適用於隨意型無線網路上繞送之省電型虛擬網格基礎架構

Power-saving virtual-grid infrastructure for wireless ad hoc network routing

顏鴻傑[1]、劉如生[1]、何應魁[2]

[1] 元智大學資訊工程研究所
csrobinl@saturn.yzu.edu.tw

[2] 中正理工學院資訊科學系
ykho@ccit.edu.tw

摘要

在隨意式無線網路中的節點通常使用電池來運作，為有效降低電量消耗以提升網路節點的存活時間，本文將提出一個區域集中管理的架構機制(*PVGI*)。經由定位系統裝置之輔助，將實體空間劃分為表格狀區域(grid)，並由每個區域內的節點裏選出一個管理者(server)來負責管理區域內部的節點。基本上，管理者需要定期收集記錄各節點的現況資訊，並依照目前的工作需求來動態調度運用區域內節點，同時讓不需要工作的節點能適時地進入休眠狀態，以節省電量之損耗。此外，我們規定管理者本身不宜擔任閘道節點，意即不負擔正常資料封包之傳送，如此可減少管理者電量之消耗及重新選舉的次數。最後經過系統模擬，顯示我們的方法可有效地降低網路節點電量的消耗，同時提高繞送路徑之穩定性。

關鍵詞：隨意式無線網路，全球定位系統，網格，省電，睡眠模式。

Abstract

In ad hoc network most nodes operate on batteries. To reduce power consumption, i.e., to increase lifetime of mobile nodes, we present a virtual-grid routing architecture, which is named PVGI in this article. With the aid of GPS, we first partition the physical network area into a finite number of grids; each grid has a server with responsibility of managing all of its grid nodes. Basically, the server will maintain each node's current information in its grid, dynamically adjust the node's status according to the different situations, and command the nodes go into sleep mode if they have no works. Besides, the servers do not act as gateways in general, i.e., they do not need to deliver inter-grid data packets. In this way, the servers' power maybe reserved, and the number of re-elections may be reduced. Finally, simulation shows that our protocol reduce energy consumption rate and increase the stability of routing path.

Keywords : ad hoc network, GPS, grid, power-saving, sleep mode

一、緒論

隨意式無線網路是一種多點跳躍式*(multi-hop)*的通訊網路，因為沒有類似存取點或者是基地台這樣的基礎建設，所以當一個移動裝置欲和超過其通訊範圍的移動裝置通訊時，必須透過其他的移動裝置幫忙傳送資料。在隨意式無線網路中，大部分的節點多是使用電池當作電量來源，在找尋繞送路徑時除了考量路徑長度與繞送控制的花費成本外，也需考量節點的電量多寡與耗電程度。

因為實際上無線網路資料的傳送仍需用廣播(broadcast)的方式才能達成，所有在傳送端之傳輸範圍內的節點都會收到封包，不參與周遭相鄰節點通訊工作的節點被迫收到與本身無關的封包的現象稱為 *overhearing*，造成節點額外的電量損耗。在目前的路徑繞送演算法中，大多假設所有的網路節點皆參與繞送工作，亦即節點的無線傳輸介面必須隨時保持在開啟狀態，因此 overhearing 的情況很普遍。目前[1][3][4][5][6][7]皆在研究有效的休眠模式操作方法，讓不參與繞送工作能適時關掉其無線傳輸介面電源（進入休眠模式），即可減低電量損耗。

本文(PVGI)的作法是將實體網路區域劃分成許多的小型區域（*grid*），在每個小型區域內將使用集中式的管理。每個區域內的節點，必須相互選舉出一個節點當管理者(server)，負責管理區域內部的節點，而非管理者節點將允許的進入休眠模式。管理者藉由定期維護各節

點的現況資訊，來協調各個非管理者節點的休眠時間長度，以便當有工作需求時，可以容易調度運用區域內的節點。

為降低維護各 grid 的成本，每當管理者要換手(handoff)時，即先由區域內的選出一個節點來當新的管理者，以避免一次選舉管理者的成本花費。另外，為減輕管理者負擔以兼顧節省電量與效能，管理者本身並不當閘道節點，亦即資料繞送的工作可直接由區域內非管理者節點負責。

本文其他部分規劃如下：相關研究將在第二部分闡述。第三部分，我們會介紹整個 PVGI 的建構與維護，而 PVGI 的路徑繞送演算法將在第四部分說明。第五部分為效能的模擬與結果分析。第六部分則為本文之結論。

二、相關研究

(一) 休眠模式操作

目前藉由操作休眠模式的省電機制，近期的研究著重在利用網路層以上各層的資訊來安排網路節點的休眠：例如在 Span[1]中，各節點根據所在區域裡節點的拓樸關係來決定進入休眠狀態的時機，或是決定是否要當協調者(coordinator)節點。這些協調者節點必須保持在醒著的狀態，來幫助相鄰節點的資料傳輸工作，而非協調者節點則會進入休眠狀態，因此能在不影響區域的網路連結關係下達到省電的效益。GAF[7]則是一種 *grid-based* 架構，亦即每個節點利用 GPS 提供的地理資訊將網路劃分成較小型的網格區域。在 GAF 中，每個網格在同一時間只需有一個醒著的節點去幫助同一網格內節點的資料繞送工作，其他的節點則進入休眠狀態。每個醒著的節點，在經過一段時間就必須回到休眠狀態，原來的工作改由另一節點醒來接替負責，因此可避免節點電量過度消耗的情況發生。

