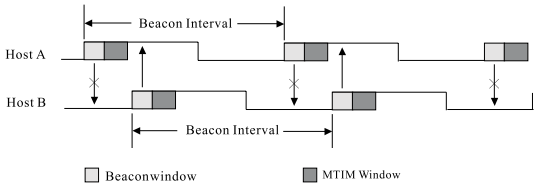


Figure 4: An example where host B will always miss A's beacons.

The above theory guarantees that a PS host is able to receive all its neighbors' beacon frames in every two beacon intervals, if there is no collision in receiving the latter's beacons. Since the response time for neighbor discovery is pretty short, this protocol is suitable for highly mobile environments.

3.2 Protocol 2: Periodically-Fully-Awake-Interval

The previous protocol requires PS hosts keep active more than half of the time, and thus is not energy-efficient. To reduce the active time, in this protocol we design two types of beacon intervals: *low-power intervals* and *fully-awake intervals*. In a low-power interval, the length of the active window is reduced to the minimum, while in a fully-awake interval, the length of the active window is extended to the maximum. Since fully-awake intervals need a lot of powers, they only appear periodically and are interleaved by low-power intervals. So the energy required can be reduced significantly.

Formally, when a host decides to enter the PS mode, it divides its time axis into fixed-length beacon intervals of length BI. The beacon intervals are classified as *low-power* and *fully-awake* intervals. The fully-awake intervals arrive periodically every p intervals, and the rest of the intervals are low-power intervals. The structures of these beacon intervals are defined as follows.

- Each low-power interval starts with an active window, which contains a beacon window followed by an MTIM window, such that $AW = BW + MW$. In the rest of the time, the host can go to the sleep mode.

- Each fully-awake interval also starts with a beacon window followed by an MTIM window. However, the host must remain awake in the rest of the time, i.e., $AW = BI$.

Intuitively, the low-power intervals are for a PS host to send out its beacons to inform others about its existence. The fully-awake intervals are for a PS host to discover who are in its neighborhood. It is not hard to see that a fully-awake interval always has overlapping with any host's beacon windows,

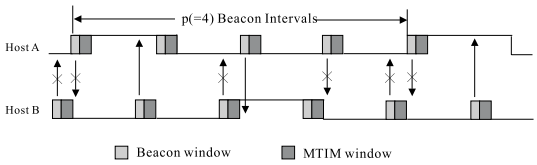


Figure 5: An example of the Periodically-Fully-Awake-Interval protocol with fully-awake intervals arrive every $p = 4$ beacon intervals.

no matter how much time their clocks drift away. By collecting other hosts' beacons, the host can predict when its neighboring hosts will wake up. Figure 5 shows an example with $p = 4$ intervals. So hosts A's and B's beacons always have chances to reach the other's active windows. Proof of the following theorem is in Appendix B.

Theorem 2 *The Periodically-Fully-Awake-Interval protocol guarantees that a PS host's beacon windows overlap with any neighbor's fully-awake intervals in every p beacon intervals, no matter how much time their clocks drift away.*

Compared to the previous Dominating-Awake-Interval protocol, which requires a PS host to stay awake more than half of the time, this protocol can save more power as long as $p > 2$. However, the response time to get aware of a newly appearing host could be as long as p beacon intervals. So this protocol is more appropriate for slowly mobile environments. One way to reduce the response time is to decrease the value of p to fit one's need.

3.3 Protocol 3: Quorum-Based

In the previous two protocols, a PS host has to contend to send a beacon in each beacon interval. In this section, we propose a protocol based on the concept of *quorum*, where a PS host only needs to send beacons in $O(1/n)$ of the all beacon intervals. Thus, when transmission takes more powers than reception, this protocol may be more energy-efficient. The concept of quorums has been used widely in distributed system design (e.g., to guarantee mutual exclusion [23], [24], [25], [26]). A quorum is a set of identities from which one has to obtain permission to perform some action [23]. Typically, two quorum sets always have nonempty intersection so as to guarantee the atomicity of a transaction. Here we adopt the concept of quorum to design PS hosts' wakeup patterns so as to guarantee a PS host's beacons can always be heard by others' active windows. This is why our protocol is named so.

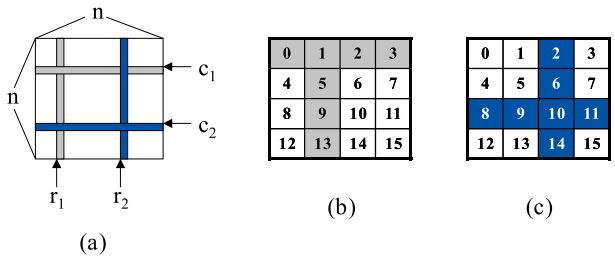


Figure 6: Examples of the Quorum-Based Protocol (a)intersections of two PS hosts' quorum intervals, (b)host A's quorum intervals, and (c)host B's quorum intervals

The quorum structure of our protocol is as follows. The sequence of beacon intervals is divided into sets starting from the first interval such that each continuous n^2 beacon intervals are called a *group*, where n is a global parameter. In each group, the n^2 intervals are arranged as a 2-dimensional $n \times n$ array in a row-major manner. On the $n \times n$ array, a host can arbitrarily pick one column and one row of entries and these $2n - 1$ intervals are called *quorum intervals*. The remaining $n^2 - 2n + 1$ intervals are called *non-quorum intervals*.

Before proceeding, let's make some observation from the quorum structure. Given two PS hosts that are perfectly time-synchronized, it is not hard to see that their quorum intervals always have at least two intersecting beacon intervals(see the illustration in Figure 6(a)). This is due to the fact that a column and a row in a matrix always have an intersection. Thus, two PS host may hear each other on the intersecting intervals. However, the above reasoning is not completely true since we do not assume that hosts are time-synchronized. For example, in Figure 6(b) and (c), host A selects intervals on row 0 and column 1 as its quorum intervals from a 4×4 matrix, while host B selects intervals on row 2 and column 2 as its quorum intervals. When perfectly synchronized, intervals 2 and 9 are the intersections.

The structures of quorum and non-quorum intervals are formally defined below.

- Each quorum interval starts with a beacon window followed by an MTIM window. After that, the host must remain awake for the rest of the interval, i.e., $AW = BI$.
- Each non-quorum interval starts with an MTIM window. After that, the host may go to the sleep mode, i.e., we let $AW = MW$.

The protocol guarantees the following property (proof in Appendix C).

Table 1: Characteristics of the proposed power-saving protocols

Protocol	Number of beacons	Active ratio	Neighbor sensitivity
Dominated-Awake	1	$1/2 + BW/BI$	BI
Periodically-Fully-Awake	1	$1/p$	$p \times BI/2$
Quorum-Based	$(2n-1)/n^2$	$(2n-1)/n^2$	$(n^2/4) \times BI$

Theorem 3 *The Quorum-Based protocol guarantees that a PS host always has at least two entire beacon windows that are fully covered by another PS host's active windows in every n^2 beacon intervals.*

The Quorum-Based protocol has advantage in that it only transmits in $O(1/n)$ of the beacon intervals(on the contrary, the earlier two protocols have to transmit a beacon in every interval). In addition, it also keeps awake in $O(1/n)$ of the time. As long as $n \geq 4$, this amount of awaking time is less than 50%. So this protocol is more energy-efficient when transmission cost is high. The backside is that a PS host may learn its vicinity at lower speed and that it might be more expensive to implement.

3.4 Summary

Table 1 summarizes the characteristics of the three proposed power-saving protocols. "Number of beacon" indicates the average number of beacons that a host need to transmit in each beacon interval,"Active ratio" indicates the ratio of time that a PS host needs to stay awake while in the PS mode, and "Neighbor sensitivity" indicates the average time that a PS host takes to hear a newly approaching neighbor's beacon. As Table 1 shows, the Quorum-Based protocol spends least power in transmitting beacons. The Periodically-Fully-Awake-Interval and the Quorum-based protocols' active ratios can be quite small as long as p and n, respectively are large enough. The Dominated-Awake-Interval protocol is most sensitive to neighbor changes, while the Quorum-based protocol is least sensitive.

4 Unicast and Broadcast Protocols for Power-Saving Hosts

This section discusses how a host sends packets to a neighboring PS host. Since the PS host is not always active, the sending host has to predict when the PS host will wake up, i.e., when the latter's

Table 2: Timing of $MTIM$ windows of the proposed protocols.

Protocol	$MTIM$ window's timing
Dominated-Awake	$[(2m+1) \times BI + BW, (2m+1) \times BI + BW + MW]$ (odd int.) $[2m \times BI + BI/2 - MW, 2m \times BI + BI/2]$ (even int.)
Periodically-Fully-Awake	$[m \times BI + BW, m \times BI + BW + MW]$
Quorum-Based	$[m \times BI + BW, m \times BI + BW + MW]$ (quorum int.) $[m \times BI, m \times BI + MW]$ (non-quorum int.)

$MTIM$ windows will arrive. To achieve this, each beacon packet has to carry the clock value of the sending host so that other hosts can calculate their time differences. Table 2 summarizes when $MTIM$ windows arrive in the proposed protocols, where m is any nonnegative integer.

Since all the hosts in the MANET adopt the same power-saving protocol, the patterns of their MTIM windows are similar, except that their clocks might be different. Consider any two asynchronous mobile hosts A and B. Without loss of generality, let A's clock be faster than B's clock by ΔT μs. Whenever host A's MTIM window arrives, host A can predict that host B's MTIM window will arrive ΔT μs later; host B can predict that host A's MTIM window arrived ΔT μs before its own MTIM window arrives. If the Quorum-Based protocol is adopted, a host first predicts when another host's quorum interval starts and then can also predict the latter's MTIM windows in a similar manner.

After correctly predicting the receiving side's MTIM windows, the sending side can contend to send MTIM packets to notify the receiver during the receiver's MTIM window, after which the buffered data packet can be sent. Below, we discuss how unicast and broadcast are achieved.

4.1 Unicast

This is similar to the procedure in IEEE 802.11's PS mode. During the receiver's $MTIM$ window, the sender contends to send its $MTIM$ packet to the receiver. The receiver, on receiving the MTIM packet, will reply an ACK after an SIFS and stay awake in the remaining of the beacon interval. After the MTIM window, the sender will contend to send the buffered packet to the receiver based on the DCF procedure.

4.2 Broadcast

The situation is more complicated for broadcasting since the sender may have to deal with multiple asynchronous neighbors. To reduce the number of transmissions, we need to divide these asynchronous neighbors into groups and notify them separately in multiple runs. The steps are described below. Note that here the broadcast is not designed to be 100% reliable at the MAC layer (reliable broadcast may be supported at a higher layer).

When a source host S intends to broadcast a packet, it first checks the arrival time of the MTIM windows of all its neighbors. Then S picks the host, say Y, whose first MTIM window arrives earliest. Based on Y's first MTIM window, S further picks those neighbors whose MTIM windows have overlapping with Y's first MTIM window. These hosts, including Y, are groups together and S will try to notify them in one MTIM frame (note that such MTIM frames need not be acknowledged due to the unreliable assumption). After notifying these neighbors, the source S can contend to send its buffered broadcast packet to this group. Broadcast packets should be sent based on the DCF procedure too. After this transmission, S considers the rest of the neighbors that have not been notified yet in the previous MTIM and repeats the same procedure again to initiate another MTIM frame and broadcast packet. The process is repeated until all its neighbors have been notified.

A neighbor, on receiving an MTIM carrying a broadcast indication, should remain awake until a broadcast packet is received or a timeout value expires (here we recommend a timeout value equal to one beacon interval be used, but this can also be an adjustable parameter during system configuration).

We comment that most traditional on-demand routing protocols, such as DSR [27] and AODV [28], rely on broadcasting route request packets to discover new routes. If hosts' clocks are not well synchronized, some PS hosts may miss the request packets. With our broadcasting protocol, the route request packets can be received, though with some delays. At the destination side, unicast can be used to send the route reply packets. When PS hosts in the chosen route receive the route reply packet, it can go to the active mode.

5 Simulation Experiments

To evaluate the performance of the proposed power-saving protocols, we have developed a simulator using C. In the simulations, we assume that the area size is $1000m \times 1000m$, and the transmission

radius is 250 meters (under such an environment, the average distance between hosts is around 3.2 hops). Hosts' transmission rate is 2M bits/sec, and the battery power of each mobile host is 100 J. The MAC part basically follows the IEEE 802.11 standard [12], except the power management part. We adopt AODV as our routing protocol. The source and the destination of each route are randomly selected. Four parameters are tunable in our simulations:

- Traffic load: Routes are generated by a Poisson distribution with rate between $1 \sim 4$ routs/sec. For each route, 10 packets, each of size 1k bytes, will be sent.
- Mobility: Host mobility follows the random way-point model. The pause time is set to 20 seconds. When moving, a host will move at a speed between $0 \sim 20$ m/sec.
- Beacon interval: The length of one beacon interval is $100 \sim 400$ ms.
- Number of hosts: The total number of mobile hosts in the MANET is $50 \sim 200$ hosts.

Basically, each simulation lasts for 100 seconds. However, when measuring the survival ratio of hosts in the MANET, the simulation will last until all the hosts have run out of energies. Each result is obtained from the average of 100 simulation runs. The confidence level shown in the figures is at 95% with the confidence interval of $(\bar{X} - 1.96\sigma/10, \bar{X} - 1.96\sigma/10)$, where $\bar{X}$ is the mean and σ is the standard deviation of the samples. For simplicity, we assume that all hosts are in the PS mode. To make comparison, we also simulate an "always-active" scheme in which all hosts are active all the time.

Four performance metrics are used in the simulations:

- neighbor discovery time: average time to discover a newly approaching neighbor.
- survival ratio: the number of surviving hosts over the total number of hosts. (A host is said to be surviving if its power is not exhausted yet.)
- route establishment probability: the total number of successfully established routes over the total number of requests.
- route request/reply delay: the time from the source host initiating the route request packet to the time the destination host receiving the packet and the time the reply is sent from the destination to the source.

Table 3: Power consumption parameters used in the simulation.

Unicast send	$454 + 1.9 \times L\ \mu J/packet$
Broadcast send	$266 + 1.9 \times L\ \mu J/packet$
Unicast receive	$356 + 0.5 \times L\ \mu J/packet$
Broadcast receive	$56 + 0.5 \times L\ \mu J/packet$
Idle	843 $\mu J/ms$
Doze	27 $\mu J/ms$

Table 4: Traffic-related parameters used in the simulation.

Unicast packet size	1024 bytes
Broadcast packet size	32 bytes
Beacon window size	4 ms
MTIM window size	16 ms

The data packet transmission delay is not observed in our simulation, because when a route is established, all the hosts in the route should be switched to the active mode and thus the data packet transmission delay of our schemes should be similar to that of the "always-active" scheme. The power model in [29] is adopted, which is obtained by real experiments on Lucent WaveLAN cards. Table 3 summarizes the power consumption parameters used in our simulations. Sending/receiving a unicast/broadcast packet has a cost $P_{base} + P_{byte} \times L$, where P_{base} is the power consumption independent of packet length, P_{byte} is the power consumption per byte, and L is the packet length. When sending a packet of the same size, unicast consumes more power than broadcast because it needs to send and receive extra control frames (RTS, CTS, and ACK). The last two entries indicate the consumption when a host has no send/receive activity and is in the active mode and PS mode, respectively. As can be seen, staying in the active mode is much more energy-consuming. The traffic-related parameters are summarized in Table 4.

In the following subsections, we show how beacon interval, mobility, traffic load, and host density affect the performance of the proposed power-saving protocols. For simplicity, the Dominating-Awake-Interval protocol is denoted as D, the Periodically-Fully-Awake-Interval protocol with parameter p is denoted as $P(p)$, the Quorum-Based protocol with parameter n is denoted as $Q(n)$, and the

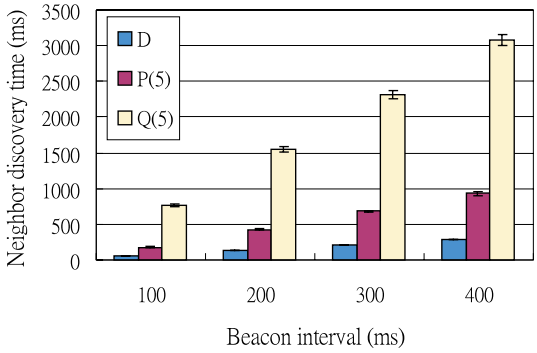


Figure 7: Neighbor discovery time vs. beacon interval length. (100 hosts, traffic load = 1 route/sec, mobility = 5 m/sec)

"always active" scheme is denoted as AA.

5.1 Impact of Beacon Interval Length

The length of beacon intervals has impact on hosts' sensitivity to environmental changes, power consumptions, route establishment probabilities, and route request/reply delays. However, these are contradicting factors. To observe its impact, we vary the beacon interval length between 100 $\sim$ 400 ms. As Figure 7 shows, longer beacon intervals only slightly increase the neighbor discovery time for schemes D and $P(5)$, but have more significant impact on schemes $Q(5)$. Overall, scheme D has the shortest neighbor discovery time, which is subsequently followed by $P(5)$ and $Q(5)$.

Figure 8 shows that longer beacon intervals may lengthen the lifetime of the MANET, because the ratio of awake time for each host becomes smaller. However, longer beacon intervals may increase the broadcasting (and thus route discovery) cost. This will be shown in the next requirement. Overall, scheme $P(5)$ has the longest network lifetime, which is subsequently followed by $Q(5)$ and D.

