行政院原子能委員會核能研究所

委託研究計畫研究報告

微電網代理人基控制與通訊協定的研究

A Research on Agent Control and Communication in Micro Grid

計畫編號：**1002001INER066**

受委託機關(構)：國立清華大學

計畫主持人：金仲達

聯絡電話：**03-574-2804**

E-mail address：king@cs.nthu.edu.tw

核研所聯絡人員：詹振旻

報告日期：　**100** 年 **12**　月　**6** 日

目　錄

中文摘要

微電網藉由資訊科技幫助，使得電力的傳輸更加有效率且穩定。其中一項關鍵技術是分散式的智慧型代理人，這些智慧代理人在微電網中不同的數位電子裝置，如 IED (intelligent electronic devices) 上執行，搜集相關的資料並分析。為使整體電力系統能穩定運作，分散的智慧型代理人所收集的資料以及其協同之運作，必須有一致而且準確的時間標記。這就需要執行這些智慧代理人的 IED 將其系統時鐘調校一致 。為解決此問題，一個較經濟的方法，是將部份的智慧型代理人執行節點上裝設 GPS 以取得精確時間，並擔任參考點。其他節點則藉由已存在的通訊網路向參考點取得時間並進行同步。問題是在電力網路中，若存有一群參考點，如何判斷那些智慧型代理人執行節點能被同步? 若無法被同步，如何加入最少的參考點以滿足其需求？本研究提出一以成本為考量之方法以解決這些問題。

Abstract

A key component in the micro grid is the distributed intelligent agents, which are executed on the intelligent electronic devices (IEDs) deployed throughout the Micro-grid. To maintain proper and robust operations of the power system, the data acquired by the distributed intelligent agents and their coordinated operations must be accurate under a common time reference. This in turn demands that the clocks of the underlying IEDs be synchronized to the required precision. A cost-effective approach is to equip some IEDs with the Global Positioning System (GPS) as the time references, and leverage the communication links already existing between the IEDs to synchronize the clocks of those without GPS. The question is, given a set of such reference nodes, how to decide whether the clocks of all the IEDs can be synchronized to the required precision. If not, how can one add fewest reference nodes to meet the requirements? This report addresses these issues and proposes efficient algorithms to solve the problems.

壹、計畫緣起與目的

微電網是一複雜且分散之系統，結合資訊科技使電力的傳輸更有效率、穩定且安全。在微電網中，分散式的智慧型代理人扮演非常重要的角色。透過分散的智慧型代理人，微電網可以對電力系統進行監視、控制、分析及預測。智慧型代理人由分佈在微電網中的智慧型電子裝置(intelligent electronic devices, IED)所執行，可收集相關的電力資料，再透過已存在的通訊網路互相連結、交換資訊，共同合作以維持電力系統的穩健運作。所搜集及交換的資料以及相關的控制運作，必須具有一定的時間精確度，方可協助電網穩定的運作。這就需要執行智慧代理人的 IED 裝置上的時鐘，時間必需被同步到一定之精確度才能滿足其需求。然而在電網中，不同的應用有不同時間精確度之需求，因此如何同時滿足不同智慧型代理人對時間同步之需求，是一非常重要的議題。

為滿足上述需求，最簡單的方式是將每個執行智慧型代理人的 IED 上都設置一精確校時裝置 (如 GPS 接收器)。但 GPS 的精準度會受到氣候及環境的影響，而且成本不低，因此這種作法不適合大規模佈建。另一方面，由於 IED 間大多已有通訊網路，並透過通訊網路交換資訊，因此可將已裝設有 GPS 接收器之節點做為時間同步之參考點，而其他的節點都藉由通訊網路與參考點進行時間同步。在過去的研究中，已針對通訊網路之時間同步有相當程度的討論，其成果可被應用於本研究之中。

本研究著重的問題在於電力網路中，給定一群參考點，如何判斷其他節點是否能夠達成時間同步？若無法達成同步時，如何加設最少的參考點，達成時間同步之需求？

貳、研究方法與過程

根據微電網中對於時間同步之不同需求及成本的考量，本章將會分成三個部份來討論。第一個部份是說明電力網路之時間同步模型及問題的定義。第二部份討論如何確認整個電力網路是否能達成所需之時間精準度。第三部份討論若電力網路無法達成時間同步時，如何佈建最少參考點以滿足時間同步之需求。