另外如 CPC[5]將所有節點分成 VAP 節點以及與 VAP 節點相鄰的 non-VAP 節點，其中所有 VAP 節點和同時連接兩個以上的 non-VAP 節點會當作 Server，其餘的節點當 Client。Server 須負責其相鄰 Client 的資料傳輸工作，並依當時 Client 連結毀壞率和剩餘電量等因素來決定 Client 的休眠時機與休眠長度。其他如 EDP[4]則是以節點的剩餘電量的機率值以及與區域內相鄰節點斷線的狀況去決定休眠與醒來的時機，來增加節點存活率並改善網路連結。

(二) GRID

GRID[2][6]是一種 *grid-based* 架構的繞送協定。由在 grid 區域內的各節點間以選舉的方式，選出一個剩餘電量最大的節點當作 leader 節點，負責幫忙 grid 區域內所有 non-leader 節點的傳輸工作。

在 GRID 中，leader 節點作為 grid 區域內與區域外的節點溝通橋樑，負責繞送路徑搜尋與資料封包傳送等工作。此外，由於 non-leader 節點在本身不傳輸資料時會進入休眠狀態，此時 leader 節點還須當作這些在休眠狀態的節點的 proxy。

當 leader 節點要離開時，所有非閘道節點就得再次以選舉的方式來選出新的 leader 節點，此時 grid 區域內的節點大多在傳收與選舉相關的控制封包上，而非傳收資料封包。由於 GRID 提供的選舉方法，在當 grid 區域內的節點增加時，花費在選舉上的通訊成本比例也會提高許多。在本文中，我們改用定期維護 grid 區域的方式，將可避免因為 leader 節點的離開而須再次的選舉，亦即減低選舉的次數以減少花費在選舉上的成本。另外，我們不再讓 leader 節點當閘道節點(gateway)，而是由各 grid 區域裡動態選取一個 non-leader 節點負責資料傳輸工作，以降低 leader 節點的工作負擔及電量損耗。

三、省電型虛擬網格基礎架構(PVGI)

本文的目的乃是在改進網路節點的耗電程度，藉以延長整體節點的存活時間。這裡我們提出一種集中式管理的 *grid-based* 架構，稱為省電型虛擬網格基礎架構(PVGI)。以下將說明 PVGI 的建構與維護。

(一) 建置 grid

由於 Grid 是一種正方形的網路區域，在建置 grid 前，我們必須先假定所有網路節點皆認可一個全域的座標原點以及 grid 之邊長 d，且每一個節點均有相同的通訊範圍。每一個 grid 區域皆有一個 grid ID，節點可藉由節點自 GPS 裝置取得的位置座標轉換成它的 grid ID。例如：一節點 A 取得之位置座標是 (x_A, y_A)，則我們可換算得 A 的 grid ID = $(\lfloor \frac{x_A}{d} \rfloor, \lfloor \frac{y_A}{d} \rfloor)$。

由於使用的 grid 邊長大小不同，對同一網路區域所切割出來的 grid 數目也會不同，但前提是相鄰的 grid 之間必須保證能互相通訊。兩個相鄰的 grid，若兩區域內的節點欲正常通訊，必須滿足 $d \le r/\sqrt{5}$，其中 r 是兩點最遠可通訊之傳輸距離，d 是 grid 的邊長。舉例來說：我們取 $d = \frac{r}{\sqrt{5}}$ 可以保證在中央區域內的節

點欲與相鄰四個 grid 互相通訊，如圖 1(a)。同理，取 $d=\frac{r}{2\sqrt{2}}$ 可以保證與相鄰八個 grid 互相通訊，如圖 1 (b)。

(二) 與節點現況資訊相關的時間

我們定義三個時間標記：T_{pass} 代表節點在預估自己還可停留在 grid 內的時間，亦即節點多久之後將會離開目前所在之 grid 範圍；T_{energy} 代表節點由其剩餘電量所預估來自己可以正常運作的時間。我們另取 T_{pass} 與 T_{energy} 較小者為 T_{active}，T_{active} 同時涵蓋了移動性與電量因素，因此節點的現況好壞即可由 T_{active} 來代表。

關於 T_{pass} 的計算：假設每個網路節點除了由 GPS 得到自己目前的位置外，尚可得到目前的速度及移動方向，我們將以這三個參數值來計算出節點的 T_{pass}。