Figure 9 shows the impact of beacon interval length over route request/reply delays. With longer beacon interval, it takes longer time to wake up hosts. For example, in the route discovery procedure, a host may have to send broadcasts to multiple groups of neighbors. The number of groups usually increases as the interval increases. As Figure 9 shows, $Q(5)$ has the least delays in route request and reply because it transmits less beacons and thus causes less contentions and collisions. Scheme D also has very low delay. Scheme $P(5)$ has the highest delays in both route request and reply.

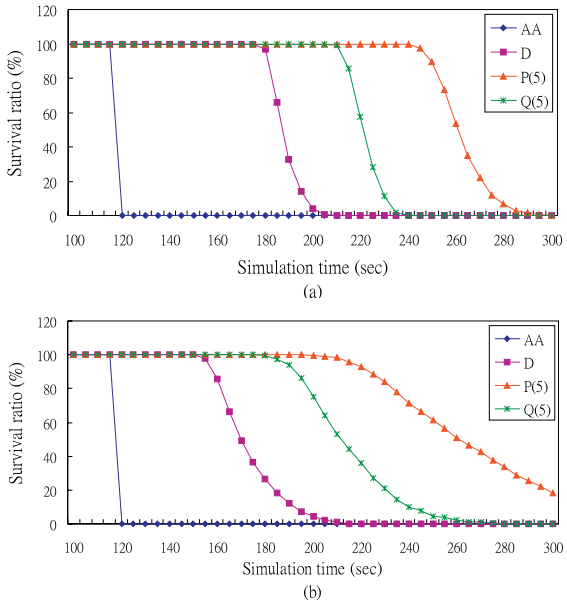


Figure 8: Host survival ratio vs. beacon interval length. (a) beacon interval = 100 ms, and (b) beacon interval = 400 ms. (100 hosts, traffic load = 1 route/sec, moving speed = 5 m/sec)

Figure 10 shows the impact of beacon interval length on route establishment probability. Longer beacon intervals do decrease the probability. This is because longer beacon intervals will increase the time to deliver the route request packets to the destination, and unfortunately, at the time when the route reply packets are issued, the desired route may have become broken. In terms of route establishment probability, the differences between the three schemes are insignificant.

5.2 Impact of Mobility

Mobility has a negative impact on survival ratio, route establishment probability, and route request/reply delays. To observe its effect, we vary hosts' moving speed between $0 \sim 20$ m/sec.

Figure 11 shows the impact of mobility on survival ratio. Mobility will incur high energy consumption because hosts may spend more power on retransmitting packets. Among the four schemes, mobility has the least impact on AA because it takes the least extra power to retransmit unicast packets.

Figure 12 shows that as the moving speed increases the route request/reply delays also increase. However, the impact is insignificant. In terms of transmission delay, scheme AA performs the best,

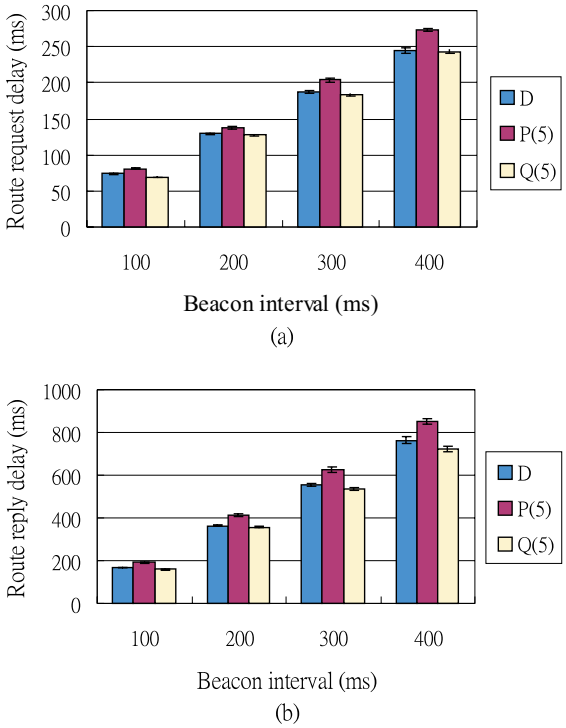


Figure 9: Route request/reply delay vs. beacon interval length. (a) route request delay, and (b) route reply delay. (100 hosts, traffic load = 1 route/sec, moving speed = 5 m/sec)

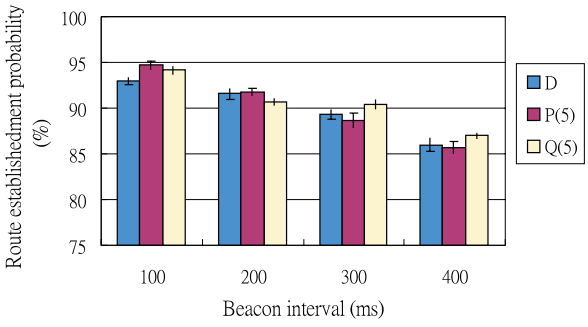


Figure 10: Route establishment probability vs. beacon interval length. (100 hosts, traffic load = 1 route/sec, moving speed = 5 m/sec)

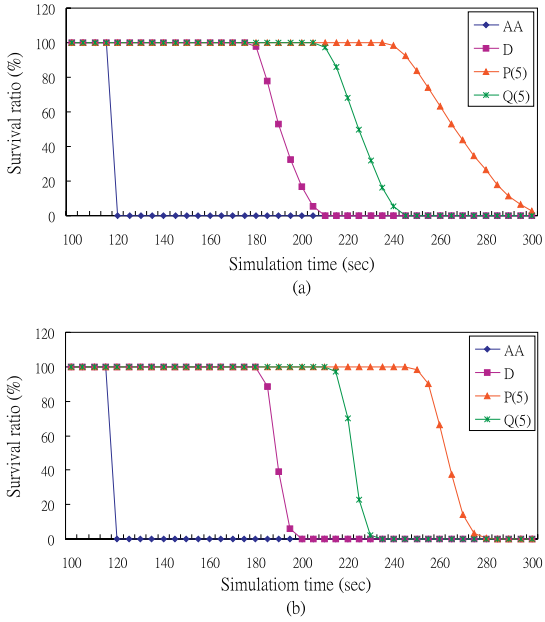


Figure 11: Survival ratio vs. mobility. (a)moving speed = 0 m/sec, and (b)moving speed = 20 m/sec. (beacon interval = 100 ms, 100 hosts, traffic load = 1 route/sec)

which is subsequently followed by $Q(5)$, D and then $P(5)$. Scheme AA performs the best because the sender needs not to wait for the receiver to become active.

Figure 13 shows the impact of mobility on the route establishment probability. Among the four schemes, the AA scheme performs the best and the D scheme performs the worst. $P(5)$ and $Q(5)$ are close to each other. Recall that the D scheme has the most accurate neighbor list. This simulation result indicates that the accuracy of neighbor list is not so important for the route establishment probability because a host missing a route request may hear the same request from other neighbors, and route request is typically done by flooding.

5.3 Impact of Traffic Load

Traffic load also has a negative effect to survival ratio, route establishment probability, and route request/reply delays. To observe its impact, we vary the traffic load between $1 \sim 4$ routes/sec.

As Figure 14 shows, when the traffic load becomes higher, the route establishment probability becomes lower because heavier traffic will cause more collisions and congestions. Among the four

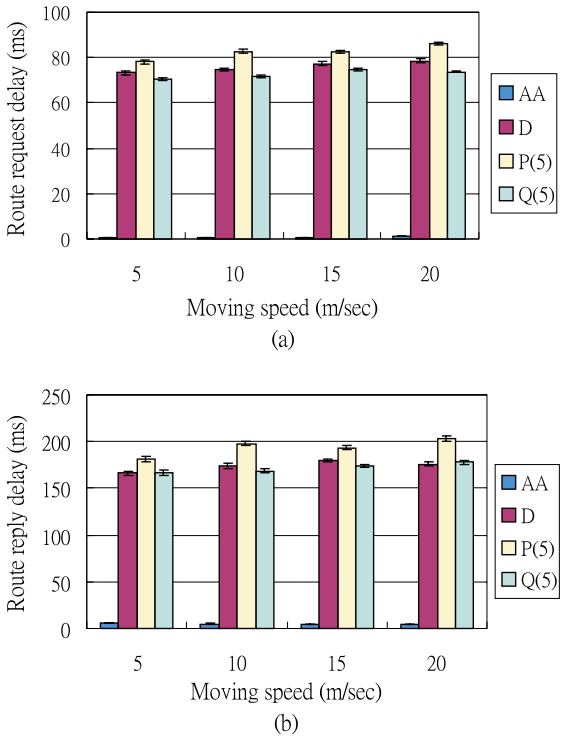


Figure 12: Route request/reply delay vs. mobility. (a)route request delay, and (b) route reply delay. (beacon interval = 100 ms, 100 hosts, traffic load = 1 route/sec)

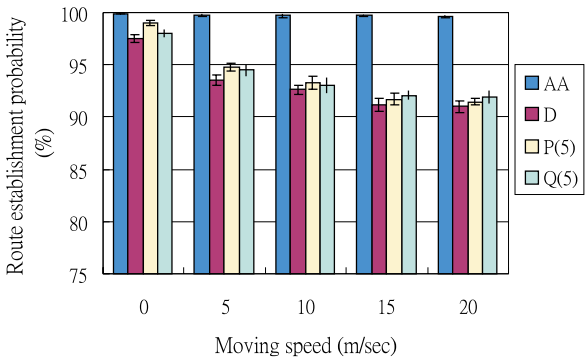


Figure 13: Route establishment probability rate vs. mobility. (beacon interval = 100 ms, 100 hosts, traffic load = 1 route/sec)

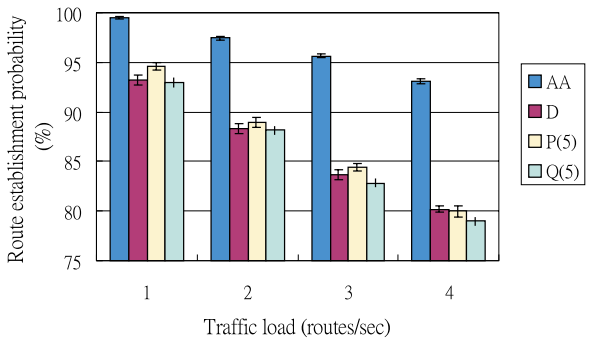


Figure 14: Route establishment probability vs. traffic load. (beacon interval = 100 ms, 100 hosts, mobility = 5 meters/sec)

schemes, the AA scheme performs the best and the $Q(5)$ scheme performs the worst. D and $P(5)$ are close to each other. Figure 15 shows the impact of traffic load on network lifetime. It is reasonable to see that higher traffic load will reduce the network lifetime.

5.4 Impact of Host Density

In this experiment, we fix the network size and vary the total number of hosts between 50 $\sim$ 200. The impact on route establishment probability is in Figure 16. We see that when the host density becomes higher, the route establishment probability somehow gets hurt. In a denser network, collision and congestion may become reasons that cause route establishment failure. Overall, AA performs the best and the $Q(5)$ scheme performs the worst. D and $P(5)$ are close to each other.

Figure 17 shows that a higher node density will bring down the network lifetime. We note that although the traffic load is the same, the broadcast cost to discover routes will become higher. When a route request is issued, not only more hosts will help searching for routes, but also the broadcast cost in each individual host will become higher as the network is denser.

6 Conclusions

In this paper, we have addressed the power management problem in a MANET, which is characterized by unpredictable mobility, multi-hop communication, and no clock synchronization. We have pointed out two important issues, the *neighbor discovery* problem and the *network-partitioning* problem,

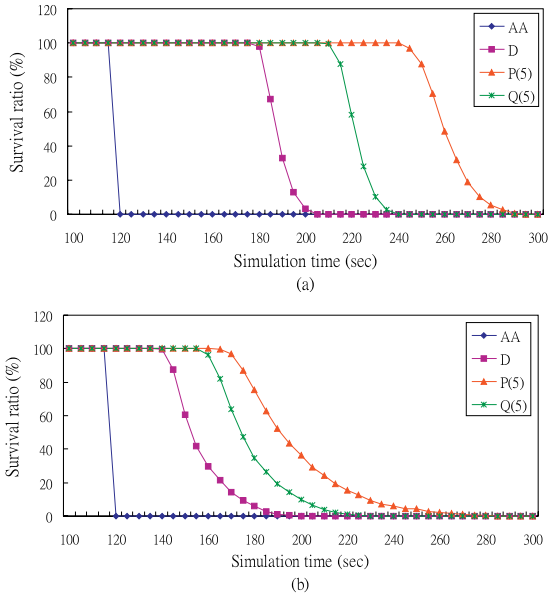


Figure 15: Survival ratio vs. traffic load. (a)traffic load = 1 route/sec, and (b)traffic load = 4 routes/sec. (beacon interval = 100 ms, 100 hosts, mobility = 5 meters/sec)

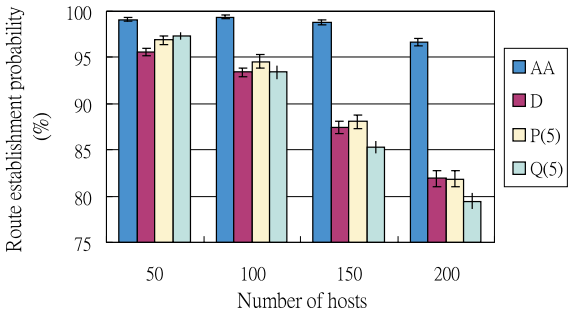


Figure 16: Route establishment probability vs. host density. (beacon interval = 100 ms, traffic load = 1 route/sec, moving speed = 5 m/sec)

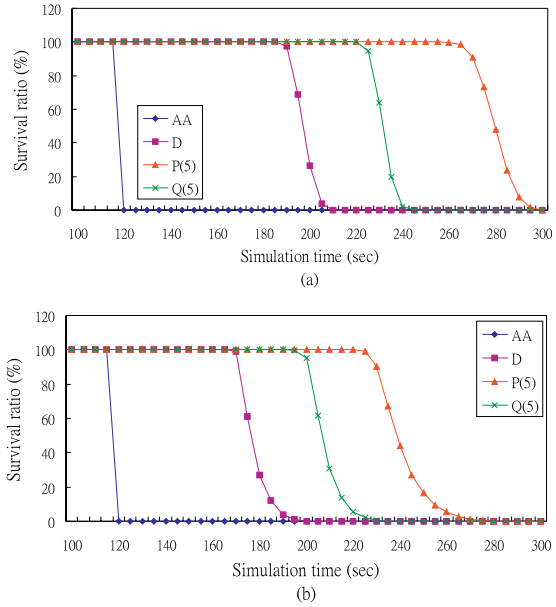


Figure 17: Survival ratio vs. node density. (a)50 hosts, and (b)200 hosts. (beacon interval = 100 ms, traffic load 1 route/sec, mobility = 5 meters/sec)

which may occur if one directly adopts the power-saving (PS) mode defined in the IEEE 802.11 protocol. As far as we know, the power-saving issues, particularly for multi-hop MANETs, have not been addressed seriously in the literature. In this paper, we have proposed three power-saving protocols for IEEE 802.11-based, multi-hop, unsynchronized MANETs. Simulation results have shown that our power-saving protocols can save lots of energies with reasonable route establishment probability. Among the three proposed protocols, the Dominating-Awake-Interval protocol is most energy-consuming but has the shortest neighbor discovery time, while the Periodical-Fully-Awake-Interval protocol is most energy-saving but has the longer route discovery delays. The Quorum-based protocol consumes more energies than the Periodical-Fully-Awake-Interval protocol, but it transmitts fewer beacon frames than the other two protocols and has shorter route discovery delays. We believe that the proposed protocols can be applied to current IEEE 802.11 wireless LAN cards easily with little modification.

References

[1] Nokia, "Wireless Broadband Access–Nokia Rooftop Solution," *Nokia Network References, http://www.wbs.nokia.com/solution/index.html*, 2001.

[2] S. L. Wu, Y. C. Tseng, and J. P. Sheu, "Intelligent Medium Access for Mobile Ad Hoc Networks with BusyTones and Power Control," *IEEE Journal on Selected Areas in Communications*, vol. 18, pp. 1647–1657, Sep 2000.

[3] L. Hu, "Topology Control for Multihop Packet Radio Networks," *IEEE Transactions on Communications*, vol. 41, pp. 1474–1481, Oct 1993.

[4] R. Ramanathan and R. Rosales-Hain, "Topology Control of Multihop Wireless Networks using Transmit Power Adjustment," *IEEE INFOCOM*, pp. 404–413, 2000.

[5] R. Wattenhofer, L. Li, P. Bahl, and Y. M. Wang, "Distributed Topology Control for Power Efficient Operation in Multihop Wireless Ad Hoc Networks," *IEEE INFOCOM*, pp. 1388–1397, 2001.

[6] C. F. Huang, Y. C. Tseng, S. L. Wu, and J. P. Sheu, "Increasing the Throughput of Multihop Packet Radio Networks with Power Adjustment," *International Conference on Computer, Cummunication, and Networks*, 2001.

[7] J. Gomez, A. T. Campbell, M. Naghshineh, and C. Bisdikian, "A Distributed Contention Control Mechanism for Power Saving in random-access Ad-Hoc Wireless Local Area Networks," *Proc. of IEEE International Workshop on Mobile Multimedia Communications*, pp. 114–123, 1999.

[8] J. H. Ryu and D. H. Cho, "A New Routing Scheme Concerning Power-Saving in Mobile Ad-Hoc Networks," *Proc. of IEEE International Conference on Communications*, vol. 3, pp. 1719–1722, 2000.