一、網路模型

為解決電力網路中時間同步之問題，可將整個電力網路用一個圖型 (graph) $G(V, E)$ 來描述。V 代表 graph 中所有節點的組合，E 則代表 graph 中所有點與點之間的連線組合。在 graph 中的每個節點 (node) 代表一個 IED，而點與點間的連線 (link) 則代表 IED 與 IED 間通訊網路之連線。在此 graph 中，部份的節點是直接連接 GPS 以校準時間，且這些節點被稱之為參考點 (reference node)，而其他的節點則被稱為一般節點 (normal node)。另外，透過網路進行時間同步時，每經過一個節點或一段連線都會因處理時間及網路延遲而造成時間上的誤差。一般根據時間同步模型大多會利用二點間的傳輸時間的平均值作為其時間誤差值。

為確認該節點是否達到所需的時間同步，可藉由傳輸線材標準，取得最大、最小之值，再以平均值補償後推算出可能的最大誤差時間。若最大可能的誤差大於可容許的範圍內則判定為未達所需之時間同步。因此，我們可以利用下列三個參數來表示節點 v_i 與節點 v_j 間連線(e_{ij})之最大、最小、及平均延遲時間: $LMax(e_{ij})$、 $LMin(e_{ij})$ 、及 $LPre(e_{ij})$，而經過此連線(e_{ij})可能的最

大誤差如下公式：

$$Deviation(e_{ij}) = max(|LMax(e_{ij})-LPre(e_{ij})|, |LMin(e_{ij})-LPre(e_{ij})|) \quad (1)$$

同樣地，經過節點 v_i 進行時間同步時最大、最小、平均處理時間之參數為： $DMax(v_i)$、$DMin(v_i)$、及 $DPre(v_i)$，而經過節點 v_i 可能的最大誤差為：

$$Deviation(v_i) = max(|LMax(v_i)-LPre(v_i)|, |LMin(v_i)-LPre(v_i)|) \quad (2)$$

當一般節點從參考點收到時間同步之訊息後，需計算此時間可能的最大誤差值，而此誤差值必需包含通訊造成的延遲時間及經過節點之處理時間等二項可能的最大誤差值。因此，從參考點 r_j 到一般節點 v_i 經過 $path_k$ 的最大可能的誤差值：

$$\begin{aligned} &Deviation(Path_k(r_j \rightarrow v_i)) \\ &= max(|TLMax(Path_k(r_j \rightarrow v_i)) \quad TLPre(Path_k(r_j \rightarrow v_i))| \\ &\quad + |TDMax(Path_k(r_j \rightarrow v_i)) \quad TDPre(Path_k(r_j \rightarrow v_i))|, \\ &\quad |TLMin(Path_k(r_j \rightarrow v_i)) \quad TLPre(Path_k(r_j \rightarrow v_i))| \\ &\quad +|TDMin(Path_k(r_j \rightarrow v_i)) \quad TDPre(Path_k(r_j \rightarrow v_i))|) \end{aligned} \quad (3)$$

where

$TLMax(Pathk(rj \rightarrow vi)) = \Sigma(LMax(eab))$

$TLPre(Pathk(rj \rightarrow vi)) = \Sigma(LPre(eab))$

$TLMin(Pathk(rj \rightarrow vi)) = \Sigma(LMin(eab)) \quad \forall e_{ab} \in E_{pa}$

$TDMax(Pathk(rj \rightarrow vi)) = \Sigma(DMax(va))$

$TDPre(Pathk(rj \rightarrow vi)) = \Sigma(DPre(va))$

$TDMin(Pathk(rj \rightarrow vi)) = \Sigma(DMin(v_a))$

$$\forall v_a \in V_{Path}$$

$$\forall v_a \in V_{Path}$$

$$\forall v_a \in V_{Path}$$

上述公式中，$TLMax(Path_k(r_j \rightarrow v_i))$、$TLPre(Path_k(r_j \rightarrow v_i))$、及 $TLMin(Path_k(r_j \rightarrow v_i))$表示從節點 r_j 到節點 v_i 經過 $path_k$ 時所產生的最大、平均、最小的傳輸時間。同樣地，$TLMin(Path_k(r_j \rightarrow v_i))$、$TLPre(Path_k(r_j \rightarrow v_i))|$、及$|TDMin(Path_k(r_j \rightarrow v_i))$表示從節點 r_j 到節點 v_i 經過 $path_k$ 時所產生的最大、平均、最小的處理時間。所有的 e_{ab}、v_a 都是屬於 $path_k$。