假設 grid 的邊長為 d，內部有一節點 A 座標為 (x_A, y_A)，移動速度為 V_A，相對於水平軸之移動方向為 θ，如圖 2 所示。假定 A 在離開目前所在的 grid 範圍前，不會改變其移動速度與移動方向，則 A 之停留時間 T_{pass} 即是：A 由目前位置移動到 grid 的邊界所花費的時間。以下我們歸納計算 T_{pass} 的公式：

$$T_{pass}=\begin{cases}\min\left(\frac{X_{offset}+d}{V_x},\frac{Y_{offset}+d}{V_y}\right) & 0<\theta<\frac{\pi}{2}\\ \min\left(\frac{X_{offset}}{V_x},\frac{Y_{offset}+d}{V_y}\right) & \frac{\pi}{2}<\theta<\pi\\ \min\left(\frac{X_{offset}}{V_x},\frac{Y_{offset}}{V_y}\right) & \pi<\theta<\frac{3\pi}{2}\\ \min\left(\frac{X_{offset}+d}{V_x},\frac{Y_{offset}}{V_y}\right) & \frac{3\pi}{2}<\theta<2\pi\end{cases}$$

$$X_{offset}=x_0-x_A$$
$$Y_{offset}=y_0-y_A$$
$$V_x=V_A\cos\theta$$
$$V_y=V_A\sin\theta$$

關於 T_{energy} 的計算：假設節點 A 估算其最大耗電功率為 $P_{\max}$，目前的剩餘電量為 $E_{remained}$，則 $T_{energy}=\frac{P_{\max}}{E_{remained}}$

(三) Server 的選舉演算法

PVGI 提供了 server 的選舉演算法，將網路節點分成 *server* 與 *client*，目的是讓節點有不同的身份區別以便賦予不同的工作與責任：由 server 負責調度 grid 內各 client，而 client 則依照 server 指示去完成 server 或自己所需求的工作。假定每一個節點初始時身份是 client。當節點剛加入一個新的 grid 區域時，必須先傾聽是否有自 server 發送的 *Hello* 封包。只要有節點在傾聽一段時間後仍未收到任何 *Hello* 封包，會認為目前 server 不存在，該節點便會立即發起 server 的選舉，以廣播的方式送出 *Bid* 封包，以使所有 grid 內的節點皆能加入競爭行列。

我們希望 server 是整個 grid 區域內狀況最佳的節點，因此當節點收到別的節點傳來 *Bid* 封包後，便會比較 *Bid* 封包的 T_{active} 欄位值。若某一節點本身的 T_{active} 值較小，則該節點即失去角逐 server 的資格，故不再理會任何傳來的 *Bid* 封包；反之，則該節點也開始廣播它的 *Bid* 封包來角逐 server。選舉到最後，只要節點在發送 *Bid* 封包後，在經過一段期間後沒有收到任何 *Bid* 封包，則該節點即主動把自己的身份轉成 server，此時選舉便告完成。

當選舉完成後，server 便開始週期性地廣播 *Hello* 封包，因此 client 可從收到的 *Hello* 封包裡得知目前 server 是誰，以及定期向 server 註冊的週期時間 T_{reg}。而 client 自選舉完成、第一次收到 *Hello* 封包後，即開始定期向 server 發出 *Reg* 封包，以便將節點的現況資訊帶給 server。 為有效掌控各 client 並給予適當的休眠時機，server 必須維護一份表格 GST (Grid Status Table)，以記錄其管轄 grid 區域內所有 client 的現況資訊。因為有了 client 定期發送過來的 *Reg* 封包，所以 server 能隨時更新 GST 資料，以提高 GST 資料的正確性。

(四) 睡眠模式操作

PVGI 定義了 client 的三種狀態，分別是：*active* 狀態表示節點正在傳送或接收資料。*waiting* 狀態表示節點正在傾聽是否有封包，此為所有節點的初始狀態。*sleeping* 狀態表示目前已關閉無線傳輸介面電源，節點不會收到任何封包。所有 client 皆會在這三種狀態間的轉換，其關係圖如圖 3 所示。

每當 grid 新加入了一個的節點，此 client 節點必須先向 server 註冊，並維持在 waiting 狀態。每個 client 會定期檢查是否自己有資料要傳，若有資料要傳時，則會先轉到 waiting 狀態，並發出傳輸要求訊息給 server。當 server