[9] J. H. Ryu, S. Song, and D. H. Cho, "A Power-Saving Multicast Routing Scheme in 2-tir Hierarchical Mobile Ad-Hoc Networks," *Proc. of IEEE Vehicular Technology Conference*, vol. 4, pp. 1974–1978, 2000.

[10] S. Singh, M. Woo, and C. S. Raghavendra, "Power-Aware Routing in Mobile Ad Hoc Networks," *Proc. of the International Conference on Mobile Computing and Networking*, pp. 181–190, 1998.

[11] I. Stojmenovic and X. Lin, "Power-aware Localized Routing in Wireless Networks," *Proc. of IEEE International Parallel and Distributed Processing Symposium*, pp. 371–376, 2000.

[12] LAN MAN Standards Committee of the IEEE Computer Society, "IEEE Std 802.11-1999, Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications," *IEEE*, 1999.

[13] H. Woesner, J. P. Ebert, M. Schlager, and A. Wolisz, "Power-Saving Mechanisms in Emerging Standards for Wireless LANs: The MAC Level Perspective," *IEEE Persinal Communications*, pp. 40–48, Jun 1998.

[14] J. C. Haartsen, "The Bluetooth Radio System," *IEEE Persinal Communications*, pp. 28–36, Feb 2000.

[15] S. Singh and C. S. Raghavendra, "Power Efficient MAC Protocol for Multihop Radio Networks," *Proc. of IEEE International Personal, Indoor and Mobile Radio Communications Conference*, pp. 153–157, 1998.

[16] C. F. Chiasserini and R. R. Rao, "A Distributed Power Management Policy for Wireless Ad Hoc Networks," *IEEE Wireless Communication and Networking Conference*, pp. 1209–1213, 2000.

[17] J. R. Lorch and A. J. Smith, "Software Strategies for Portable Computer Energy Management," *IEEE Personal Communications*, pp. 60–73, Jun 1998.

[18] A. K. Salkintzis and C. Chamzas, "An In-Band Power-Saving Protocol for Mobile Data Networks," *IEEE Transactions on Communications*, vol. 46, pp. 1194–1205, Sep 1998.

[19] T. Simunic, H. Vikalo, P. Glynn, and G. D. Micheli, "Energy Efficient Design of Portable Wireless Systems," *Proc. of the International Symposium on Low Power Electronics and Design*, pp. 49–54, 2000.

[20] B. Chen, K. Jamieson, H. Balakrishnan, and R. Morris, "Span: An Energy-Efficient Coordination Algorithm for Topology Maintenance in Ad Hoc Wireless Networks," *Proc. of the International Conference on Mobile Computing and Networking*, pp. 85–96, 2001.

[21] Y. Xu, J. Heidemann, and D. Estrin, "Geography-informed Energy Conservation for Ad Hoc Routing," *Proc. of the International Conference on Mobile Computing and Networking*, pp. 70–84, 2001.

[22] A. K. Salkintzis and C. Chamzas, "Performance Analysis of a Downlink MAC Protocol with Power-Saving Support," *IEEE Transactions on Vehicular Technology*, vol. 49, pp. 1029–1040, May 2000.

[23] H. Garcia-Molina and D. Barbara, "How to Assign Votes in a Distributed Systems," *Journal of the ACM*, vol. 32, pp. 841–860, Oct 1985.

[24] D. Agrawal and A. El Abbadi, "An Efficient and Fault-Tolerance Algorithm for Distributed Multual Exclusion," *ACM Transactions on Computer Systems*, vol. 9, pp. 1–20, Feb 1991.

[25] A. Kumar, "Hierarchical Quorum Consensus: A New Algorithm for Managing Replicated Data," *IEEE Transactions on Computers*, vol. 40, pp. 996–1004, Sep 1991.

[26] Y. C. Kuo and S. T. Huang, "A Geometric Approach for Constructing Coteries and k-Coteries," *IEEE Transactions on Parallel and Distributed Systems*, vol. 8, pp. 402–411, Apr 1997.

[27] D. B. Johnson and D. A. Maltz, "Dynamic source routing in ad hoc wireless networks," *Mobile Computing*, pp. 153–181, 1996.

[28] C. E. Perkins and E. M. Royer, "Ad-hoc on-demand distance vector routing," *IEEE Workshop on Mobile Computing Systems and Applications*, pp. 90–100, 1999.

[29] L. M. Feeney and M. Nilsson, "Investigating the Energy Consumption of Wireless Network Interface in an Ad Hoc Networking Environment," *IEEE INFOCOM*, pp. 1548–1557, 2001.

Appendix

A Proof of Theorem 1

Proof. It suffices to prove the theorem under the condition $AW = BI/2 + BW$. Consider any two asynchronous mobile hosts A and B. Without loss of generality, let A's clock be faster than B's clock by $\Delta T = k \times BI + \Delta t$, where $0 \leq \Delta t < BI$ and $k \geq 0$ is an integer. In the following derivation, let's use A's clock as a reference to derive B's clock. Assuming that n and m are non-negative integers, we can derive the following timings for A and B.

- A's active windows: $[n \times BI, n \times BI + BI/2 + BW]$.
- A's beacon windows in odd intervals: $[(2m+1) \times BI, (2m+1) \times BI + BW]$.
- A's beacon windows in even intervals: $[2m \times BI + BI/2, 2m \times BI + BI/2 + BW]$.
- B's active windows: $[n \times BI + \Delta T, n \times BI + BI/2 + BW + \Delta T]$.
- B's beacon windows in odd intervals: $[(2m+1) \times BI + \Delta T, (2m+1) \times BI + BW + \Delta T]$.
- B's beacon windows in even intervals: $[2m \times BI + BI/2 + \Delta T, 2m \times BI + BI/2 + BW + \Delta T]$.

Below, we prove that for host A, in every pair (odd and even) of beacon intervals, there is at least one entire beacon window overlapping with host B's active window, and vice versa. This is done by showing that the former's beacon window will start later than the later's active window, and terminates earlier than the later's active window. We separate from two cases.

- *Case 1*: $0 \leq \Delta t \leq BI/2$
    1. Consider B's odd ($(2m+1)$-th) interval. We prove that B's beacon window will be covered by A's ($2m+1+k$)-th active window. The following derivation shows that B's beacon window starts later than A's active window:
    $(2m+1+k) \times BI \leq (2m+1+k) \times BI + \Delta t = (2m+1) \times BI + k \times BI + \Delta t = (2m+1) \times BI + \Delta T$,
    and the following shows that B's beacon window ends earlier than A's active window:
    $(2m+1) \times BI + BW + \Delta T = (2m+k+1) \times BI + BW + \Delta t \leq (2m+k+1) \times BI + BI/2 + BW$.

2. Consider A's even ($(2m)$-th) interval. We prove that A's beacon window will be covered by B's ($2m - k$)-th active window. The following derivation shows that A's beacon window starts later than B's active window:

   $(2m - k) \times BI + \Delta T = (2m - k) \times BI + k \times BI + \Delta t \leq 2m \times BI + BI/2$,

   and the following shows that A's beacon window ends earlier than B's active window:

   $2m \times BI + BI/2 + BW \leq 2m \times BI + BI/2 + BW + \Delta t = (2m - k) \times BI + BI/2 + BW + k \times BI + \Delta t = (2m - k) \times BI + BI/2 + BW + \Delta T$.

- *Case 2*: $BI/2 < \Delta t < BI$, assume that $\Delta t = BI/2 + \Delta d$, $0 < \Delta d < BI/2$

  1. Consider B's even ($(2m)$-th) interval. We prove that B's beacon window will be covered by A's ($2m + 1 + k$)-th active window. The following derivation shows that B's beacon window starts later than A's active window:

     $(2m + 1 + k) \times BI < (2m + 1 + k) \times BI + \Delta d = 2m \times BI + BI/2 + k \times BI + (BI/2 + \Delta d) = 2m \times BI + BI/2 + \Delta T$,

     and the following shows that B's beacon window ends earlier than A's active window:

     $2m \times BI + BI/2 + BW + \Delta T = 2m \times BI + BI/2 + BW + k \times BI + (BI/2 + \Delta d) = (2m + 1 + k) \times BI + BW + \Delta d < (2m + 1 + k) \times BI + BI/2 + BW$.

  2. Consider A's odd ($(2m + 1)$-th) interval. We prove that A's beacon window will be covered by B's ($2m - k$)-th active window. The following derivation shows that A's beacon window starts later than B's active window:

     $(2m - k) \times BI + \Delta T = (2m - k) \times BI + k \times BI + \Delta t < (2m + 1) \times BI$,

     and the following shows that A's beacon window ends earlier than B's active window:

     $(2m + 1) \times BI + BW < 2m \times BI + BI/2 + BW + (BI/2 + \Delta d) = (2m - k) \times BI + BI/2 + BW + k \times BI + (BI/2 + \Delta d) = (2m - k) \times BI + BI/2 + BW + \Delta T$.

□

B Proof of Theorem 2

Proof. Consider any two asynchronous mobile hosts A and B. Without loss of generality, let A's clock be faster than B's clock by $\Delta T = k \times BI + \Delta t$, where $0 \leq \Delta t < BI$ and k is a non-negative

integer. In the following derivation, let's use A's clock as a reference to derive B's clock. Assuming that n is a non-negative integer and the fully-awake intervals arrive periodically every p intervals, we can derive the following timings for A and B.

- A's active windows: $[n \times p \times BI, (n \times p + 1) \times BI + BW + MW]$.
- A's beacon windows: $[n \times BI, n \times BI + BW]$.
- B's active windows: $[n \times p \times BI + \Delta T, (n \times p + 1) \times BI + BW + MW + \Delta T]$.
- B's beacon windows: $[n \times BI + \Delta T, n \times BI + BW + \Delta T]$.

Below, we prove that for host A, in every p beacon intervals, there is at least one entire beacon window overlapping with host B's active window, and vice versa. This is done by showing that the former's beacon window will start later than the later's active window, and terminates earlier than the later's active window.

- Consider B's $(n \times p)$-th interval. We prove that B's beacon window will be covered by A's $(n \times p + k)$-th active window. The following derivation shows that B's beacon window starts later than A's active window:

  $(n \times p + k) \times BI \leq n \times p \times BI + k \times BI + \Delta t = n \times p \times BI + \Delta T$,

  and the following shows that B's beacon window ends earlier than A's active window:

  $n \times p \times BI + BW + \Delta T = (n \times p + k) \times BI + BW + \Delta t < (n \times p + k + 1) \times BI + BW + MW$.

- Consider A's $(n \times p + k + 1)$-th interval. We prove that A's beacon window will be covered by B's $(n \times p)$-th active window. The following derivation shows that A's beacon window starts later than B's active window:

  $n \times p \times BI + \Delta T = (n \times p + k) \times BI + \Delta t < (n \times p + k + 1) \times BI$,

  and the following shows that A's beacon window ends earlier than B's active window:

  $(n \times p + k + 1) \times BI + BW < (n \times p + 1) \times BI + BW + MW + k \times BI + \Delta t = (n \times p + 1) \times BI + BW + MW + \Delta T$.

□

C Proof of Theorem 3

Proof. Consider any two asynchronous mobile hosts A and B. Without loss of generality, let A's clock be faster than B's clock by $\Delta T = k \times BI + \Delta t$, where $0 \leq \Delta t < BI$ and $k \geq 0$ is an integer.

Assume that A picks row r_1 and column c_1 as its quorum intervals, while B picks row r_2 and column c_2 as its quorum intervals, where $0 \leq r_1, c_1, r_2, c_2 < n$.

In the following derivation, let's use A's clock as a reference to derive B's clock. Assume that a, x, and y are non-negative integers, $0 \leq x, y < n$. We can derive the following timings for A and B.

- A's active window in its chosen row: $[(a \times n^2 + r_1 \times n) \times BI, (a \times n^2 + (r_1 + 1) \times n) \times BI + MW]$.
- A's active window in its chosen column: $[(a \times n^2 + y \times n + c_1) \times BI, (a \times n^2 + y \times n + c_1 + 1) \times BI + MW]$.
- A's beacon window in its chosen row: $[(a \times n^2 + r_1 \times n + x) \times BI, (a \times n^2 + r_1 \times n + x) \times BI + BW]$.
- A's beacon window in its chosen column: $[(a \times n^2 + y \times n + c_1) \times BI, (a \times n^2 + y \times n + c_1) \times BI + BW]$.
- B's active window its chosen row: $[(a \times n^2 + r_2 \times n) \times BI + \Delta T, (a \times n^2 + (r_2 + 1) \times n) \times BI + MW + \Delta T]$.
- B's active window in its chosen column: $[(a \times n^2 + y \times n + c_2) \times BI + \Delta T, (a \times n^2 + y \times n + c_2 + 1) \times BI + MW + \Delta T]$.
- B's beacon window in its chosen row: $[(a \times n^2 + r_2 \times n + x) \times BI + \Delta T, (a \times n^2 + r_2 \times n + x) \times BI + BW + \Delta T]$.
- B's beacon window in its chosen column: $[(a \times n^2 + y \times n + c_2) \times BI + \Delta T, (a \times n^2 + y \times n + c_2) \times BI + BW + \Delta T]$.

Below, we prove that for host A, in every n^2 beacon intervals, there are at least two beacon windows overlapping with host B's active window, and vice versa. This is done by showing that two of the former's beacon windows will start later than the later's active windows, and terminate earlier than the later's active windows. Assume that $c_1 - k = -p \times n + u$, $c_2 + k = q \times n + v$, where p, q, u, and v are nonnegative integers, $0 \leq v, u < n$.

- We prove that one of B's beacon window in its chosen column will be covered by A's active window in its chosen row.

  Consider B's $(a \times n^2 + (r_1 - q) \times n + c_2)$-th interval in its chosen column. We prove that B's beacon window will be covered by A's active window in its chosen row. The following derivation shows that B's beacon window starts later than A's active window:

  $(a \times n^2 + r_1 \times n) \times BI \leq (a \times n^2 + (r_1 - q) \times n + q \times n + v) \times BI + \Delta t = (a \times n^2 + (r_1 - q) \times n + c_2 + k) \times BI + \Delta t = (a \times n^2 + (r_1 - q) \times n + c_2) \times BI + \Delta T,$

  and the following shows that B's beacon window ends earlier than A's active window:

  $(a \times n^2 + (r_1 - q) \times n + c_2) \times BI + BW + \Delta T = (a \times n^2 + (r_1 - q) \times n + c_2 + k) \times BI + BW + \Delta t = (a \times n^2 + r_1 \times n + v) \times BI + BW + \Delta t \leq (a \times n^2 + (r_1 + 1) \times n) \times BI + MW.$

- We prove that one of B's beacon window in its chosen row will be covered by A's active window in its chosen column.

  Consider B's $(a \times n^2 + r_2 \times n + u)$-th interval in its chosen row and A's $(a \times n^2 + (r_2 + p) \times n + c_1)$-th interval in its chosen column. We prove that B's beacon window will be covered by A's active window. The following derivation shows that B's beacon window starts later than A's active window:

  $(a \times n^2 + (r_2 + p) \times n + c_1) \times BI \leq (a \times n^2 + (r_2 + p) \times n + c_1 - k) \times BI + k \times BI + \Delta t = (a \times n^2 + (r_2 + p) \times n - p \times n + u) \times BI + \Delta T = (a \times n^2 + r_2 \times n + u) \times BI + \Delta T,$

  and the following shows that B's beacon window ends earlier than A's active window:

  Since $u + k = p \times n + c_1 \Rightarrow$

  $(a \times n^2 + r_2 \times n + u) \times BI + BW + \Delta T = (a \times n^2 + r_2 \times n + u + k) \times BI + BW + \Delta t = (a \times n^2 + (r_2 + p) \times n + c_1) \times BI + BW + \Delta t < (a \times n^2 + (r_2 + p) \times n + c_1 + 1) \times BI + MW.$

- We prove that one of A's beacon window in its chosen column will be covered by B's active window in its chosen row.

  - if $(0 < u < n) \Rightarrow$

    Consider A's $(a \times n^2 + (r_2 + p) \times n + c_1)$-th interval in its chosen column. We prove that A's beacon window will be covered by B's active window in its chosen row. The

following derivation shows that A's beacon window starts later than B's active window:

Since $k = p \times n + c_1 - u \Rightarrow$

$(a \times n^2 + r_2 \times n) \times BI + \Delta T = (a \times n^2 + r_2 \times n + k) \times BI + \Delta t = (a \times n^2 + (r_2 + p) \times n + c_1 - u) \times BI + \Delta t < (a \times n^2 + (r_2 + p) \times n + c_1) \times BI$,

and the following shows that A's beacon window ends earlier than B's active window:

Since $p \times n + c_1 = k + u \Rightarrow$

$(a \times n^2 + (r_2 + p) \times n + c_1) \times BI + BW \leq (a \times n^2 + r_2 \times n + k + u) \times BI + BW + \Delta t = (a \times n^2 + r_2 \times n + u) \times BI + BW + \Delta T < (a \times n^2 + (r_2 + 1) \times n) \times BI + MW + \Delta T$.

- if $(u = 0) \Rightarrow$

  Consider A's $(a \times n^2 + (r_2 + p + 1) \times n + c_1)$-th interval in its chosen column. We prove that A's beacon window will be covered by B's active window in its chosen row. The following derivation shows that A's beacon window starts later than B's active window:

  Since $k = p \times n + c_1 \Rightarrow$

  $(a \times n^2 + r_2 \times n) \times BI + \Delta T = (a \times n^2 + r_2 \times n + k) \times BI + \Delta t = (a \times n^2 + (r_2 + p) \times n + c_1) \times BI + \Delta t \leq (a \times n^2 + (r_2 + p + 1) \times n + c_1) \times BI$,

  and the following shows that A's beacon window ends earlier than B's active window:

  $(a \times n^2 + (r_2 + p + 1) \times n + c_1) \times BI + BW \leq (a \times n^2 + (r_2 + 1) \times n + k) \times BI + BW + \Delta t < (a \times n^2 + (r_2 + 1) \times n) \times BI + MW + \Delta T$.