二、確認電力網路是否達成時間同步

在電力網路中，假設已有一群節點(R)因直接連接精確校時設備而當作參考點。為確認整體電力網路是否能因而達到時間同步，最直接的方式是將每個一般節點到所有參考點的所有路線都檢查一次。但此種方式非常耗時。本研究因此提出 SPT-Threshold 及 SPT-Connection 二種演算法來解決此一問題：

(一)SPT-Threshold

是利用最短路徑樹(The Shortest Path Tree)來確認各節點是否能達成時間同步的要求。其演算法如圖一所示，主要分成

二個部份，第一個部份(8~24 行)，由每個參考點建立一最短路徑樹，而為了避免不必要的計算，SPT-Threshold 在建立最短路徑樹的過程中，若遇到其他參考點或累積誤差已達所有節點可容忍誤差之最大值時，將會停止建此一路徑。

在第二部份(27~40 行)，每個一般節點再根據連結到它的所有最短路徑樹，選擇一個能讓其時鐘達成時間同步之參考點，並確認是否滿足其所需之時間同步，計算的方法則根據公式(3)計算。

SPT-Threshold

```
1.  R = the set of reference nodes in G
2.  Deviation(Path(r_i→v_u)) = the accumulative deviation of the shortest path
      from r_i to v_u
3.  Q_i=the set of nodes in the shortest path tree rooted at r_i
4.  T_u= the allowable deviation at node v_u
5.  TM = max{T_u} for all normal nodes v_u in G
6.  flag_u= whether node v_u can be synchronized to the required precision
7.  Net_Not_Sync=whether G can be synchronized to the require precision
8.  for each reference node r_i in R
9.    Q_i = {r_i}
10.   do
11.     Min =  ∞
12.     for each neighbor v_u of each node in Q_i
13.         if (v_u is not in R and not in Q_i)
14.           if    (Deviation(Path(r_i→v_u))<Min)
15.                 Min = Deviation(Path(r_i→v_u))
16.                 Candid = v_u
17.           end if
18.         end if
19.     end for
20.     if    (Deviation(Path(r_i→Candid))<TM)
21.        insert Candid into Q_i
22.     end if
23.   while (Deviation(Path(r_i→Candid))<TM)
24. end for
25. return net_unsynched();
26.
27. net_unsynched()
28.     Net_Not_Sync = false
29.     for each node v_u not in R
30.
```

```
31.         for each Q_i
32.           if (v_u ∈ Q_i && Deviation (Path(r_i →v_u ))<T_u)
33.                flag_u=true
34.                break
35.           end if
36.         end for
37.         if (flag_u==false)
38.              Net_Not_Sync = true
39.         end if
40.      end for
       return Net_Not_Sync
```

圖一、SPT-Threshold 之演算法

(二)SPT-Connection

SPT-Connection 與 SPT-Threshold 類似，同樣是以最短路徑樹驗證電力網路是否滿足其時間同步之需求。其不同點在於 SPT-Connection(如圖二) 考量在某些情況下，並不需要為所有的參考點建最短路徑樹，只需要部份的參考點即可滿足所有節點的需求，因此考量參考點的 degree 作為建 tree 的先後順序，及累積的時間誤差作為停止建 tree 之門檻。

SPT-Connection 的演算法同樣分成二個部份，在第一部份先找出最大 degree 的參考點，以此參考點建立最短路徑樹，建完後確認是否有節點尚未被滿足，及是否還有參考點仍未被建立最短路徑樹(如第 25 行)。若否，則再找次高 degree 的參考點，再建此參考點之最短路徑樹，直到所有的點都被滿足，或是參考點都已建立其最短路徑樹。而在建 tree 的過程中，遇到其他參考點不會停止建 tree，但當從參考點累積的可能最大誤差值已超過所有節點可容忍之最大值，才會停止建路徑。第二部份，則與 SPT-Threshold 相同。

SPT-Connection

1. RN = the set of reference nodes in G that the corresponding shortest path tree is not yet built
2. V = the set of normal nodes in G
3. $Deviation\ (Path(r_i \rightarrow v_u))$ = the accumulated deviation of the shortest path from r_i to v_u
4. Q_i=the set of nodes in the shortest path tree rooted at r_i
5. TM = the maximum allowable deviation
6. Out_i= the degree of reference node r_i in G
7. do
8. find a node r_i in RN that has the maximum Out_i
9. remove r_i from RN
10. $Q_i = \{r_i\}$
11. do
12. $Min = \infty$
13. for each neighbor v_u in V of each node in Q_i
14. if (v_u is not in Q_i)
15. if ($Deviation(Path(r_i \rightarrow v_u)) < Min$)
16. $Min = Deviation(Path(r_i \rightarrow v_u))$
17. $Candid = v_u$