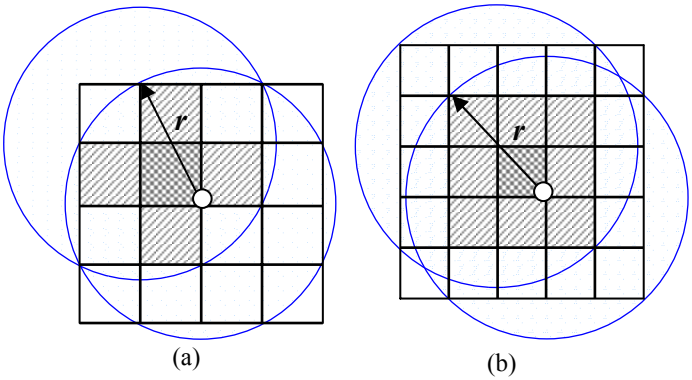

圖 1. Grid 邊長與相鄰 grid 的關係(a) $d=\frac{r}{\sqrt{5}}$，(b) $d=\frac{r}{2\sqrt{2}}$

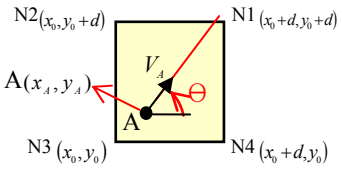

圖 2. grid 之型態

收到傳輸要求訊息，在建立好繞送路徑後，便回覆 client 通訊准許訊息。當 client 收到了通訊准許訊息，便開始傳遞資料封包，並且轉到 active 狀態，如箭頭 C 所示。在 active 狀態的 client，當他完成資料的傳送之後或是本身之 T_{active} 值即將歸零，則會由 active 狀態轉回 waiting 狀態，如箭頭 D 所示。

Server 在下列的狀況發生時，需要有 client 節點來完成所需的工作：(1)當 server 即將從目前的 grid 移開或是本身剩餘電量不足，這時 server 需有一個 client 來接替原 server 的工作。(2)當正在傳輸中的 client（非來源節點或目的節點）即將從目前的 grid 移開或本身電量不足時，這時 server 需有一個可用的 client，來接替此節點的傳輸工作。(3)當 server 收到路徑搜尋封包(*RREQ*)時，若封包裡的繞送目的節點即位於此 grid 區域內，則送出繞送回覆封包(*RREP*)回前一個 grid 之 server。(4)當 server 收到路徑回覆封包，表示繞送路徑將經過此 grid，這時 server 亦需有一個可用的 client 來負責資料傳輸的工作。

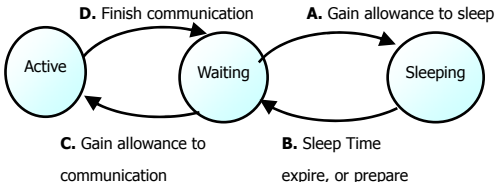

圖 3. client 狀態轉換

只要 server 這裡沒有上述之工作需求時，在 waiting 狀態的 client 節點皆可向 server 提出休眠需求。只要 client 收到休眠准許訊息，即可轉到 sleeping 狀態，如箭頭 A 所示。在 sleeping 狀態的 client，其 GPS 裝置仍保持開啟狀態，並且要週期地取得並更新目前節點的位址資訊，以便在該 client 移出目前所在的 grid 區域時，可以立即醒來。每個 client 會有一個適當的休眠時間，在此休眠時間終了後，便會喚醒自己，由 sleeping 狀態轉回 waiting 狀態。

此外，若(1)client 本身有資料要傳送，(2)client 下一次應該註冊的時間到了，或是(3)client 移到另一個 grid 區域，此時在 client 也必須從 sleeping 狀態立即醒來。如箭頭 B 所示。

(五) 節點的休眠時間長度

在 PVGI 中，使用了休眠時間劃分的概念[3]。已知 client 每經過週期時間 T_{reg} 就必須向其 server 註冊，且同一 grid 區域內的 client 具有相同的週期時間 T_{reg}。我們定義 T_{reg} 由 m 個長度相同的時槽所組成，而每個 client 在此 m 個時槽中最多有 k 個時槽長度的休眠時間。換言之，k 為註冊的週期時間內各 client 休眠時間的上限值，其中 k 應介於 0～m 之間。每當 client 向 server 註冊後，該 client 即可重新配置 k 個時槽的休眠時間。只要 client 本身沒有通訊需求或 server 這裡沒有工作需求時，client 便可允許進入休眠狀態，並在經過一個時槽的時間後醒來。但若後來 client 本身有通訊需求或 server 有工作需求，則當 client 醒來後，必須要等到工作完成後才可再次進入休眠狀態。

為避免因在同一時間有過多的節點處於休眠狀態，或是因 grid 內的節點個數突然變少，導致 server 找不到可用的節點，我們允許 k 值可依照當時 grid 內部節點數量來調整。若目前 grid 內有較多的 client，則 k 值可以提高。反之，若 client 數量過少，則 k 值應小於 $\frac{m}{2}$。圖 4 為上述的關係圖（假設 $m=10$、$k=4$）。

(六) Server 換手(Server-handoff)

在 server 由目前的 grid 移出前或耗完其電量前，必須執行換手來交接 server 的工作。為減低選舉次數，只要在目前的 grid 區域內能找到可用的 client，就不發起 Server 的選舉。因此 server 可由 GST 裡排名最佳的節點開始依序向各 client 詢問，直到找到尚在 waiting 狀

態的 client。之後，原來的 server 發出 *Handoff* 封包給新的 server，目的是要傳遞 GST 給新的 server，使其可依照 GST 資訊接管原 server 的工作。接下來，新的 server 也開始週期地廣播他的 *Hello* 封包，而 client 在收到 *Hello* 封包後，可判知 server 已經換人，此時應主動向此新的 server 註冊。