- We prove that one of A's beacon window in its chosen row will be covered by B's active window in its chosen column.

  - if $(0 \leq v < n - 1) \Rightarrow$

    Consider A's $(a \times n^2 + r_1 \times n + v + 1)$-th interval in its chosen row and B's $(a \times n^2 + (r_1 - q) \times n + c_2)$-th interval in its chosen column. We prove that A's beacon window will be covered by B's active window. The following derivation shows that A's beacon window starts later than B's active window:

    $(a \times n^2 + (r_1 - q) \times n + c_2) \times BI + \Delta T = (a \times n^2 + (r_1 - q) \times n + q \times n + v) \times BI + \Delta t = (a \times n^2 + r_1 \times n + v) \times BI + \Delta t < (a \times n^2 + r_1 \times n + v + 1) \times BI$,

    and the following shows that A's beacon window ends earlier than B's active window:

$(a \times n^2 + r_1 \times n + v + 1) \times BI + BW = (a \times n^2 + (r_1 - q) \times n + q \times n + v + 1) \times BI + BW < (a \times n^2 + (r_1 - q) \times n + c_2 + k + 1) \times BI + MW + \Delta t = (a \times n^2 + (r_1 - q) \times n + c_2 + 1) \times BI + MW + \Delta T$.

– if $(v = n - 1) \Rightarrow$

Consider A's $(a \times n^2 + r_1 \times n)$-th interval in its chosen row and B's $(a \times n^2 + (r_1 - q - 1) \times n + c_2)$-th interval in its chosen column. We prove that A's beacon window will be covered by B's active window. The following derivation shows that A's beacon window starts later than B's active window:

$(a \times n^2 + (r_1 - q - 1) \times n + c_2) \times BI + \Delta T = (a \times n^2 + (r_1 - q - 1) \times n + q \times n + v) \times BI + \Delta t = (a \times n^2 + (r_1 - 1) \times n + v) \times BI + \Delta t < (a \times n^2 + r_1 \times n) \times BI$,

and the following shows that A's beacon window ends earlier than B's active window:

$(a \times n^2 + r_1 \times n) \times BI + BW = (a \times n^2 + (r_1 - q - 1) \times n + q \times n + v + 1) \times BI + BW < (a \times n^2 + (r_1 - q - 1) \times n + c_2 + k + 1) \times BI + MW + \Delta t = (a \times n^2 + (r_1 - q - 1) \times n + c_2 + 1) \times BI + MW + \Delta T$.

□

論文名稱：Structures for In-network Moving Object Tracking in Wireless Sensor Networks

作者：Chih-Yu Lin and Y.-C. Tseng

發表情況: ***Broadnets*** 2004

Structures for In-Network Moving Object Tracking in Wireless Sensor Networks *

Chih-Yu Lin and Yu-Chee Tseng
Department of Computer Science and Information Engineering
National Chiao Tung University
Hsin-Chu, 30050, Taiwan
Email: {lincyu, yctseng}@csie.nctu.edu.tw

Abstract

One important application of wireless sensor networks is tracking moving objects. The recent progress has made it possible for tiny sensors to have more computing power and storage space. Therefore, a sensor network can be considered as a distributed database, on which one can conduct in-network data processing. This paper considers in-network moving object tracking in a sensor network. This typically consists of two operations: location update and query. We propose a message-pruning tree structure that is an extension of the earlier work [7], which assumes the existence of a logical structure to connect sensors in the network. We formulate this problem as an optimization problem. The formulation allows us to take into account the physical structure of the sensor network, thus leading to more efficient solutions than [7] in terms of communication costs. We evaluate updating and querying costs through simulations.

1. Introduction

The rapid progress of wireless communication and embedded micro-sensing MEMS technologies have made *wireless sensor networks* possible. Such environments may have many inexpensive wireless nodes, each capable of collecting, processing, and storing environmental information, and communicating with neighboring nodes. In the past, sensors are connected by wire lines. Today, this environment is combined with the novel *ad hoc* networking technology to facilitate inter-sensor communication [10, 11]. The flexibility of installing and configuring a sensor network is thus greatly improved. Recently, a lot of research activities have been dedicated to sensor networks [3, 4, 5, 7, 8, 12, 13].

Object tracking is an important application in wireless sensor networks (e.g. military intrusion detection and habitat monitoring). In [13], the authors present a localized prediction approach for power-efficient object tracking by putting unnecessary sensors into sleep mode. Techniques of cooperative tracking by multiple sensors have been explored in [1, 9, 14]. In [14], the authors propose a *convoy tree*, which is a localized tree rooted at a sensor nearby the current location of the object, where data aggregation is conducted. In [2], a *tracking tree* is proposed to solve a *pursuer-evader* problem in sensor networks. Sensors communicate periodically with neighbors and maintain a dynamic tracking tree that is always rooted at the evader.

In [7], the authors consider a sensor network as a distributed database and propose a scalable message-pruning hierarchy tree called *DAB* (*Drain-And-Balance*) for object tracking. The tree is a logical tree to connect all sensors and each internal node in the tree maintains a *detected set* containing its descendants' coverage. The main idea is similar to *data aggregation* and *in-network data processing* [3, 5, 6, 8], and thus a lot of redundant transmissions are eliminated.

This work is an extension of the work in [7]. Instead of assuming the existence of a logical tree, we try to realize the logical tree by the sensors directly. In this way, the real communication cost can be evaluated more accurately. We then propose two message-pruning tree structures called *DAT* (*Deviation-Avoidance Tree*) and *Z-DAT* (*Zone-based DAT*). We formulate communication costs associated with trees. Through simulations, we demonstrate the advantage of our approach.

The remainder of this paper is organized as follows. Section 2 defines the problem. Constructions of message-pruning trees are presented in Section 3. Our simulation results are in Section 4. Section 5 concludes this paper.

*Y. C. Tseng's research is co-sponsored by the NSC Program for Promoting Academic Excellence of Universities under grant number 93-2752-E-007-001-PAE, by Computer and Communications Research Labs., ITRI, Taiwan, by Intel Inc., by the Institute for Information Industry and MOEA, R.O.C, under the Handheld Device Embedded System Software Technology Development Project and the Communications Software Technology Project, and by Chung-Shan Institute of Science and Technology under contract number BC93B12P.

2 Problem Statement

We consider a wireless sensor network deployed in a field for the purpose of object-tracking. Sensors' locations are already known at a special node called *sink*, which serves as the gateway of the sensor network to the outside world. We adopt a simple *nearest-sensor* model, in the sense that the system only tries to identify the sensor that receives the strongest signal from the object. So the sensing field can be partitioned into a Voronoi graph, in which each polygon denotes the sensing range of a sensor, as depicted in Fig. 1(a).

Our goal is not to propose a location-tracking model, but to propose a data-aggregation model for this kind of service. We assume that whenever an object arrives at or departs from the sensing range (polygon) of a sensor, a *detection event* will be reported (note that this report message does not always need to be forwarded to the sink, as will be elaborated later on). Following the model in [7], two sensors are called *neighbors* if their sensing ranges share a common boundary on the Voronoi graph; otherwise, they are *non-neighbors*. We also consider that the sensor network is responsible for tracking a large number of objects that may move frequently in the field (such as workers in an office environment or animals in a farm). So from the mobility history of objects, it is possible to collect the *event rates* between each pair of neighboring sensors. An event rate represents the frequency of objects travelling from one sensor to another in statistics. For example, in Fig. 1(a), the arrival and departure rates between sensors are shown on the edges of the Vonoroi graph.

We consider the sensor network as a database with distributed storages among sensors. In-network information processing will be conducted to support object-tracking services. Therefore, the events detected by sensors are not always reported to the sink. The problem is modelled as follows. The communication range of sensors is assumed to be large enough so that neighboring sensors (in terms of their sensing ranges) can communicate with each other directly. So the network topology can be regarded as an undirected weighted graph $G = (V_G, E_G)$ with V_G representing sensors and E_G representing links between neighboring sensors. The weight of each link $(a, b) \in E_G$, denoted by $w_G(a, b)$, is the sum of event rates from a to b and b to a. This is because both arrival and departure events will be reported in our scheme. So both a and b will experience the same rate (arrival plus departure). For example, Fig. 1(b) shows the graph corresponding to the sensor network in Fig. 1(a).

Given G, our goal is to construct from G a logical weighted tree $T = (V_T, E_T)$, called a *message-pruning tree*, such that the total communication cost is as low as possible, where $V_T = V_G$ and E_T consists of $|V_T| - 1$ edges with the sink as the root. Intuitively, T is a logical tree constructed from G, in which each edge $(a, b) \in E_T$ is a shortest path connecting sensors a and b in G. So the weight of each edge $(a, b) \in E_T$, denoted by $w_T(a, b)$, is modelled by the minimum hop count between a and b in G. For example, Fig. 2(a) shows a logical message-pruning tree T for the network G in Fig. 1(b). Our purpose is to construct such a message-pruning tree T for in-network object-tracking. We adopt an aggregation model similar to the STUN hierarchy proposed in [7]. Specifically, each node a in T maintains a *detected list* $\boldsymbol{DL}_a = (L_0, L_1, \ldots, L_k)$ such that L_0 is the set of objects currently inside the coverage of sensor a itself, and $L_i, i = 1, \cdots, k$, is the set of objects currently inside the coverage of any sensor who is in the subtree rooted by the i-th child of sensor a, where k is the number of children of a. When an object o moves from sensor a's to b's sensing range such that $(a, b) \in E_G$, a departure event $dep(o, a, b)$ and an arrival event $arv(o, b, a)$ will be reported by a and b, respectively, alone the tree T. On receiving such an event, a sensor x takes the following actions:

- If the event is $dep(o, a, b)$, x will remove o from the proper L_i in $\boldsymbol{DL}_x$ such that sensor a belongs to the i-th subtree of x in T. If $x = a$, then o will be removed from L_0 in $\boldsymbol{DL}_a$. Then x checks whether sensor b belongs to the subtree rooted at x in T or not. If not, the event $dep(o, a, b)$ is forwarded to the parent node of x in T.

- If the event is $arv(o, b, a)$, x will add o to the proper L_i in $\boldsymbol{DL}_x$ such that sensor b belongs to the i-th subtree of x in T. If $x = b$, then o will be added to L_0 in $\boldsymbol{DL}_b$. Then x checks whether sensor a belongs to the subtree rooted at x in T or not. If not, the event $arv(o, b, a)$ is forwarded to the parent node of x in T.

The above event-reporting actions guarantee that, disregarding transmission delays, the data structure $\boldsymbol{DL}_i$ always maintains the objects under the coverage of each subtree of each sensor i in T. So searching the location of an object can be done efficiently in T. For example, Fig. 2(a) shows how to query the location of Car1 in T. Fig. 2(b) shows how events are reported as Car1 and Car2 move and how we query the new location of Car1.

Based on the above formulation, we define a cost function of T by counting the number of events transmitted in the network:

$$C(T) = \sum_{(u,v)\in E_G} w_G(u, v) * (dist_T(u, par(u, v)) + dist_T(v, par(u, v))), \quad (1)$$

where $par(u, v)$ denotes the root of the minimum subtree in T that includes both u and v, and $dist_T(x, y)$ is

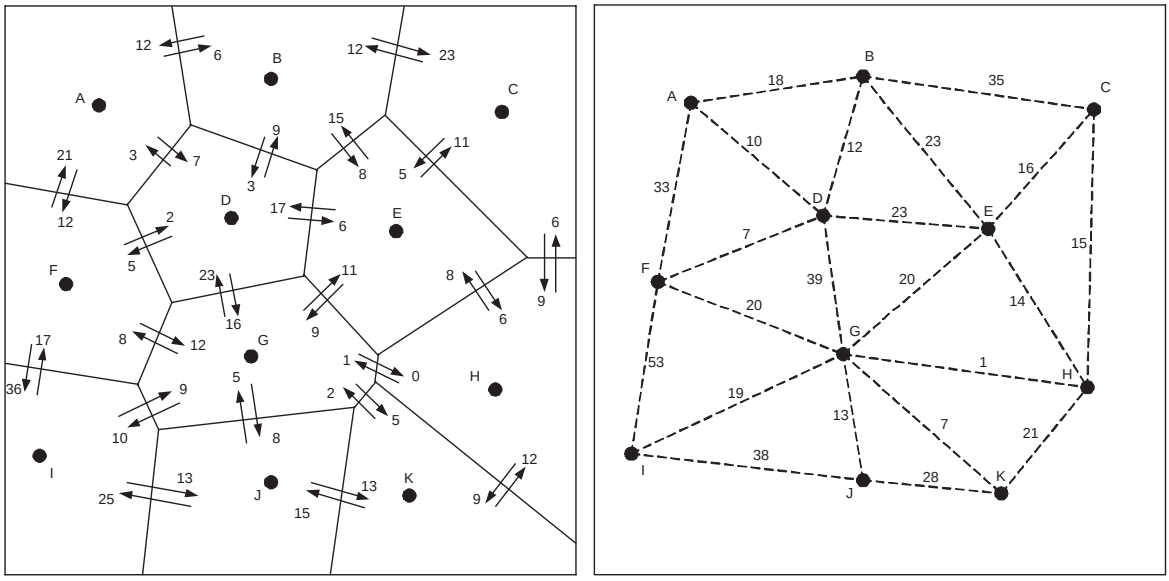


Figure 1. (a) The Voronoi graph of a sensor network. The arrival and departure rates between sensors are the numbers associated with arrows. (b) The graph G corresponding to the sensor network in (a). The number labelled on each edge represents its weight.

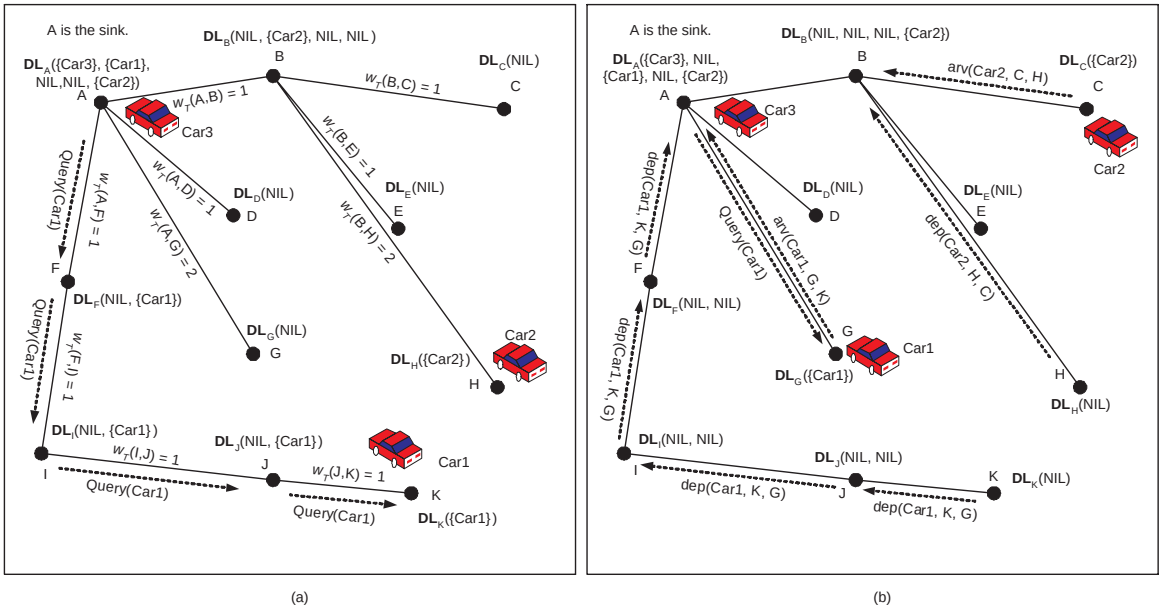


Figure 2. (a) A logical message-pruning tree T constructed from G in Fig. 1. Dotted lines show how to query the location of Car1. (b) The event reporting when Car1 moves from sensor K to G and Car2 moves from H to C.

the sum of weights of the edges on the path connecting x and y in T. For example, in Fig. 2(a), $dist_T(F,K) = w_T(F,I) + w_T(I,J) + w_T(J,K) = 3$, although the minimum hop count between F and K is 2 in G. In order to realize what factors can impact the value of $C(T)$, we will show that $C(T)$ can be expressed by another form.

Theorem 1. *Given any logical tree T of the sensor network G, we have*

$$C(T) = \sum_{(u,v)\in E_T, u=par(v)} w_T(u,v) \times \sum_{(x,y)\in E_G, x\in Subtree(v), y\notin Subtree(v)} w_G(x,y), \quad (2)$$

where $Subtree(v)$ is the subtree of T rooted at node v.

Proof. This can be proved by observing whether an event between two sensors will be reported along a link in T. Consider any $(u,v) \in E_T$. For any $(x,y) \in E_G$, any event between sensors x and y such that $x \in Subtree(v)$ and $y \notin Subtree(v)$ will cause a message being transmitted from v to u. Further, no other event will cause any message being transmitted from v to u. This leads to the theorem. □

Note that although our derivation starts from the construction of graph G from the Vonoroi diagram, one may construct G from different ways and the rest of the derivation will be similar. For example, when there are obstacles in the sensing field, using Voronoi diagram to construct G may not be reasonable. One may simply construct G according to the adjacency relationship between sensors, as illustrated in Fig. 3.