18. end if
19. end if
20. end for
21. if (*Deviation*(*Path*(r_i→*Candid*))<*TM*)
22. insert *Candid* into Q_i
23. end if
24. while (*Deviation*(*Path*(r_i→*Candid*))<*TM*)
25. while(**net_unsynched()** == true && *RN* is not empty)
26. return **net_unsynched**();

圖二、SPT-Connection 之演算法

三、最少參考點之佈建

當現有的參考點無法滿足所有節點的時間同步需求時，必需增設參考點以滿足其時間同步需求。最簡單的方式是將每個不被滿足之節點都裝設精準的校時設備(如 GPS 接收器)。但此種方法可能導致成本過大之問題，為此本研究提出 CP Deployment、CP and Cost Deployment 及 Degree Deployment 等三種演算法：

(一)CP Deployment

CP Deployment 主要考量是選擇可縮減網路中所有節點時間誤差總合之最大者，也就是當要新增一個參考點時，先計算此新參考點可縮短網路中所有節點誤差之總合，選擇一個可縮短最多誤差時間之節點來增設參考點。若是仍有節點不被滿足時，則再以此方式新增一個參考點，直到所有的節點都被滿足為止。其演算法如圖三所示。

(二)CP and Cost Deployment

CP and Cost Deployment 與 CP Deployment 相似，其不同點在於考慮的對象不同，CP and Cost Deployment 考慮是所有

未被同步之節點可縮減之誤差總合，而非網路中所有的節點，也就是說 CP and Cost Deployment 在選擇一個新的參考點時，計算可縮減的誤差是針對未被同步之節點。

CP and Cost Deployment 與 CP Deployment 有個最大的缺點是需要為每一個可能的參考點都建立一完整的最短路徑樹，並在新增參考點時需逐一確認每個可能參考點的縮減程度，其計算量非常龐大且複雜，因此提出一啟發式演算法 -Degree Deployment。

CP Deployment

1. R = the set of reference nodes in G
2. V = the set of normal nodes in G
3. CP_i= the amount of reduction in total time deviation when node i becomes a reference node
4. *Cand_ref*=the candidate of reference node
5. for each reference node j in R
6. build the shortest path tree Q_j
7. end for
8. while (**net_unsynched()** ==true)
9. Max = 0
10. for each node i in V
11. calculate CP_i
12. if ($CP_i > Max$)
13. $Max = CP_i$
14. *Cand_ref* = i
15. end if
16. end for
17. remove *Cand_ref* from V
18. add *Cand_ref* to R

19. for each reference node j in R
20. build the shortest path tree Q_j
21. end for
22. end while

圖三、CP Deployment 之演算法

(三)Degree Deployment

Degree Deployment 與上述方法(CP and Cost Deployment 與 CP Deployment)最大不同點在於，不需對每個參考點建立完整之最短路徑樹，只需要利用前一步驟提出之 SPT-Threshold 或 SPT-Connection 取得未被同步之節點即可。

Degree Deployment 考慮每個未被滿足之節點的 degree，來決定可以設置參考點的節點。一般在電力網路中，若是 degree 高的節點大多是重要的節點，其影響力將會更勝於其他的節點。因此在新增參考點時，以節點之 degree 為首要考慮的條件。Degree Deployment 之演算法如圖四所示，先利用先前所提 SPT-Threshold 或 SPT-Connection 方法找出不被滿足之節點，再從這些未被同步的節點中，選擇一個連接最多不被滿足節點的節點，將此節點設為參考點，若仍有未能滿足其時間精準度之節點，則以相同方式新增一參考點，直至所有的節點都被滿足時才停止。

Degree Deployment

1. R = the set of reference nodes in G
2. V = the set of normal nodes in G
3. $degree_j$ = the number of unsynchronized nodes to which a unsynchronized node j in V is connected
4. *Cand_ref*=the candidate of reference node
5. run SPT-Threshold or SPT-Connection
6. while (**net_unsynched()** == true)
7. *Max* = 0
8. for each unsynchronized node j in V
9. if ($degree_j$ > *Max*)
10. *Max* = $degree_j$
11. *Cand_ref*=j
12. end if
13. end for
14. remove *Cand_ref* from V
15. add *Cand_ref* to R
16. run SPT-Threshold or SPT-Connection
17. end while