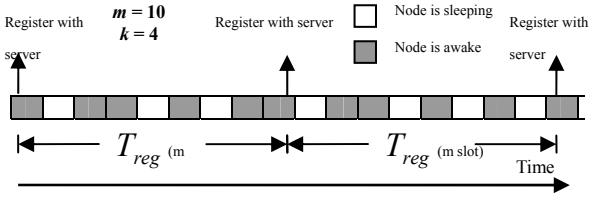

圖 4. 註冊間隔時間與休眠時間關係圖

四、PVGI 的繞送演算法

PVGI 與 GRID 同是 grid-based 架構，因此其繞送封包是以 grid-by-grid 的型式來傳送。路徑搜尋工作是由各 grid 區域裡的 server 負責，亦即由 server 來發出 RREQ 封包，以廣播的方式往相鄰的 grid 傳送。圖 5 為 *RREQ* 封包的傳遞過程中。欲建立 X →Y 的繞送路徑，由 S 發出的 *RREQ*，最後傳遞到達 D。當 D 收到 *RREQ* 後，由於 D 知道目的端 Y 在其管轄之 grid 內，故可由 D 來發送 *RREP*。在 PVGI 中，除了來源端與目的端外，基本上路徑中各 server 並不負責傳輸資料的工作，而是交由各 grid 區域裡的 client 來負責。因此在回覆 *RREP* 封包的同時，server 必須通知負責傳輸資料的 client，讓他知道其下一個繞送節點是誰。

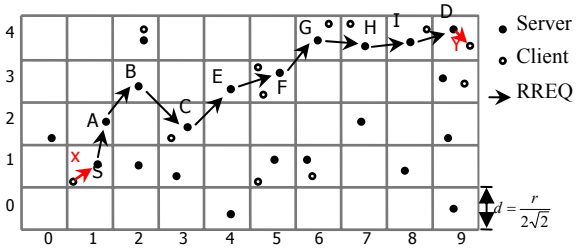

圖 5. 路徑搜尋（節點 X 到節點 Y）

我們以圖 6 來說明：D 先將目的端 Y 的 host ID 附加在 *RREP* 後，才傳送給 I。當 I 收到 *RREP* 時，假設找到 I9 當作繞送路徑的中間節點，則將 I9 的 host ID 附加在 *RREP* 後，再往前送到 H，同時 I 也將收到的 RREP 裡的 host ID 告知 I9。當 *RREP* 傳到 E 時，因為 E 所在的區域裡並沒有 client，故 E 將自己的 host ID 附加在 *RREP* 後再往前傳。最後 *RREP* 傳回到 S，此時我們得到了 X→Y 與 S→D 兩條路徑。前者為由負責繞送資料封包的節點組成；而後者則純粹由 server 組成，將作為維護 X→Y 的繞送工作之用。

在維護繞送路徑方面，可分成下列兩種狀況做討論：(1) 繞送路徑的中間節點移出目前所在的 grid 或電量不足：此節點應立即發出訊息告知 server，此時 server 必須在 grid 區域內另外選出一個可用的節點來替代。以圖 7 為例，假設 F 將離開 grid，因為 F 目前負責兩條繞送路徑，此時 F 會發出訊息告知其 B，因此 B 在選出替代節點 H 後即通知給 F，讓 F 把自己目前的繞送表傳給 H，並且將此變更訊息分別發給 E 與 D。因此 E 與 D 可立即把資料封包改由 H 來傳送。換言之，由 E→I 及 F→G 的繞送工作可以維持正常不中斷。(2)繞送路徑的來源端或目的端移動到相鄰的 grid：與[2]的作法類似，亦即來源端或目的端將後來所在的 grid 之 grid ID 告知前一個 grid 的 server，讓他可以依照收到的 grid ID 之狀況適時調整繞送路徑，若需要時可以在此末端的連結中間插入一個節點幫助繞送。

來源端 S 移動到圖 8(a)中虛線箭頭所指的 5 個相鄰 grid，則原 grid 的 server 可以選出一個 client 節點 E 插入原 S→A 這段連結中，而成為 S→E→A。若是移到圖 8(b)中的另外 3 個相鄰 grid，則不需要對此繞送路徑做任何調整。因此只要原來 S 所在的 grid 有一個以上的節點存在，將可保證繞送路徑維持而不中斷。由此可知，server 不會當閘道節點，實際資料的繞送仍是由各 grid 區域內的 client 負責，server 僅負責管理 grid 內部的節點調度以及路徑搜尋工作。除非 grid 內只存在一個節點時，此時 server 才須暫時擔任資料繞送的工作，直到此 grid 有新節點加入為止，如此將可避免因承載較多的繞送工作而造成較多的電量損耗。另外在路徑的維護上，PVGI 提供的維護路徑方式，盡可能避免路徑中斷毀壞，以提高路徑的穩定性。