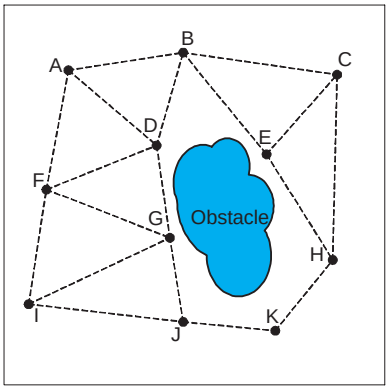


Figure 3. An example of constructing network G in a sensing field with an obstacle.

3 Tree Construction Algorithms

In this section, we propose two algorithms to construct message-pruning trees. The first solution is a greedy approach based on a deviation-avoidance idea. Then we introduce a zone-based approach to further reduce the cost when sensors are deployed like a grid.

3.1 A Greedy Deviation-Avoidance Tree

Our goal is to find a tree T that incurs a low $C(T)$. From Eq. 1 and Eq. 2, we make three observations about $C(T)$:

- From Eq. 1, we observe that the minimal value of $dist_T(u, par(u,v))$ is $dist_G(u, par(u,v))$, where $dist_G(u, par(u,v))$ denotes the minimum hop count between sensor u and sensor $par(u,v)$. In general, we would expect that $dist_T(u, sink) = dist_G(u, sink)$ for each $u \in V_G$; otherwise, we say that u *deviates from* its shortest path to the sink. If this condition is met, we say that T is *deviation-free*. For example, the trees in Fig. 4(a), (c), and (d) are all deviation-free trees corresponding to the graph G in Fig. 1(b). On the contrary, Fig. 4(b) is not a deviation-free tree, because $dist_T(E,A) = 3$ but $dist_G(E,A) = 2$

- From Eq. 2, we observe that the minimal value of $w_T(u,v)$ is 1 when $u \neq v$, i.e. not only $(u,v) \in E_T$ but also $(u,v) \in E_G$. Therefore, we would expect that each sensor's parent is its neighbor. Only the tree shown in Fig. 4(d) satisfies this observation. Conducting this observation to Eq. 1, it means that the average value of $dist_T(u, par(u,v)) + dist_T(v, par(u,v))$ is reduced. For example, the average values of $dist_T(u, par(u,v)) + dist_T(v, par(u,v))$ are 3.591, 2.864, and 2.227 in Fig. 4(a), (c), and (d) respectively.

- In Eq. 1, the weight $w_G(u,v)$ will be multiplied to $dist_T(u, par(u,v)) + dist_T(v, par(u,v))$. For two edges (u,v) and $(u',v') \in E_G$ such that $w_G(u,v) > w_G(u',v')$, we would expect that $dist_T(u, par(u,v)) + dist_T(v, par(u,v)) < dist_T(u', par(u',v')) + dist_T(v', par(u',v'))$. Based on this observation and the second observation, an edge (u,v) with a higher $w_G(u,v)$ should be merged into T early and $par(u,v)$ should be either u or v.

Based on above observations, we design our message-pruning tree. The detailed algorithm is shown in Fig. 5. Initially, we treat each sensor as a singleton subtree. Then we sort E_G into a list L according to their event rates in a decreasing order and examine edges in L one by one. This step is due to the third observation. For each edge (u,v) being examined, (u,v) will be included into T only if u

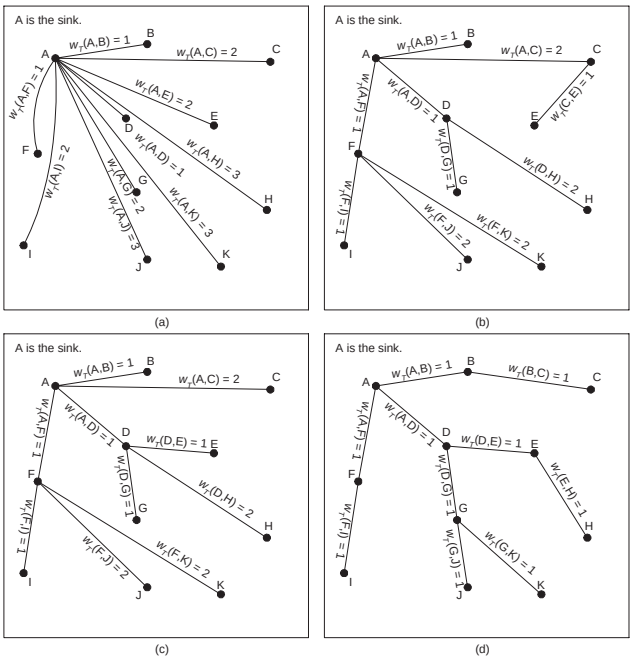


Figure 4. Four possible message-pruning trees with the corresponding graph shown in Fig. 1(b). Those trees in (a), (c), and (d) are deviation-avoidance trees. However the tree in (b) is not a deviation avoidance tree, since $dist_T(E, A)$ is three and $dist_G(E, A)$ is two. The average values of $dist_T(u, par(u, v)) + dist_T(v, par(u, v))$ for each $(u, v) \in E_G$ are 3.591, 2.864, and 2.227 in (a), (c), and (d) respectively.

and v belong to different subtrees in T. Also, (u, v) will be included into T only if at least one of them is a root of a subtree and the other node is on a shortest path in G from the former node to the sink (these conditions are reflected by the ***if*** statements in lines 7 and 10). A link passing these checking will then be included into T. Note that without these conditions, deviation as mentioned in the first observations may occur. Also, as a result of our construction, T is always a subgraph of G (so the second observation is followed) and $w_T(u, v) = 1$ for all $(u, v) \in E_T$.

Theorem 2. *If G is connected, then the T constructed by the DAT algorithm is connected, deviation-free, and is a shortest path tree rooted at the sink.*

Proof. First, we show that T is connected. Each sensor x can be the root exactly once. Because the G is connected, each sensor always can find a neighboring sensor y such that $dist_G(x, sink) = dist_G(y, sink) + 1$, except the sink. Therefore, we can get a connected T rooted at the sink eventually. Second, we show that T is a shortest path tree rooted at the sink. There are two key factors in the algorithm that make the tree be a shortest path tree rooted at the sink. First, we choose the sensor, called v, whose value of $dist_G(v)$ is larger than another sensor, called u, as a child. If the tree path from sensor u to the sink is the shortest path, then the path from sensor v to the sink will also be the shortest path. Second, we only connect these two vertices when v itself is the root of the subtree containing v. This guarantees that all of nodes in the subtree rooted at v will not deviate, because the root is always the one who is closest to the sink in the subtree. If sensor v is not a root, the tree must not be a shortest path tree for the sink. For example, there are two subtrees in Fig. 6, one rooted at sensor 1, another rooted at sensor 2. If we include the edge (u, v) to the message-pruning tree and make the new tree to be rooted at sensor 1, then sensor w and sensor 2 will suffer the deviation problem. If we make the new tree to be rooted at sensor 2, the similar problem will still occur. Therefore, these two factors make the tree be a shortest path tree rooted at the sink. □

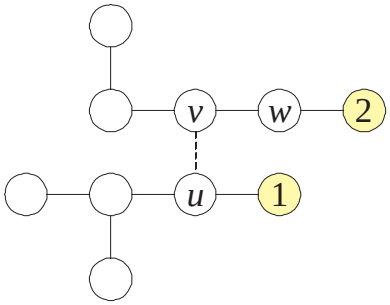


Figure 6. Two subtrees rooted at sensor 1 and sensor 2 respectively. The subtree rooted at sensor 1 and the subtree rooted at sensor 2 cannot be merged by adding the edge (u, v) to the message-pruning tree. Otherwise, the deviation problem occurs.

3.2 A Zone-based Deviation-Avoidance Tree

The algorithm proposed in this subsection is derived based on Eq. 2. Consider the term $\sum_{(x,y)\in E_G, x\in Subtree(v), y\notin Subtree(v)} w_G(x, y)$ in $C(T)$. Let u be v's patent in T. According to our discussion in Section 2, for any edge $(x, y) \in E_G$ such that $x \in Subtree(v)$ and $y \notin Subtree(v)$ any arrival/departure event occurring on (x, y) will cause a message being transmitted on (u, v) (and thus impacts the value of $\sum_{(x,y)\in E_G, x\in Subtree(v), y\notin Subtree(v)} w_G(x, y)$). This leads

DAT(G)

```
1   Let T = (V_T, E_T) such that V_T = V_G and E_T = φ
2
3   Sort E_G into a list L in a decreasing order of their event rates.
4
5   for each (u, v) ∈ E_G in L
6   do if (root(u) ≠ root(v))
7         then if (u = root(u)) ∧ (dist_G(u, sink) = dist_G(v, sink) + 1)
8                 then Let E_T = E_T ∪ (u, v) and let the root of the new subtree be
9                      the original root of the subtree of v.
10              else if (v = root(v)) ∧ (dist_G(v, sink) = dist_G(u, sink) + 1)
11                    then Let E_T = E_T ∪ (u, v) and let the root of the new subtree be
12                         the original root of the subtree of u.
```

Figure 5. The deviation avoidance tree construction algorithm. $root(x)$ denotes the root of subtree that includes x.

to the following interesting observation: the perimeter that bounds the area covered by sensors in $Subtree(v)$ may have a significant impact on the cost function $C(T)$. A longer perimeter usually implies that more events may across the boundary. For example, consider the three $Subtree(v)$ in Fig. 7. Although each subtree has the same number of sensors, the perimeter associated with the subtree in Fig. 7(a) is smaller than that in Fig. 7(b), which is in turn less than that in Fig. 7(c). In fact, in geometry it is well known that a circle has the shortest perimeter to bound the same area among other shapes. Unfortunately, circle-like shapes are difficult to use in a recursive construction. Therefore, in our approach we will adopt square-like shapes as the coverage of each subtree.

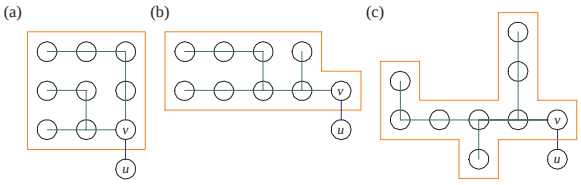


Figure 7. Possible structures of subtrees with nine sensors.

Based on the above observation, we propose a zone-based deviation-avoidance tree construction algorithm. The algorithm is shown in Fig. 8. This algorithm assumes the sensing field to be a square area and takes two input parameters α and δ. T is constructed in an iterative manner. In the first iteration, we first partition the sensing field horizontally into α strips. Further, for each boundary between strips, it is allowed to move up and down no more than a distance of δ from its original position. We will pick a boundary such

Z-DAT(G, α, δ)

```
1   Divide the network into α × α zones.
2   Adjust the zone's boundary according to δ
3   Run DAT(z) for each zone z
4   i ← 1
5
6   while ⌊α/2^i⌋ ≠ 0
7   do Divide the network into ⌊α/2^i⌋ × ⌊α/2^i⌋ zones.
8      Run DAT(z) for each zone z with proper sorted list
9      i ← i + 1
```

Figure 8. The Zone-based Deviation Avoidance tree construction algorithm.

that the total event rate crossing the boundary is the smallest. Next, we partition the sensing field vertically also into α strips. The boundaries between these vertical strips are also adjusted similarly. Using these boundaries, the sensing field is partitioned into $\alpha \times \alpha$ square-like zones. Now in each zone, we run the DAT algorithm in Section 3.1 to form a subtree. This results in α^2 subtrees. Note that the design of parameter δ is based on the third observation in Section 3.1 (so that a boundary with a high total event rate will not be selected).

In the second iteration, we further evenly partition the above $\alpha \times \alpha$ partitions horizontally into $\lfloor \frac{\alpha}{2} \rfloor$ strips, and vertically into $\lfloor \frac{\alpha}{2} \rfloor$ strips. Then in each partition, we merge the subtrees into one by sorting all links connecting the subtrees in this partition according to their event rates, and feeding the sorted list to steps 5 to 12 in algorithm DAT. The rest of the iterations is similar. Specifically, in the i-th iteration, $\lfloor \frac{\alpha}{2^i} \rfloor \times \lfloor \frac{\alpha}{2^i} \rfloor$ subtrees will be formed, until one single tree is

obtained. The idea is illustrated in Fig. 9.

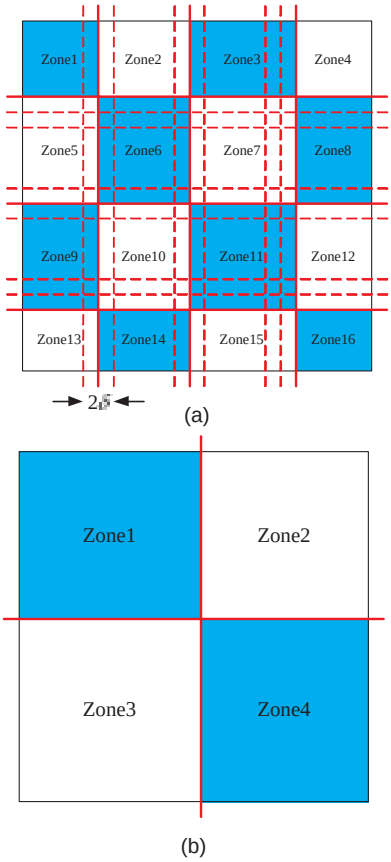


Figure 9. An example of ZDA algorithm with $\alpha = 4$. (a) In the first iteration, we divide the field into $\alpha \times \alpha$ zones and adjust the boundary according to δ. (b) In the second iteration, we divide α by 2 until a single tree is obtained.

4 Simulation Results

In this section, we present our simulation results. Two sensing field models are evaluated. In the first model, we deploy 4096 sensors in a 64×64 grid network, one in each grid. In the second model, we consider a 256×256 grid network in which 4096 sensors are randomly deployed. In both of models, the sink may be located at the center of the network or at one corner of the network.

We adopt the *city mobility model* proposed in [7] to generate the event rates between sensors. Assuming the sensing field to be a square of size $r \times r$, this model evenly divides the sensing field into four sub-regions, called *level-1* regions. Each level-1 region is recursively divided into four smaller sub-regions, called *level-2* regions. Given an object located in any position in the network, it has a probability p_1 to leave its current leavel-1 region, and a probability $1 - p_1$ to stay in its current leavel-1 region. In the former case, it will move either horizontally or vertically with a distance of $r/2$ with an equal probability. In the latter case, it has a probability p_2 to leave its current leavel-2 region, and a probability $1 - p_2$ to stay. Again, in the former case, it will move either horizontally or vertically with a distance of $r/2^2$ with an equal probability, and in the latter case it may cross leavel-3 regions. The process repeats recursively. The probability p_i is determined by an exponential probability $p_i = e^{-C \cdot 2^{d-i}}$, where C is a positive constant and d is the total number of levels. Note that a higher C will exhibit higher locality in movement.

We consider two performance metrics. The first one is the updating cost $C(T)$. The second one is the *querying cost* $Q(T)$, which is defined as the cost spent on transmitting querying requests and replies when objects' locations are inquired from the sink. Note that we only count the numbers of messages without considering message sizes.

We compare our result to a non-message-pruning scheme and the DAB scheme. The non-message-pruning tree scheme always sends all location updates to the sink through the shortest path and thus the sink always has the most up-to-date information. It has no querying cost, but may incur high updating cost. Note that DAB assumes that all sensors are leaf nodes of the message-pruning tree, and there is a logical tree to connect these leaf nodes. To make fair comparison to our schemes, we assume that whenever a new subtree is formed by DAB, the sensor that is closest to the sink will be the root of the subtree (note that this does not thoroughly eliminate the deviation problem).

4.1 Observed Results

First we compare the updating costs $C(T)$. We generate 64 objects, which moves according to the above city mobility model in each time unit. We adopt the 5-step DAB tree construction. For Z-DAT, we set $\alpha = 20$ and $\delta = 1$ grid. We adjust the value of C to reflect moving locality. Inside a 64×64 grid network, Fig. 10(a) and (b) show the results when the sink is at a corner and the center of the network, respectively. A larger C, i.e. a higher moving locality, leads to lower updating cost, which is reasonable. The non-message-pruning scheme has the highest cost because each movement will incur a lot of update messages. Because of the deviation problem, DAB does not perform well either. Our DAT scheme does avoid deviation and thus performs better. Our Z-DAT scheme further reduces the updating cost because the shapes of subtrees are maintained. Fig. 10(c) and (d) show the results when the 256×256 grid network is used. In this case, DAT and Z-DAT schemes perform equally well. The reason is that the Vonoroi dia-

gram now becomes irregular, making maintain the shapes of subtrees in Z-DAT difficult. However, the performance gap between DAT/Z-DAT and DAB is still quite large.

Next, we compare the querying cost $Q(T)$. We vary the querying rate (i.e. the number of queries per unit time) and observe the querying cost (i.e. the number of query and reply messages transmitted per unit time). Fig. 11 shows the results in a 64×64 grid. The querying cost generally increases linearly with the querying rate. Note that there is no querying cost for the non-message-pruning scheme. The querying costs for DAT and Z-DAT schemes are always the same, because querying messages are always transmitted along shortest paths. Not using shortest paths, DAB has higher $Q(T)$ that depends on the tree structure in DAB.

In Fig. 12, we show the combined updating and querying costs. We make two observations. First, as the querying rate becomes higher, using message-pruning tree will gradually lose its advantage because non-message-pruning scheme has no querying cost. Second, as C becomes smaller, objects will move more frequently, thus leading to more saving in using DAB/DAT/Z-DAT.