圖四、Degree Deployment 之演算法

參、主要發現與結論

為驗證所提出方法之有效性，我們在 IEEE 13-NodeTest Feeder、IEEE 34-NodeTest Feeder、及 IEEE37-Node Test Feeder 等三種電力網路上進行實驗，並分成二個部份來討論及分析。在實驗中，將以亂數方式任選二個節點作為參考點，並實驗所有的亂數組合的結果。

為設定每段節點有不同的傳輸時間，我們利用連線的距離線性增加其傳輸的延遲時間，每單位(100 英呎)連線之最大、最小、平均延遲時間設定為 30 ns、20 ns、及 10ns，每增加一單位則線性增加。

一、確認電力網路是否達成時間同步之實驗

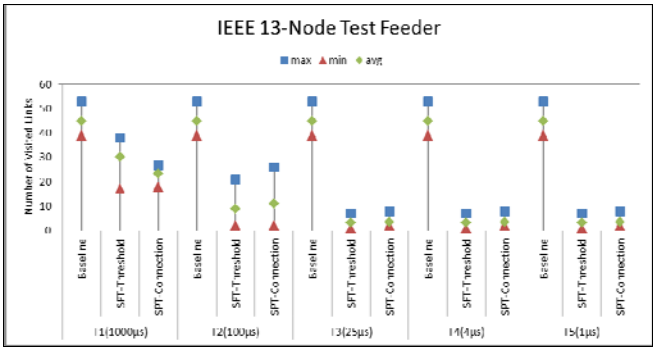
為驗證提出二個方法(SPT-Threshold 及 SPT-Connection)，我們利用 IEC1850 提出五種不同應用類型所需的時間精確度進行實驗(如表一)，並以網路所需檢查連線次數為評斷的依據。

表一、容忍時間誤差的五種類型

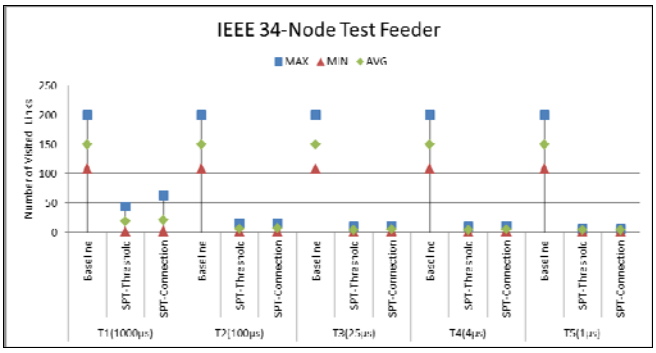
Level of Tolerable Deviation	Time
T1	± 1000μs
T2	± 100μs
T3	± 25μs
T4	± 4μs
T5	± 1μs

我們在實驗中增加 Baseline 方法，與所提的二個方法作為比較對象。Baseline 方法是為所有參考點建立完整最短路徑樹，其實驗結果如圖五~圖七所示。從實驗結果中我們可以發現，所提的二個方法(SPT-Threshold 及 SPT-Connection)能更加有效減少檢查連線的次數。SPT-Threshold 及 SPT-Connection 的結果非常相近，原因在於在建樹的過程中，考慮到若超過所能容忍誤差最大值時，立即停止建樹，因此可有效減少檢查連線的次數。

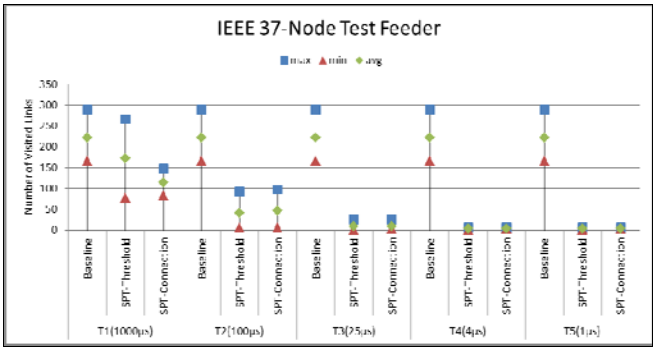
當整個網路容忍的誤差值愈大，且不需所有參考點均建立最短路徑樹時，SPT-Connection 比 SPT-Threshold 所需的檢查次數少(如圖五、圖七)。在 IEEE 34-NodeTest Feeder(如圖六)網路中，因為每段連線的距離較大，造成網路中所有參考點皆需建立其最短路徑樹，以確認所有的一般節點的同步與否，因此無法明顯看出 SPT-Connection 的優點。但也可藉此發現，當網路容忍的誤差變小需要較多參考點時，SPT-Threshold 可比 SPT-Connection 減少更多檢查連線的次數。這是因為 SPT-Threshold 在建樹時，同時考慮最大誤差值及其他參考點，因此可比 SPT-Connection 減少檢查連線的次數。