五、模擬與分析

(一) 模擬環境參數

本文之模擬環境是在一個 1000m x 1000m 的網路區域，每一個節點之最大傳輸半徑為 250m，資料傳輸率為 2Mbps。通訊需求與流量方面，每秒鐘產生一個繞送需求，其資料流量為每秒產生 20 個封包，每個封包大小為 512bytes。節點的移動方式將依據 *Random way-point* 模型，其移動速度介於 10 m/s ~ 15 m/s，Pause time 設為 10 秒。在電量參數上，本文採用[8]的耗電功率參數，即 Transmit: 1.4W, Receive: 1.0W, Listen: 0.83W, Sleep: 0.043W, GPS: 0.165W。另外假定節點每秒利用 GPS 更新一次目前位置資訊，這項工作亦會消耗少許電量。

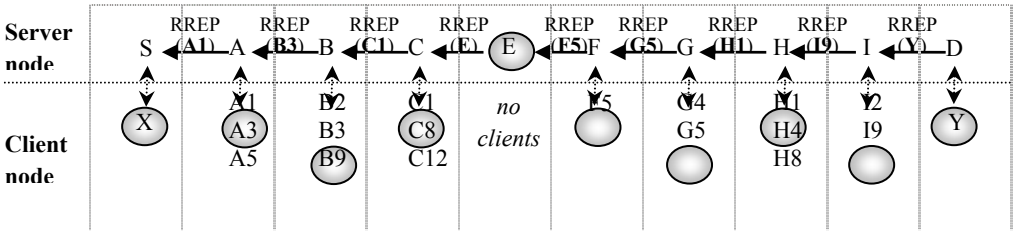

圖 6. 路徑搜尋結果（節點 X 到節點 Y）

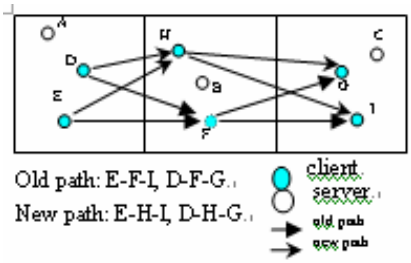

圖 7. 中間節點 F 離開 grid

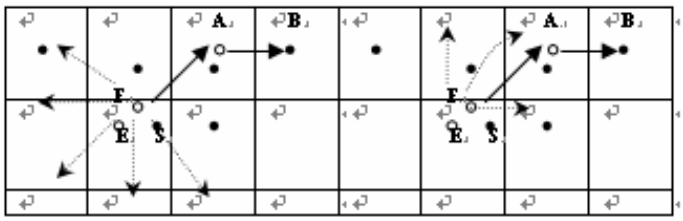

圖 8. 路徑維護（來源端 S 移動到相鄰的 grid）

(二) 模擬結果

- 休眠時間長短對 PVGI 效能的影響

為比較休眠時間長短對 PVGI 效能的影響。這裡將 T_{reg} 分成 100 個時槽，調整每個節點在此 100 個時槽中最多可以休眠的時槽個數(即 k 值)。首先，我們比較休眠時間長短對存活節點個數的影響，如圖 9。假設節點數目為 100，存活節點比例為存活節點個數與網路節點總數的比值，k 分別取 0、10、20、50、與 100。當 k 值為 0 時，由於節點並沒有給予任何休眠時間，因此在 400 秒時，所有節點電量都將耗盡。而當 k 值為 10 時，雖然給予的休眠時間仍短，但所有節點將到 700 秒後才會耗盡電量。當 k 持續加大，可存活的節點將會越來越多。這是因為每個節點能夠最多可以有 k 個時槽的休眠時間，k 值越大表示節點可以休眠的時間也越長。但當 k 值超過 50 後，其差異已不明顯。這是因為當節點本身有通訊需求或 server 指派了工作，則此節點必須等到工作完成後才可在再次進入休眠狀態。此外，不論節點是否睡了 k 個時槽的時間，只要一到下一次註冊時間節點救一定要醒來。因此 k 值並不需設得太大，取適當的 k 值即可達到省電的效益。接著我們比較節點平均電量消耗功率與 k 值大小的關係。平均電量消耗功率指單位時間內，節點消耗的電量大小。這裡我們只取 k 介於 0～50 來比較，因為在圖 9 中，當 k 超過 50 時，其省電的程度已不顯著。

圖 10 兩曲線分別代表 grid 邊長取 進入休眠狀態，因此 PVGI-L 將能夠節省較多電量的損耗。

- 關於 grid 維護之效能比較(PVGI 與 GRID)