Next, we vary the parameters α and δ in Z-DAT. Note that $\alpha = 1$ and $\delta = 0$ in Z-DAT is equivalent to DAT. Fig. 13 shows the results. We observe that an α larger than 4 can already obtain good performance for 64×64 grid networks. However, as explained earlier varying α in Z-DAT may not lead to better result than DAT in 256×256 random grid networks. Also, a $\delta = 1$ or 2 may improve the performance of Z-DAT because it can provide some level of flexibility.

5 Conclusions

We have presented two message-pruning structures for moving object tracking in a sensor network. The work extends the earlier work [7] by taking into account the physical topology of a sensor network. This is essential because this can reflect the real communication costs. We are currently investigating more mobility models other than the city mobility model to observe their effects.

References

[1] J. Aslam, Z. Butler, F. Constantin, V. Crespi, G. Cybenko, and D. Rus. Tracking a Moving Object with Binary Sensors. In *Proc. of ACM SenSys*. ACM Press, November 2003.

[2] M. Demirbas, A. Arora, and M. Gouda. A Pursuer-Evader Game for Sensor Networks. *Sixth Symposium on Self-Stabilizing Systems(SSS'03)*, 2003.

[3] D. Ganesan, R. Cristescu, and B. Beferull-Lozano. Power-Efficient Sensor Placement and Transmission Structure for Data Gathering under Distortion Constraints. In *Proc. of Int'l Workshop on Information Processing in Sensor Networks(IPSN)*, 2004.

[4] C.-F. Huang and Y.-C. Tseng. The Coverage Problem in a Wireless Sensor Network. In *Proc. of ACM Int'l Workshop on Wireless Sensor Networks and Applications(WSNA)*, Sep. 2003.

[5] C. Intanagonwiwat, R. Govindan, and D. Estrin. Directed Diffusion: A Scalable and Robust Communication Paradigm for Sensor Networks. In *Proc. of 6th ACM/IEEE Mobicom Conference*, 2000.

[6] B. Krishnamachari, D. Estrin, and S. Wicker. Modelling Data-Centric Routing in Wireless Sensor Networks. In *Proc. of IEEE Infocom*, 2002.

[7] H. T. Kung and D. Vlah. Efficient Location Tracking Using Sensor Networks. In *Proc. of 2003 IEEE Wireless Communications and Networking Conference(WCNC)*, March 2003.

[8] S. Madden, R. Szewczyk, M. J. Franklin, and D. Culler. Supporting Aggregate Queries over Ad-Hoc Wireless Sensor Networks. In *Proc. of 4th IEEE Workshop on Mobile Computing and Systems Application*, 2002.

[9] K. Mechitov, S. Sundresh, and Y. Kwon. Cooperative Tracking with Binary-Detection Sensor Networks. *University of Illinois at Urbana-Champaign, Technical Report UIUCDCS-R-2003-2379*, 2003.

[10] G. J. Pottie and W. J. Kaiser. Wireless Integrated Network Sensors. *Communications of the ACM*, 43(5):51–58, 2000.

[11] K. Sohrabi, J. Gao, V. Ailawadhi, and G. J.Pottie. Protocols for Self-organization of a Wireless Sensor Network. *IEEE Personal Communications*, 7(5):16–27, Oct. 2000.

[12] Y.-C. Tseng, S.-P. Kuo, H.-W. Lee, and C.-F. Huang. Location Tracking in Wireless Sensor Network by Mobile Agents and Its Data Fusion Strategies. In *Proc. of Int'l Workshop on Information Processing in Sensor Networks(IPSN)*, 2003.

[13] Y. Xu and W.-C. Lee. On Localized Prediction for Power Efficient Object Tracking in Sensor Networks. In *Proc. of Int'l Workshop on Mobile Distributed Computing (MDC)*, May 2003.

[14] W. Zhang and G. Cao. DCTC: Dynamic Convoy Tree-Based Collaboration for Target Tracking in Sensor Networks. *IEEE Transactions on Wireless Communication*. to appear.

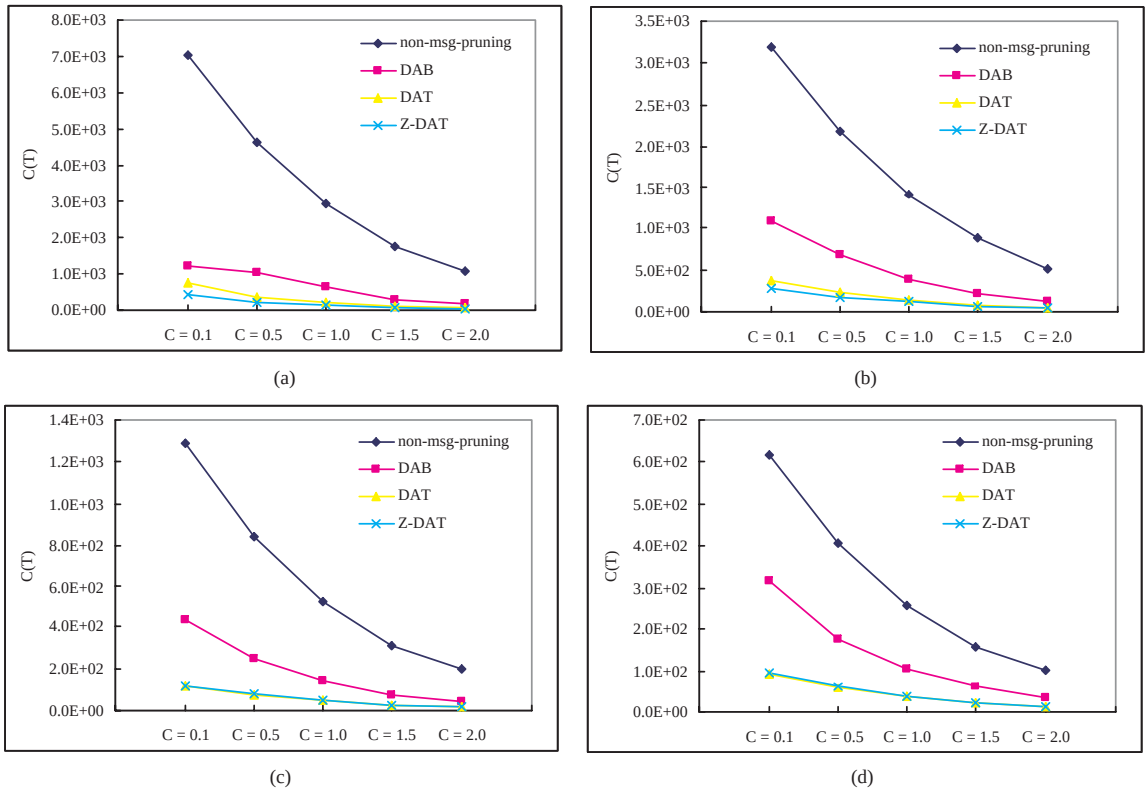


Figure 10. Comparison of updating costs: (a) 64×64 grid, sink at a corner, (b) 64×64 grid, sink at the center, (c) 256×256 grid, sink at a corner, and (d) 256×256 grid, sink at the center.

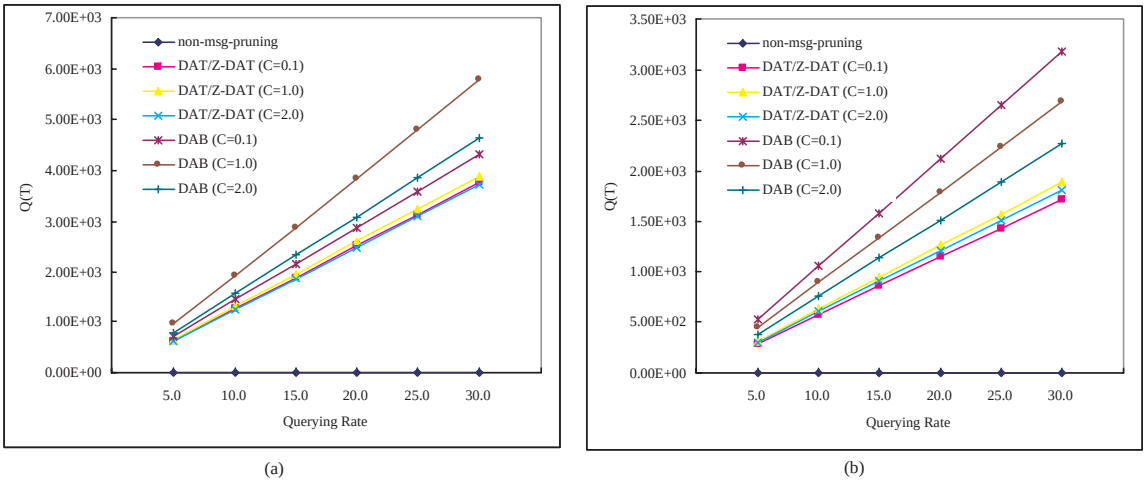


Figure 11. Comparison of querying costs in a 64×64 grid: (a) sink at a corner and (b) sink at the center.

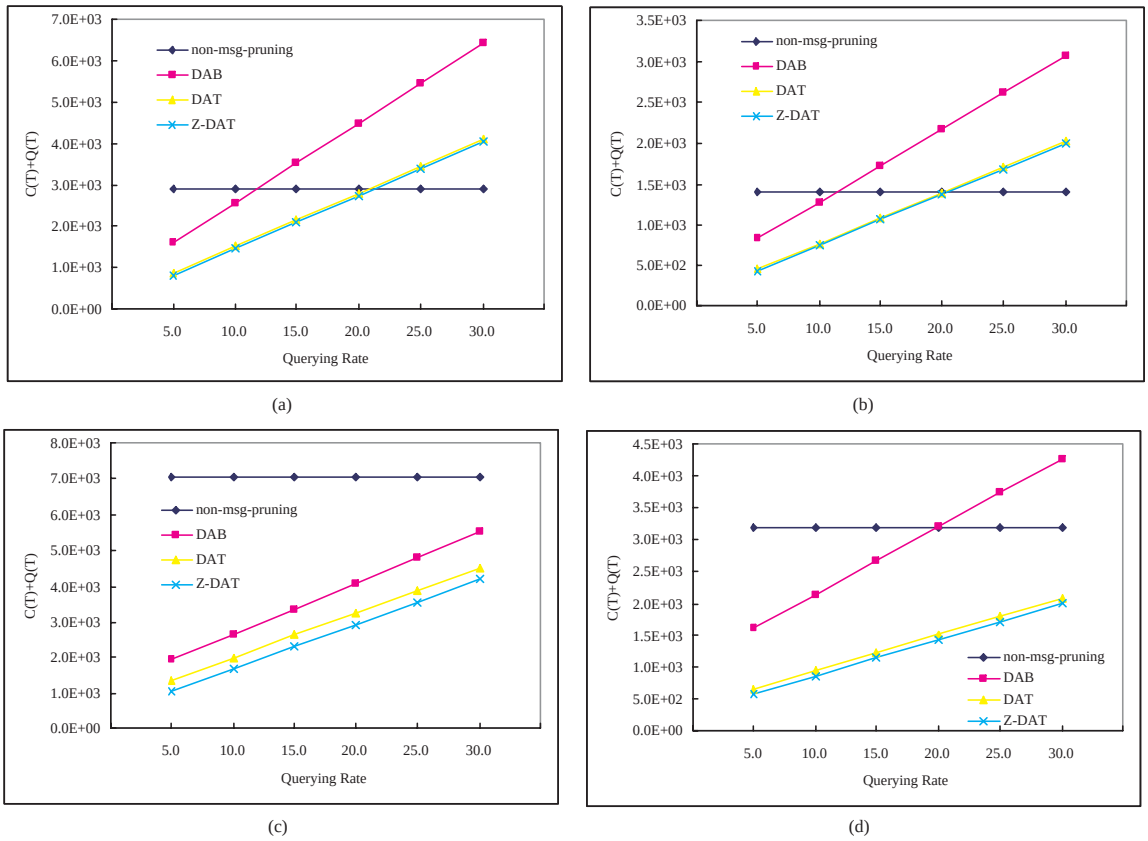


Figure 12. Comparison of combined (updating plus querying) costs in a 64×64 grid: (a) sink at a corner and $C = 1.0$, (b) sink at the center and $C = 1.0$, (c) sink at a corner and $C = 0.1$, and (d) sink at the center and $C = 0.1$.

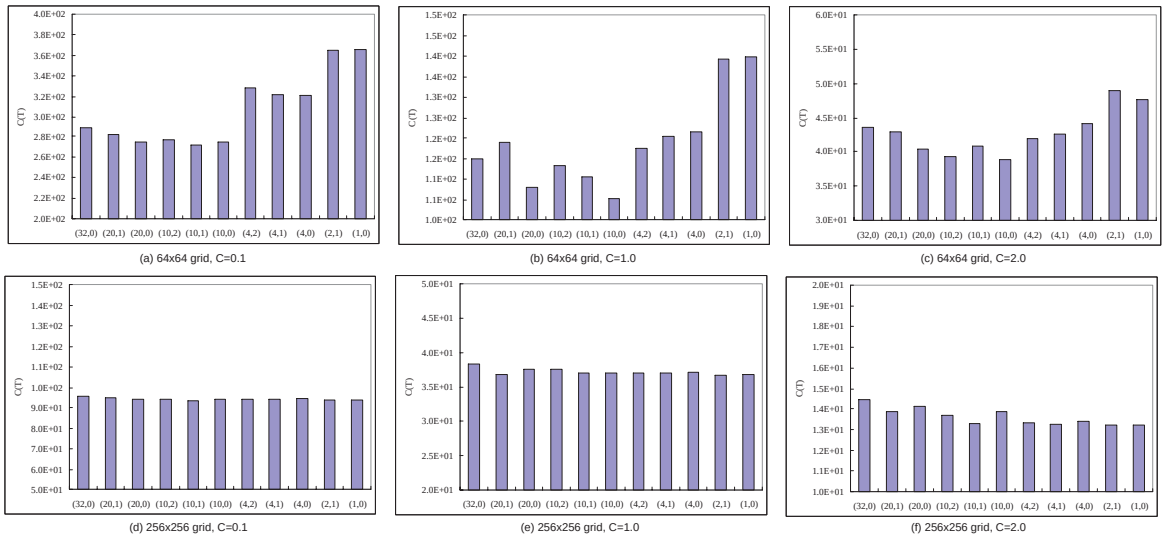


Figure 13. Comparison of updating costs with different (α, δ) pairs. Sinks are located at the center of the network.

論文名稱：Efficient Deployment Algorithms for Ensuring Coverage and Connectivity of Wireless Sensor Networks

作者：You-Chiun Wang, Chun-Chi Hu, and Y.-C. Tseng

發表情況: ***WICON*** 2005

Efficient Deployment Algorithms for Ensuring Coverage and Connectivity of Wireless Sensor Networks

You-Chiun Wang, Chun-Chi Hu, and Yu-Chee Tseng*
Department of Computer Science and Information Engineering
National Chiao Tung University, Hsin-Chu, 30010, Taiwan
Email:{wangyc, cchu, yctseng}@csie.nctu.edu.tw

Abstract

Sensor deployment is a critical issue since it reflects the cost and detection capability of a wireless sensor network. Although lots of work has addressed this issue, most of them assume that the sensing field is an open space and there is a special relationship between the communication range and sensing range of sensors. In this work, we consider the sensing field as an arbitrary-shaped region possibly with obstacles. Besides, we allow an arbitrary relationship between the communication range and sensing range, thus eliminating the constraints of existing results. Our approach is to partition the sensing field into smaller sub-regions based on the shape of the field, and then to deploy sensors in these sub-regions. Simulation results show that our method requires fewer sensors compared to existing results.

1 Introduction

Recently, wireless sensor networks have been studied intensively for applications such as monitoring physical environments. A wireless sensor network is composed of many tiny, low-power nodes that integrate sensing units, transceivers, and actuators with limited on-board processing and wireless communication capabilities [1]. These devices are deployed in a region of interest to gather information from the environment, which will be reported to a remote base station. Wireless sensor networks have been considered in many potential applications, such as surveillance, biological detection, and traffic, pollution, habitat, and civil infrastructure monitoring [2, 3, 6, 11, 13].

Sensor deployment is a critical issue since it reflects the cost and detection capability of a wireless sensor network. A good deployment should consider both *coverage* and *connectivity* [14, 18, 21]. Coverage requires that every location in the sensing field is monitored by at least one sensor. Connectivity requires that the network is not partitioned in terms of nodes' communication capability. Note that coverage is affected by sensors' sensitivity, while connectivity is influenced by sensors' communication ranges.

The *art gallery* problem has been studied extensively previously [7, 15, 17]. The problem asks how to use a minimum set of guards in a polygon such that every point of the polygon is watched by at least one guard. It is typically assumed that a guard can watch a point if line-of-sight exists. So the results cannot be directly applied to the sensor deployment problem since the sensing range of a sensor is normally finite. Besides, the art gallery problem does not address the communication issue between guards. Therefore, several methods have been proposed to solve the deployment problem for sensor networks. The work in [20] mainly discusses how to adjust sensors' locations to satisfy the coverage requirement in an open space, but without considering obstacles. The work in [12, 22] do consider sensing fields with obstacles when deploying sensors, but the results are limited to the special case when communication ranges are equal to sensing ranges. The work in [4, 5, 16] place sensors in a grid-like manner to satisfy coverage and connectivity. However, such approaches are not efficient in terms of the number of sensors being used. How to adaptively put sensors into the sleep mode to save energy while maintain full coverage of the sensing fields is proposed in [10, 19, 21]. The goal is different from our work, which assumes that we can choose the locations to deploy sensors. Also, such work normally assumes that the communication ranges of sensors are much larger than their sensing ranges.