圖五、IEEE 13-node test feeder 實驗結果



圖六、IEEE 34-node test feeder 實驗結果



圖七、IEEE 37-node test feeder 實驗結果

二、最少參考點佈建之實驗

當現有的參考點無法滿足所有節點所需之時間同步時，需要新增參考點以滿足需求，問題是該如何增加最少參考點以滿足需求？在實驗中，我們以增加參考點的個數為比較的依據。在 IEEE 13-node test feeder 實驗中，新增 Exhaustive Search algorithm 與其他的方法分析比較。Exhaustive Search algorithm 是利用窮舉的方法找出最佳解，可藉由此方法了解提出的方法與最佳解的差距。

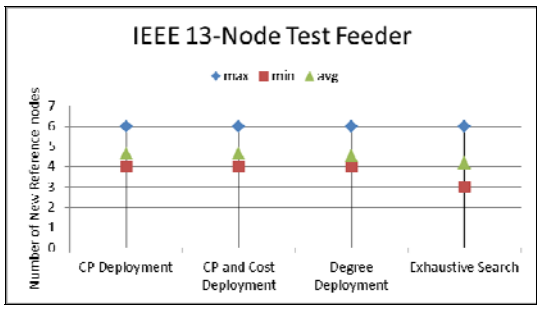
在實驗中我們考慮二種情況。情況一，在網路中每個節點都有相同的誤差容忍度(±100μs)；情況二，考慮每個節點有不同程度的誤差容忍度。在電力網路中，當節點的 degree 較大

時，節點通常扮演較為重要的角色，因此其節點可容忍的誤差較小。因此在實驗中設計當該節點的 degree 較大者(大於 3)，則設定其誤差容忍度為±1μs，其他則為±100μs。

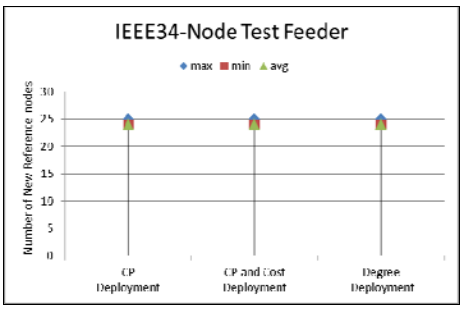
(一) 情況一：所有的節點的誤差容忍度皆為±100μs

其實驗結果(如圖八、(a))中可發現，所提出的三個方法需新增參考點的總數非常接近於最佳解方法(Exhaustive Search algorithm)。由於提出的三種方法在本質上是利用貪婪(greedy)的方式，每當要新增一個參考點時，根據當時的情況選擇，因此在最差的情況時會比最佳解差。基本上，所提出的三種方法的結果會非常相似，但在網路節點數多時，CP Deployment 會比其他二個方法需要新增較多的參考點(如圖八(c))，原因是因為 CP Deployment 是考慮因素是針對所有節點，而非尚未被同步之節點。

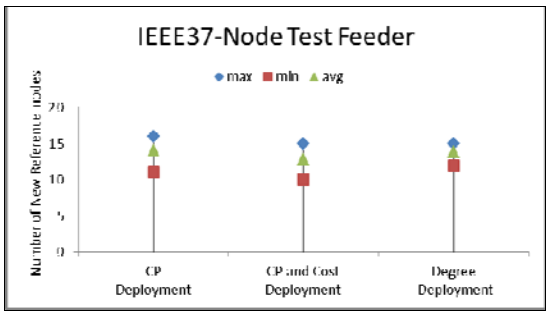
在圖九中，我們討論每個方法的時間複雜度。時間複雜度即是需要檢查連結的個數。於 CP Deployment 及 CP and Cost Deployment 是需要每個參考點都建完整的最短路徑樹，而 Degree Deployment 可利用 SPT-Connection 或 SPT-Threshold 檢查，因此其所需檢查的連線數遠少於 CP Deployment 及 CP and Cost Deployment。此外，也可發現由於實驗的環境皆屬於需要每個參考點都建立其最短路徑樹的情況，因此 SPT-Threshold 比 SPT-Connection 需更少的檢查次數。