因為維護 grid 需要靠各種控制封包傳遞訊息來達成，因此我們必須觀察 grid 內各種控制封包發送的數量與比例關係。這裡的 grid 控制封包指的是 *Hello*、*Reg*、*Handoff* 及 *Bid* 等封包。圖 11 為 500 秒的模擬時間內，所有 grid 內產生之控制封包總數與節點個數的關係圖。當節點個數由 100 增加到 300，PVGI 與 GRID 的控制封包皆會增加，此現象應是為維持 grid 正常運作，而產生了較多的控制封包所致。我們觀察 PVGI 在控制封包總數上明顯低於 GRID，這是因為 PVGI 在處理 server 換手時，會先尋找一個替代節點。只要找得到這樣的節點，就不需發起 Server 的選舉，因此可以減少 *Bid* 封包的數量。由於 *Bid* 封包減少的數量遠大於其他非 *Bid* 的控制封包增加的數量，因此 PVGI 在控制封包總數上比 GRID 來的低。

接著，我們比較 GRID 與 PVGI 在 Server 選舉上所花費的成本差異。這裡我們定義 Server 選舉的成本為：所有在 grid 內節點發出之「*Bid* 封包總數」除以所有在 grid 內節點發出之「控制封包總數」。圖 12 為 PVGI 與 GRID 在 Server 選舉成本的比較圖。在圖中，PVGI 的選舉成本幾乎維持在 9%左右，而 GRID 則由 72%開始往上增加到 84%，這是因為當 server 節點需要換手時，GRID 總是要啟動 Server 的選舉，而 PVGI-L 只有在未找到替代

節點時才需要啟動 Server 的選舉，因此 PVGI-L 花費在選舉的成本極小。降低花費在 Server 選舉的成本，其意義是：當有任何繞送需求通過此 grid 時，grid 區域內的節點不太需要付出 overhead 在傳收 Bid 封包上，因此可以將頻寬用在傳送非控制封包的資料上。

六、結論

藉由定位裝置（GPS）的輔助，我們可以獲得節點的位置、移動速度、移動方向與同步時間等資訊。這些資訊除了可用來建置網格架構外，還可作為網路節點的移動性參數。本文將節點的移動性與節點的剩餘電量資訊轉換成時間的形式，來表達節點的現況資訊，也因此能夠較準確地選出當時最佳的節點作為 grid 區域裡的管理者。節省電量消耗可以藉由操作休眠模式達成，本文提出之節點狀態轉換與調節休眠時間長度的方法，來讓節省電量與效能之間的取得一個平衡。在我們的模擬結果顯示 PVGI 在取適當的 k 值時，可以節省電量消耗。

本文調整 server 的工作角色，即 server 不當閘到節點，所有資料封包皆直接由 grid 內的 client 節點負責繞送，以減輕 server 的負擔，且能提高繞送路徑穩定性。另外，為減少 server 選舉成本的產生，當 server 節點本身要離開 grid 或電量不足時，即由選出的 client 節點來接替。最後的模擬結果顯示：PVGI 可明顯降低與維護 grid 相關的控制封包數量及 server 選舉成本，因此能獲得效能上的改善。

在未來的研究方向，我們希望以需求導向(on-demand)的方式來維護 grid，減低原先需要定期發送的控制封包(如 Hello 與 Reg)。此外，關於節點休眠時槽的安排上，需要一種較為深入的排程演算法，在不影響效能的情況下讓省電能達到最佳化。

七、參考文獻

[1] B. Chen, K. Jamieson, H. Balakrshnan, and R. Morris, “Span: an energy efficient coordination algorithm for topology maintenance in ad hoc wireless networks,” in *ACM MobiCom*, pp. 85-96, July 2001.

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

[3] S. PalChaudhuri and D.B. Johnson, “Power mode scheduling for ad hoc networks,” in *Proceedings of 10th IEEE International Conference on Network Protocols*, pp.12-15, Nov. 2002.

[4] M. R. Pearlman, J. Deng, B. Liang, and Z. J. Haas, “Elective Participation in Ad Hoc Networks Based on Energy Consumption,” in *Proceedings of IEEE Global Telecommunications Conference/Symposium*, Nov. 2002.

[5] C. Srisathapornphat, and C.-C Shen, “Coordinated Power Conservation for Ad hoc Networks,” in *Proceedings of IEEE International Conference on Communications*, pp. 3330-3335, Apr. 2002.

[6] Y.-C. Tseng and T.-Y. Hsieh, “Fully power-aware and location-aware protocols for wireless multi-hop ad hoc networks,” in *Proceedings of 11th IEEE International Conference on Computer Communications and Networks,* pp. 608-613, Oct. 2002.

[7] Y. Xu, J. Heidemann, and D. Estrin, “Geography-informed Energy Conservation for Ad Hoc Routing,” in *ACM MobiCom*, pp. 70–84, July 2001.