In this work, we consider the sensing field as an arbitrary-shaped region possibly with obstacles. An obstacle can have any shape too. So the results may model an indoor environment. Also, we do not assume any relationship between sensing ranges and communication ranges, thus eliminating the constraints of existing deployment schemes.

*Y. C. Tseng's research is co-sponsored by the NSC Program for Promoting Academic Excellence of Universities under grant number 93-2752-E-007-001-PAE, by Computer and Communications Research Labs., ITRI, Taiwan, and by Intel Inc.

Our approach is to partition the sensing field into smaller sub-regions according to obstacles. Then sensors are deployed in each sub-region. Our simulations show that fewer sensors are required compared to existing results.

The rest of this paper is organized as follows. Section 2 formally defines the problem and reviews some related work. Sections 3 and 4 propose our sensor deployment algorithms. Simulation results are presented in Section 5. Conclusions are drawn in Section 6.

2 Preliminaries

2.1 Problem Definition

We are given a sensing field A in which sensors are to be deployed. Each sensor has a communication range r_c, within which it can transmit packets to other sensors, and a sensing range r_s, within which it can correctly monitor. We assume that all sensors have the same r_c and r_s. However, we make no assumption about the relationship between r_c and r_s. Our goal is to deploy sensors in A to ensure both *sensing coverage* (i.e., every point in A can be monitored) and *network connectivity* (i.e., no sensor gets disconnected) using as few sensors as possible.

The sensing field A is modeled by an arbitrary polygon on a 2D plane. Obstacles may exist inside A, which are also modeled as polygons. However, obstacles do not partition A (otherwise, maintaining network connectivity wouldn't be possible). For obstacles with arc or curve boundaries, we can approximate them by polygons. With the presence of obstacles, we define two sensors S_i and S_j to be *connected* if $|\overline{S_iS_j}| \leq r_c$ and $\overline{S_iS_j}$ does not intersect any obstacle or A's boundary; otherwise, they are *disconnected*. Fig. 1 shows two examples about the connectivity of two sensors. Obstacles may also reduce the coverage of a sensor. We assume that a point can be monitored by a sensor if it is within a distance of r_s and line-of-sight exists with the existence of obstacles. Fig. 2 shows two examples. Note that the above definitions assume that sensors need line-of-sight to sense/communicate. Although the assumption may be conservative, it does guarantee better coverage of the field and better connectivity among sensors. If this assumption is removed, our results can even be simplified. Also note that the sensing field A may already contain some sensors, which can be easily treated as a special case of obstacles.

We conclude the discussion by a sensor deployment example in an office environment as shown in Fig. 3. Note that we assume $r_c = r_s$ in this example.

2.2 Related Work

Some work assumes mobile sensors. The work [22] proposes a *virtual force* algorithm to enhance coverage after an

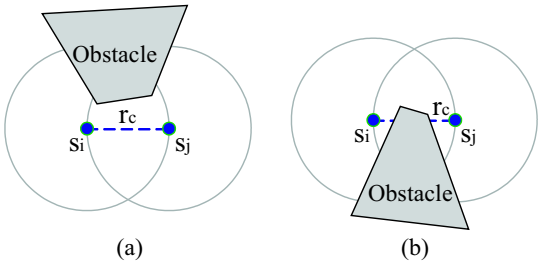


Figure 1. (a) S_i and S_j are connected, and (b) the obstacle disconnects S_i and S_j.

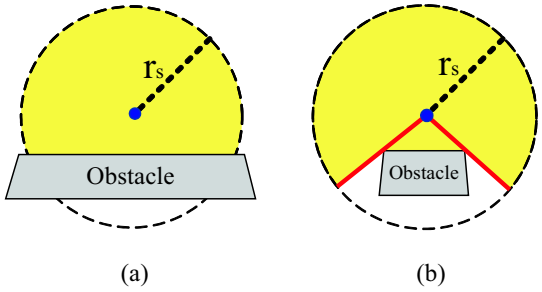


Figure 2. The coverage of a sensor blocked by obstacles (shaded areas are covered).

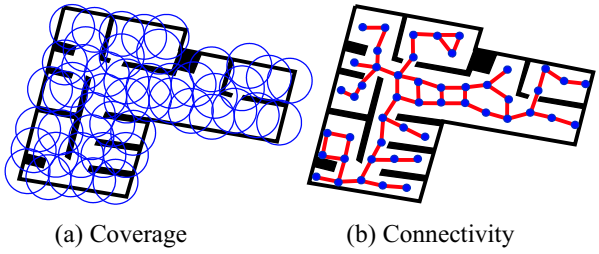


Figure 3. A sensor deployment example in an office environment.

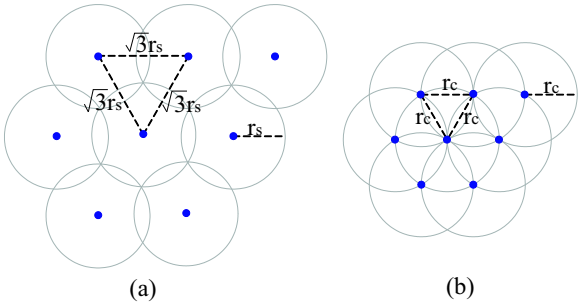
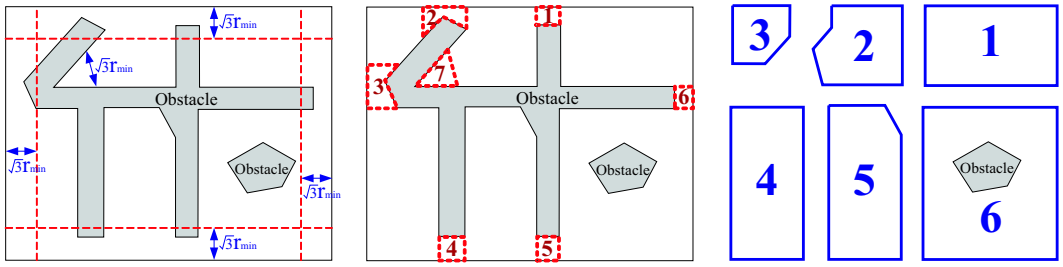


Figure 4. Two intuitive deployment solutions: (a) considering coverage property first and (b) considering connectivity property first.



(a) A sensing field with obstacles (b) Small regions (c) Large regions

Figure 5. Partitioning a sensing field: (a) the sensing field, (b) small regions, and (c) large regions.

initial random placement of sensors. Sensors will be moved by the attractive or repulsive forces of neighboring sensors and obstacles. In [20], Voronoi diagrams are used to discover coverage holes after the initial deployment. Sensors are then moved from densely deployed areas to these holes.

The work in [4, 5, 16] place sensors in a grid-like manner to satisfy coverage and connectivity. It is clear that a hexagon-like placement saves more sensors. So this kind of deployment is not efficient, especially when there are arbitrary relationships between r_c and r_s. Besides, obstacles may destroy the regularity of grids. In [12], it is suggested to deploy sensors along the x-axis by a distance of r_c and then along the y-axis by a distance of $(1 + \frac{\sqrt{3}}{2})r_c$. However, lots of sensors are needed to satisfy connectivity when $r_c \geq \sqrt{3}r_s$. The work [21] indicates that when $r_c \geq 2r_s$, full coverage will guarantee connectivity. Besides, to satisfy full coverage, the distance between adjacent sensors should be $\sqrt{3}r_s$. The result is very limited since only special relationship of r_s and r_c is considered. Also, obstacles are not considered.

Sensor deployment is also addressed in the field of robotics [9, 8]. With robots, sensors can be deployed one by one. The information gathered by deployed sensors can be used to determine the location of the next sensor.

2.3 Two Naive Deployment Algorithms

The sensor deployment problem does pose much challenge. Below, we make some observations based on two extreme solutions. The first one tries to satisfy the coverage property first. In this scheme, to keep a minimal number of sensors, we have to minimize the overlapping coverage as much as possible. The result would be as shown in Fig. 4 (a), where neighboring sensors are evenly separated by a distance of $\sqrt{3}r_s$. This scheme is very efficient when $r_c \geq \sqrt{3}r_s$ since connectivity is automatically guaranteed. However, when $r_c < \sqrt{3}r_s$, extra sensors have to be added to maintain connectivity. Inefficiency may be incurred because all sensing field has been covered and these newly added sensors will not make any contribution to coverage.

The second solution is to satisfy the connectivity property first. This will result in a deployment as shown in Fig. 4 (b), where neighboring sensors are evenly separated by r_c. This scheme will be very efficient when $r_c \leq \sqrt{3}r_s$ because coverage is automatically guaranteed. However, when $r_c > \sqrt{3}r_s$, extra sensors have to be added to maintain coverage. Inefficiency may be incurred because the overlapping coverage could be large.

3 Deployment Algorithms

Given a sensing field A, our goal is to deploy as few sensors as possible to maintain both coverage and connectivity. We first partition A into a number of regions, each being a polygon. Regions are classified as *large* and *small*. We define a *small region* as a belt-like area between obstacles or A's boundary, and its width is not larger than $\sqrt{3}r_{min}$, where $r_{min} = min(r_s, r_c)$. Excluding small regions, the other regions are *large regions*. Fig. 5 gives an example to partition a sensing field. There are seven small regions and six large regions. Note that a region may still exist obstacles, e.g., region 6. How to partition a sensing field will be discussed in Section 4.

Below, we discuss how to deploy sensors in a single region. Note that in our schemes, extra sensors will be deployed on boundaries of regions, so connectivity between different regions are automatically guaranteed.

3.1 Deploying Sensors in Small Regions

For a small region, we can find its bisector and then deploy a sequence of sensors along the bisector to satisfy both coverage and connectivity. How to find a bisector of a region can be achieved by doing a triangulation on that region, as shown in Fig. 6. A bisector can be formed from connecting the midpoints of all dotted lines. Note that if the end of a small region forms a corner (e.g., the case of Fig. 6(b)), then the corner is also considered a midpoint. After finding a bisector, we can deploy a sequence of sensors by a distance of r_{min} along each line segment of the bisector to

ensure coverage and connectivity of that region, as shown in Fig. 6. Note that we always add an extra sensor at the end of the bisector for ensuring connectivity to neighboring regions.

3.2 Deploying Sensors in Large Regions

A region that cannot be simply covered by a sequence of sensors as above is treated as a large region. Multiple rows of sensors will be needed. Below, we first consider a simple large region without boundaries and obstacles. Then we extend our result to an environment with boundaries and obstacles.

3.2.1 Simple Large Regions

Given a 2D plane without boundaries and obstacles, we will deploy sensors row by row. The basic idea is to form a row of sensors that is connected. Adjacent rows should guarantee continuous coverage of the area. Finally, we will add some sensors between adjacent rows, if necessary, to maintain connectivity. Based on the relationship between r_s and r_c, we separate the discussion into two cases.

Case 1: $r_c \leq \sqrt{3}r_s$. In this case, sensors on each row are separated by a distance of r_c. So the connectivity of sensors in each row is already guaranteed. Since $r_c < \sqrt{3}r_s$, each row of sensors can cover a belt-like area with a width of $2 \times \sqrt{r_s^2 - \frac{r_c^2}{4}}$. Adjacent rows will be separated by a distance of $r_s + \sqrt{r_s^2 - \frac{r_c^2}{4}}$ and shifted by a distance of $\frac{r_c}{2}$. With such an arrangement, the coverage of the whole area is guaranteed. Fig. 7(a)–(c) show three possible cases. Note that in the case of $r_c < \sqrt{3}r_s$, the distance between two adjacent rows is larger than r_c, so we need to add a column of sensors between two adjacent rows, each separated by a distance no larger than r_c, to connect them.

Case 2: $r_c > \sqrt{3}r_s$. In this case, the previous approach will waste a lot of sensors because the small r_s requires two rows to be very close. So when $r_c > \sqrt{3}r_s$, we propose to deploy sensors in a typical hexagon manner such that adjacent sensors are regularly separated by a distance of $\sqrt{3}r_s$. Both coverage and connectivity properties are satisfied.

3.2.2 Large Regions with Boundaries and Obstacles

Next, we modify the above solution for deploying sensors in a region with boundaries and obstacles. Observe that in our solution, sensors are deployed in regular patterns. Thus, the above solution can be transformed into an incremental approach where sensors are added into the field one by one. In Table 1, we summarize the coordinates of a sensors's six neighbors. Thus, we can first place a sensor in any location of the region, from which the six locations that can potentially be deployed with sensors are determined.

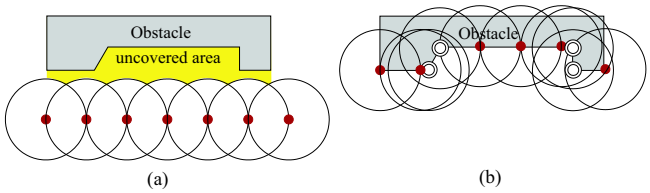


Figure 8. (a) uncovered area around an obstacle, and (b) extra sensors along the boundary to cover the uncovered area.

These locations are inserted into a queue Q. We then enter a loop in which each time an entry (x, y) is dequeued from Q. If (x, y) is not inside any obstacle and not outside of the region, a sensor will be placed in (x, y). Also, the six neighboring locations are calculated according to Table 1 and inserted into Q if they have not be deployed with sensors. This process is repeated until Q becomes empty.

The above approach may leave three problems unsolved. First, some areas near the boundaries or obstacles may be left uncovered. Second, as mentioned before, when $r_c < \sqrt{3}r_s$, we need to add extra sensors between adjacent rows to maintain connectivity. Third, connectivity to neighboring regions needs to be maintained. These problems can be easily solved by sequentially placing sensors along the boundaries of the region and obstacles. Fig. 8 gives an example (we assume that $r_s = r_c$). Note that since obstacles may disconnect adjacent sensors, extra sensors may need to be placed at corners of obstacles (shown by double circles in Fig. 8(b)). There are two cases for the distance between adjacent sensors:

- When $r_c \leq \sqrt{3}r_s$, since the maximum width of the uncovered area does not exceed r_c, sensors should be separated by r_c.
- When $r_c > \sqrt{3}r_s$, since the maximum width of the uncovered area does not exceed $\sqrt{3}r_s$, sensors should be separated by $\sqrt{3}r_s$. Since $r_c > \sqrt{3}r_s$, the connectivity between these extra-added sensors and the regularly deployed sensors are guaranteed.

4 Partitioning a Sensing Field into Small and Large Regions

Section 3 does not explain how to partition the sensing field A into small and large regions. Below, we show how to identify small regions. After excluding small regions, the remaining regions are considered large.

To identify small regions, we first expand the perimeters of obstacles outwardly and A's boundaries inwardly by a distance of $\sqrt{3}r_{min}$. Such an expansion may cause overlapping with the original obstacles and A's boundary. For

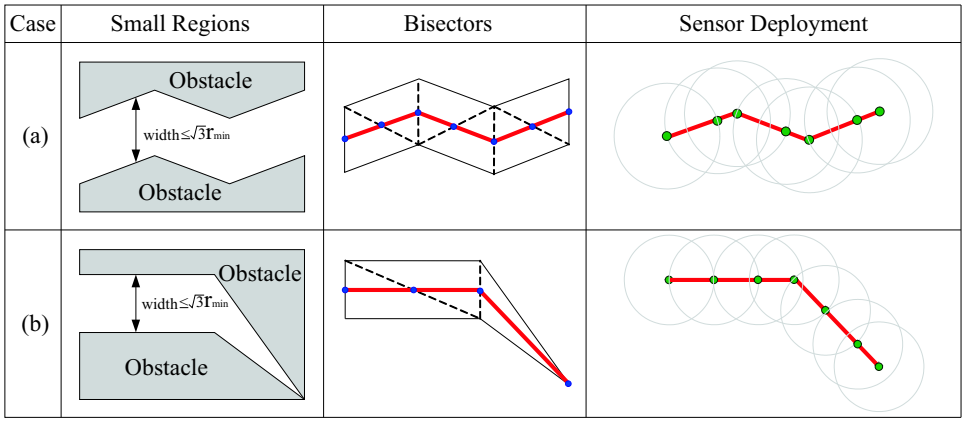


Figure 6. Two examples to find bisectors of small regions and the corresponding sensor deployments.