(a)

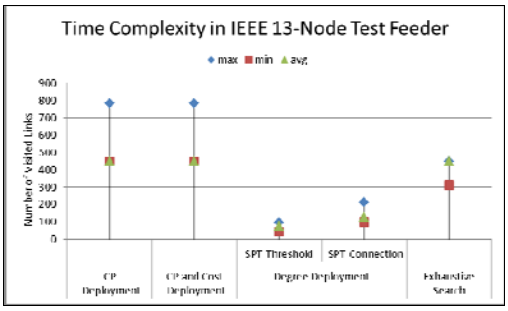


(b)

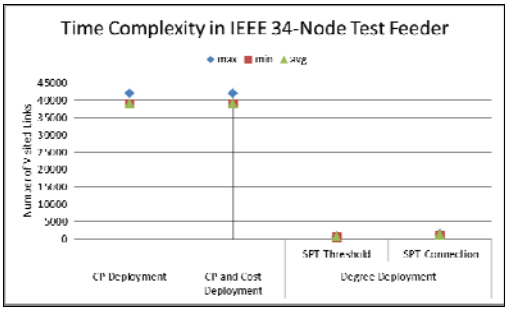


(c)

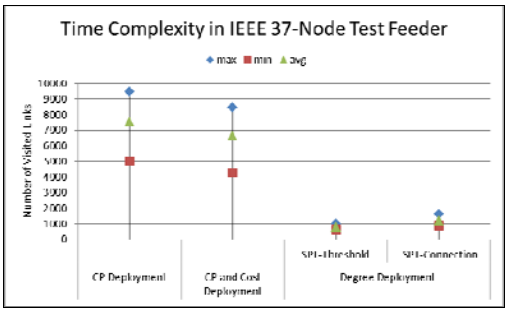
圖八、情況一各網路需新增節點數



(a)



(b)



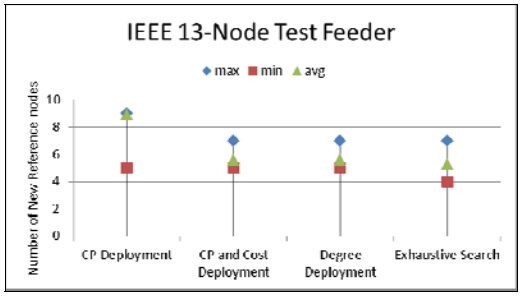
(c)

圖九、情況一各網路所需檢查次數

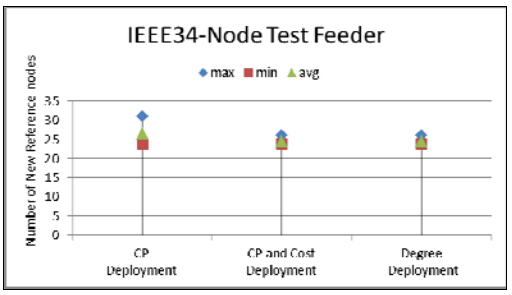
(二)情況二、不同程度的誤差容忍度

情況二中有一些節點是有時間高精準度的需求，有些則否。從實驗結果(如圖十)中，可發現 CP Deployment 比其他二個方法需新增更多的參考點。主要的原因在於 CP Deployment 不針對未達成時間同步之節點，而是考慮整個網路的全部節點。此外，Degree Deployment 略優於 CP and Cost Deployment，因為 Degree Deployment 考慮關鍵未被同步之節點，如考慮最大 degree 未被同步之節點，利用此節點的影響力，減少新增參考點的數量。此外，從實驗結果(如圖十一)中，可發現 Degree

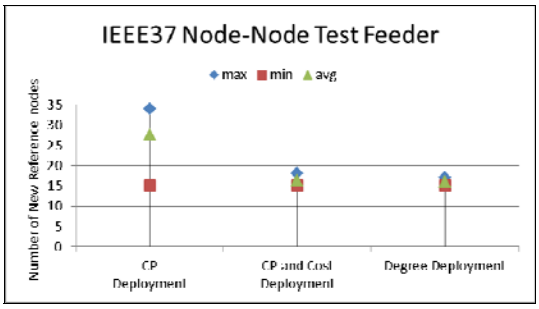
Deployment 利用 SPT-Connection 或 SPT-Threshold 檢查網路，因此比其他二個方法，運用較少的檢查次數，即可完成。