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

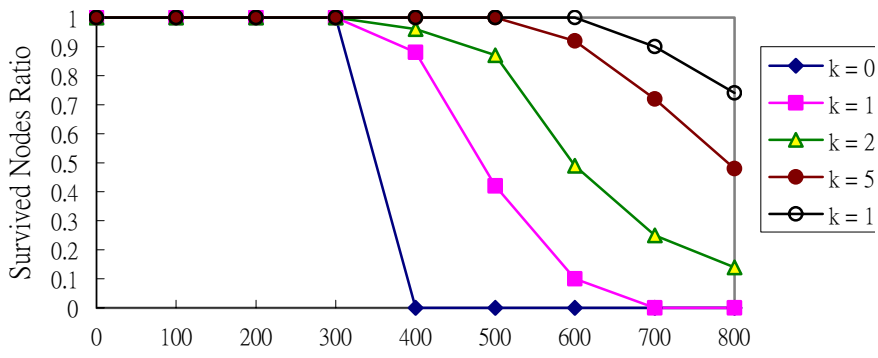

圖 9. k 值與存活節點比例的關係（節點數 100，速度 10~15m/s）

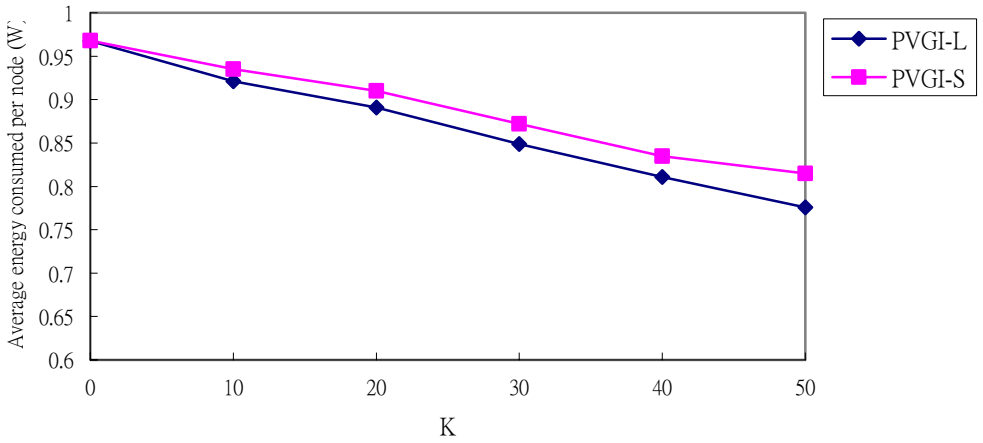

圖 10. k 值與節點平均電量消耗功率的關係
（節點數 100，速度 10~15m/s，模擬時間 500s）

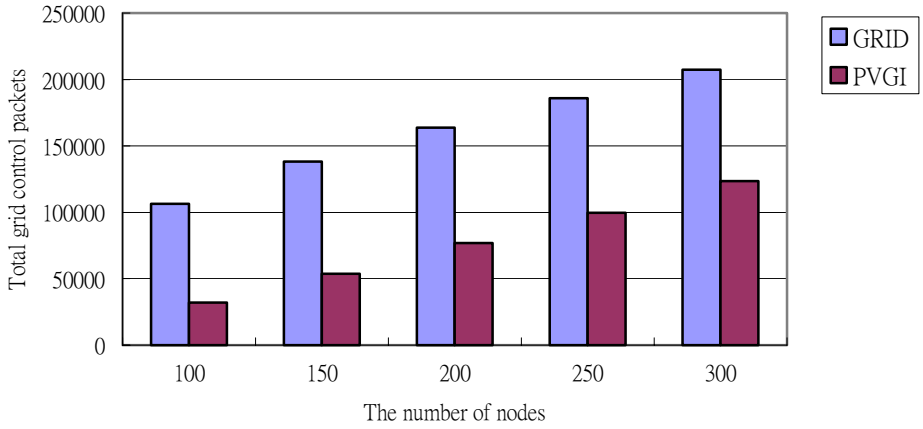

圖 11. Grid 維護之控制封包總數與節點個數的關係
（節點數 100，速度 10~15m/s，模擬時間 500s）

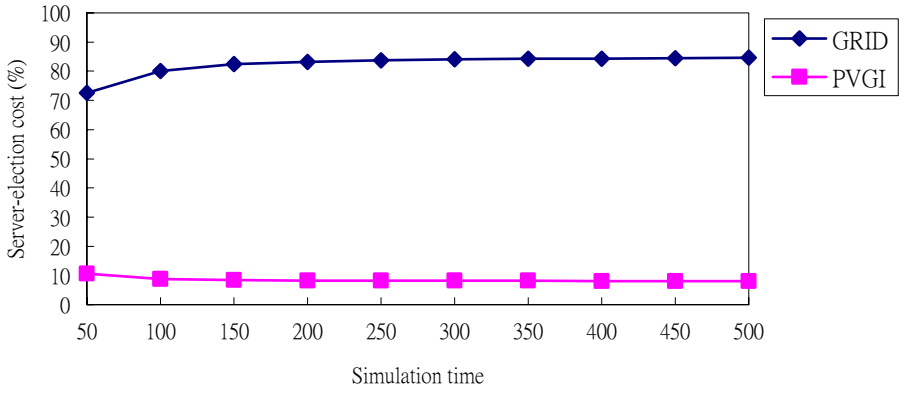

圖 12. Server 選舉成本比較圖（節點數 100，速度 10~15m/s）