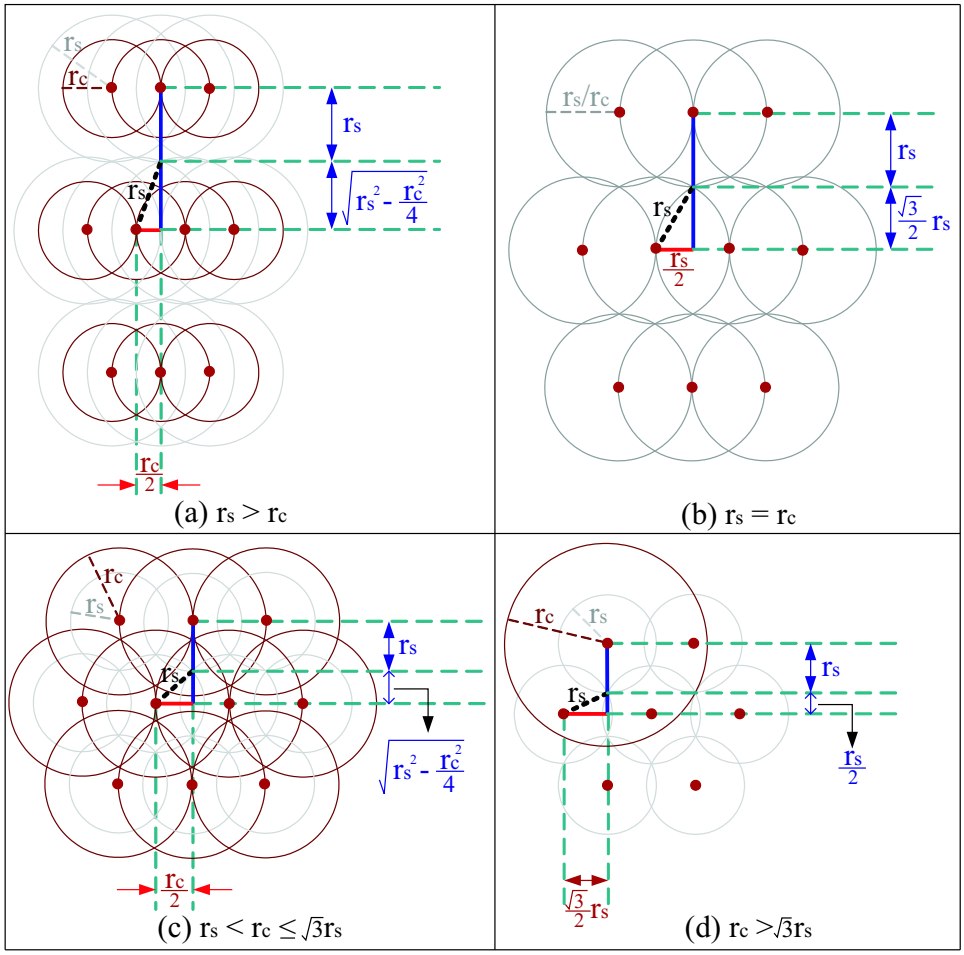


Figure 7. Deploying sensors in simple large regions.

Table 1. Coordinates of the six neighbors of a sensor in location (x, y).

Neighbor	$r_c \leq \sqrt{3}r_s$	$r_c > \sqrt{3}r_s$
N_1	$(x + r_c, y)$	$(x + \sqrt{3}r_s, y)$
N_2	$(x + \frac{r_c}{2}, y - \sqrt{r_s^2 - \frac{r_c^2}{4}} - r_s)$	$(x + \frac{\sqrt{3}r_s}{2}, y - \frac{3r_s}{2})$
N_3	$(x - \frac{r_c}{2}, y - \sqrt{r_s^2 - \frac{r_c^2}{4}} - r_s)$	$(x - \frac{\sqrt{3}r_s}{2}, y - \frac{3r_s}{2})$
N_4	$(x - r_c, y)$	$(x - \sqrt{3}r_s, y)$
N_5	$(x - \frac{r_c}{2}, y + \sqrt{r_s^2 - \frac{r_c^2}{4}} + r_s)$	$(x - \frac{\sqrt{3}r_s}{2}, y + \frac{3r_s}{2})$
N_6	$(x + \frac{r_c}{2}, y + \sqrt{r_s^2 - \frac{r_c^2}{4}} + r_s)$	$(x + \frac{\sqrt{3}r_s}{2}, y + \frac{3r_s}{2})$

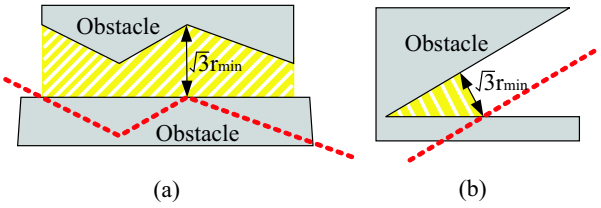


Figure 9. Two examples to find small regions. The dotted lines are expansions of obstacles.

those parts with overlapping, we can take a projection back to the original perimeters to obtain some small regions. Taking Fig. 5(a) as an example, the dotted lines are expansion of A's boundaries. For these overlaps, we can take a projection to obtain small regions, as numbered from 1 to 6 in Fig. 5(b). Fig. 9 shows two examples of the expansions of obstacles. Note that the above expansions may result in multiple different small regions in the same place. In this case, we can select the largest one as a small region.

5 Simulation Results

In this section, we present some experimental results to verify the effectiveness of the proposed sensor deployment algorithm. We design six kinds of sensing fields, as shown in Fig. 10. We consider four cases: (r_s, r_c) = $(7, 5)$, $(5, 5)$, $(3.5, 5)$, and $(2, 5)$ to reflect the relationships of $r_s > r_c$, $r_s = r_c$, $r_s < r_c \leq \sqrt{3}r_s$, and $\sqrt{3}r_s < r_c$, respectively. We mainly compare our algorithm and two deployment methods discussed in Section 2.3 (namely coverage-first and connectivity-first methods). The comparison metric is the average number of sensors being used (for each field, we place the first sensor at different location, and average the results of all deployments).

Fig. 11 compares the number of sensors being used when $r_c \leq \sqrt{3}r_s$ in different sensing fields. The connectivity-first method is dominated by the value of r_c, so the number of sensors is fixed when $r_c \leq \sqrt{3}r_s$. Thus, when $r_s \geq r_c$, this method uses the most sensors because the overlapping in coverage is very large. On the contrary, when $r_s < r_c \leq \sqrt{3}r_s$, the coverage-first method uses the most sensors, because it needs many extra sensors to maintain connectivity between neighboring sensors. The proposed method uses the least sensors because it can adjust the distance between two adjacent rows according to the relationship of r_s and r_c.

Fig. 12 makes a similar comparison when $r_c > \sqrt{3}r_s$. Our algorithm still uses the least sensors in all cases. Note that when $r_c > \sqrt{3}r_s$, our algorithm works the same as the coverage-first method in each individual region, so we omit its performance in Fig. 12.

6 Conclusions

In this work, we have proposed a systematical solution for sensor deployment. The sensing field is modeled as an arbitrary polygon possibly with obstacles. Thus, the result can be used in an indoor environment. The result can be applied to sensors with arbitrary relationships of communication ranges and sensing ranges. Fewer sensors are required to ensure fully coverage of the sensing field and connectivity of the network as compared to other methods. Note that in this work we assume that sensors have predictable communication range r_c and sensing range r_s. This may result in fragile networks when the terrain factor is concerned. To resolve this problem, we can substitute r_c and r_s by r_c' and r_s' which are slightly smaller than r_c and r_s, respectively. This should result in a stronger network. Also, in our solution in Section 3.2.1, we can add more columns of sensors among adjacent rows to improve the reliability of the network.

References

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. A survey on sensor networks. *IEEE Communications Magazine*, 40(8):102–114, 2002.

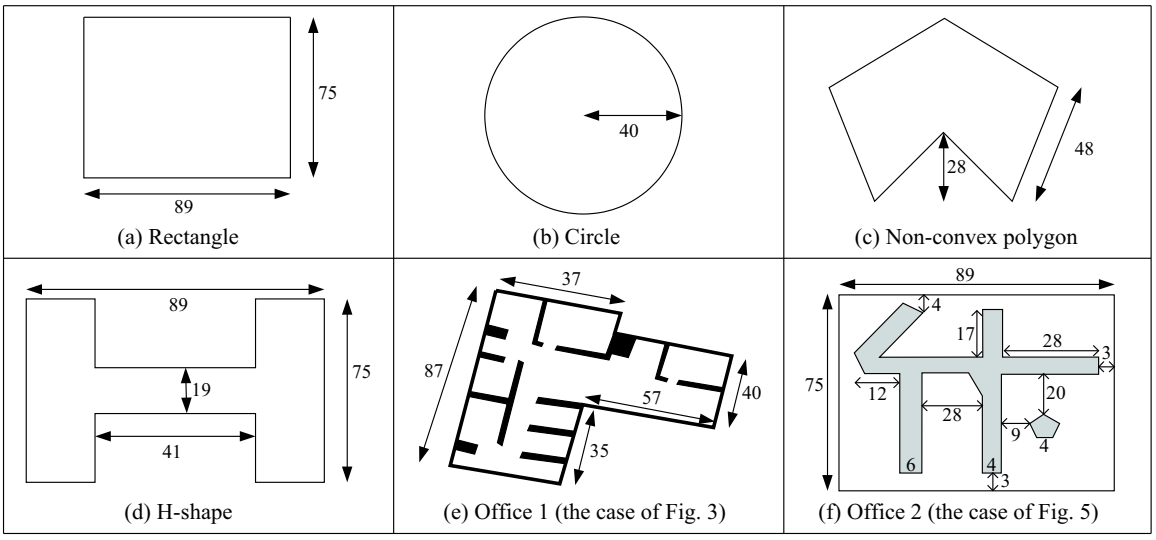


Figure 10. Sensing fields used in the simulations.

[2] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. Wireless sensor networks: a survey. *Computer Networks*, 38(4):393–422, 2002.

[3] J. Burrell, T. Brooke, and R. Beckwith. Vineyard computing: sensor networks in agricultural production. *IEEE Pervasive Computing*, 3(1):38–45, 2004.

[4] T. Clouqueur, V. Phipatanasuphorn, P. Ramanathan, and K. K. Saluja. Sensor deployment strategy for target detection. In *ACM International Workshop on Wireless Sensor Networks and Applications*, pages 42–48, 2002.

[5] S. S. Dhillon, K. Chakrabarty, and S. S. Iyengar. Sensor placement for grid coverage under imprecise detections. In *International Conference on Information Fusion*, pages 1581–1587, 2002.

[6] D. Estrin, R. Govindan, J. Heidemann, and S. Kumar. Next century challenges: scalable coordination in sensor networks. In *ACM/IEEE International Conference on Mobile Computing and Networking*, pages 263–270, 1999.

[7] F. Hoffmann, M. Kaufmann, and K. Kriegel. The art gallery theorem for polygons with holes. In *IEEE Symposium on Foundations of Computer Science*, pages 39–48, 1991.

[8] A. Howard, M. J. Matarić, and G. S. Sukhatme. An incremental self-deployment algorithm for mobile sensor networks. *Auton. Robots*, 13(2):113–126, 2002.

[9] A. Howard, M. J. Matarić, and G. S. Sukhatme. Mobile sensor networks deployment using potential fields: A distributed, scalable solution to the area coverage problem. In *International Symposium on Distributed Autonomous Robotics Systems*, 2002.

[10] C. F. Huang and Y. C. Tseng. The coverage problem in a wireless sensor network. In *ACM International Workshop on Wireless Sensor Networks and Applications*, pages 115–121, 2003.

[11] J. M. Kahn, R. H. Katz, and K. S. J. Pister. Next century challenges: mobile networking for "Smart Dust". In *ACM/IEEE International Conference on Mobile Computing and Networking*, pages 271–278, 1999.

[12] K. Kar and S. Banerjee. Node placement for connected coverage in sensor networks. In *Proceedings of WiOpt*, 2003.

[13] A. Mainwaring, D. Culler, J. Polastre, R. Szewczyk, and J. Anderson. Wireless sensor networks for habitat monitoring. In *ACM International Workshop on Wireless Sensor Networks and Applications*, pages 88–97, 2002.

[14] S. Meguerdichian, F. Koushanfar, M. Potkonjak, and M. B. Srivastava. Coverage problems in wireless ad-hoc sensor networks. In *INFOCOM*, pages 1380–1387, 2001.

[15] I. B. Sachs and D. L. Souvaine. An efficient algorithm for guard placement in polygons with holes. *Discrete & Computational Geometry*, 13:77–109, 1995.

[16] S. Shakkottai, R. Srikant, and N. Shroff. Unreliable sensor grids: coverage, connectivity and diameter. In *INFOCOM*, pages 1073–1083, 2003.

[17] T. C. Shermer. Recent results in art galleries. *Proceedings of the IEEE*, 80(9):1384–1399, 1992.

[18] D. Tian and N. D. Georganas. A coverage-preserving node scheduling scheme for large wireless sensor networks. In *ACM International Workshop on Wireless Sensor Networks and Applications*, pages 32–41, 2002.

[19] D. Tian and N. D. Georganas. A node scheduling scheme for energy conservation in large wireless sensor networks. In *Wireless Communication and Mobile Computing (WCNC)*, pages 271–290, 2003.

[20] G. Wang, G. Cao, and T. L. Porta. Movement-assisted sensor deployment. In *INFOCOM*, pages 2469–2479, 2004.

[21] H. Zhang and J. C. Hou. Maintaining sensing coverage and connectivity in large sensor networks. In *NSF International Workshop on Theoretical and Algorithmic Aspects of Sensor, Ad Hoc Wireless, and Peer-to-Peer Networks*, 2004.

[22] Y. Zou and K. Chakrabarty. Sensor deployment and target localization based on virtual forces. In *INFOCOM*, pages 1293–1303, 2003.

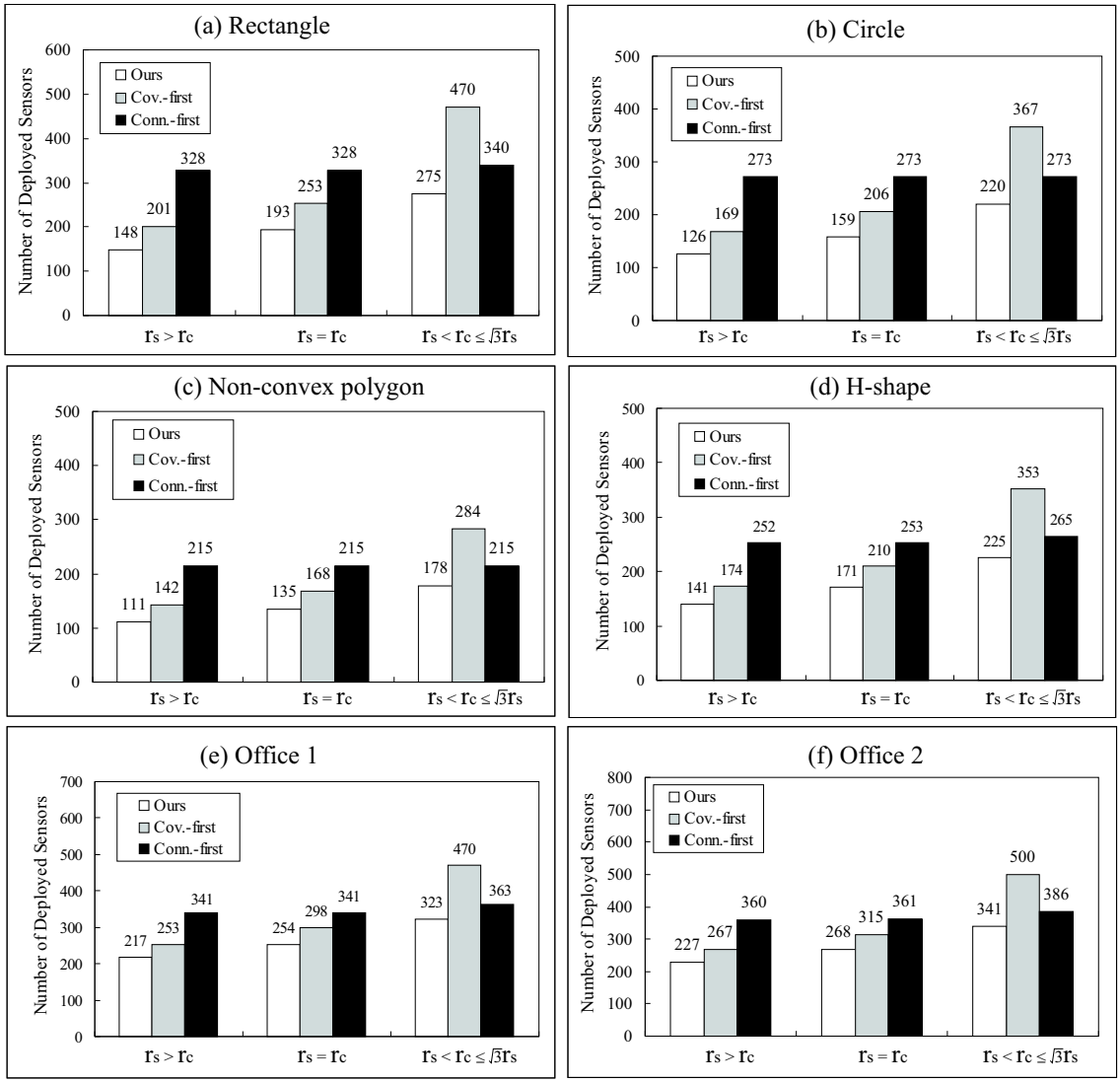


Figure 11. Average number of sensors used when $r_c \leq \sqrt{3}r_s$ under different shapes of sensing fields.

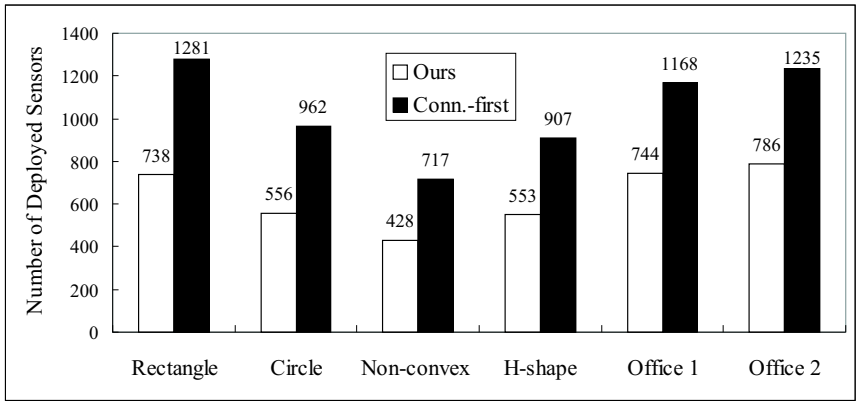


Figure 12. Average number of sensors used when $r_c > \sqrt{3}r_s$ under different shapes of sensing fields.