(a)

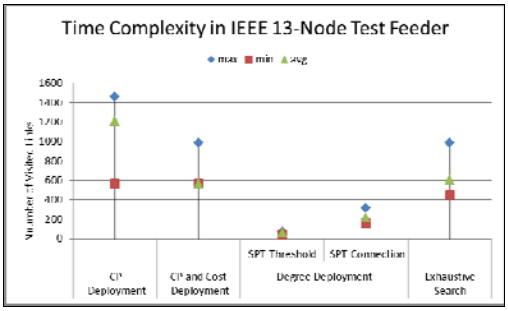


(b)

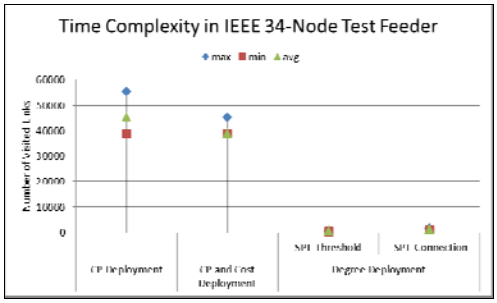


(c)

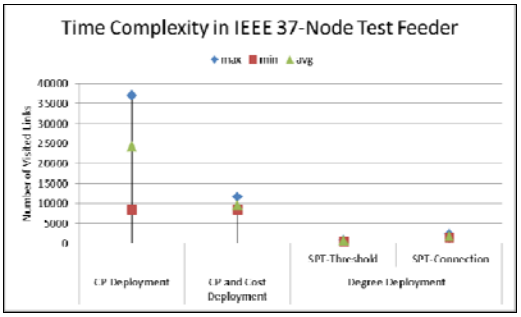
圖十、情況二各網路需新增節點數



(a)



(b)



(c)

圖十一、情況二各網路需檢查連線次數

為維持微電網的運作，並使其運作更加穩健，智慧型代理人扮演一非常重要的角色，但由於智慧型代理人被執行在分散的電子裝置之上，並透過電子裝置所搜集的資料進行分析及預測，而為使其所有的代理人能夠共同合作維持整個微電網的運作，各代理人所搜集交換的資料都必需精確的時間，為此本研究根據此需求提出以最低成本的方法解決，並分成二個步驟解決，而在每個步驟中都提出二種以上不同以成本為考量的方法，經過實驗後亦証明其方法可達成其時間同步需求外，亦能有效降低其成本。

肆、參考文獻

1.V. Vyatkin, G. Zhabelova, N. Higgins, M. Ulieru, K. Schwarz, and N. K. C. Nair, “Standards-enabled Smart Grid for the Future EnergyWeb,” *Proceedings of the IEEE PES Conference on Innovative Smart Grid Technologies* (ISGT 2010), pp. 1-9 , 2010

2.“IEEE Standard for a Precision Clock Synchronization Protocol for Nettowrked Measurement and Control Systems,” *IEEE 1588-2002 Standard*, November 2002

3.J. C. Eidson and B. Hamilton, “IEEE-1588 Node Synchronization Improvement by High Stability Oscillators”, in *Proceedings of Workshop IEEE-1588*, pp.102-112, 2003.

4.X. Li, Y. Chen, and S. Liang, “Improvement of Precise Time Synchronization Algorithm based on IEEE 1588,” *Proceeding of International Conference on Computer, Mechatronics, Control and Electronic Engineering* (CMCE 2010), pp.70 - 73, 2010

5.IEC61850-5, Communication Networks and Systems in Substations, part5: Communication Requirements for Functions and Device Models, 2003

6.Carta, N. Locci, C. Muscas and S. Sulis, “A Flexible GPS-Based System for Synchronized Phasor Measurement in Electric Distribution Network” *IEEE Transactions on Instrumentation and Measurement*, Vol. 57, No. 11, pp.2450-2456, NOVEMBER 2008

7.Distribution Test Feeder Working Group, http://ewh.ieee.org/soc/pes/dsacom/testfeeders/index.html, 2011

8.L. Mills David, “Precision Synchronization of Computer Network Clocks,” *ACM SIGCOMM Computer Communication Review*, Vol.24, No.2, pp.28-43, April 1994

9.M.Hojo, K. Abe, Y.Mitani, H. Ukai, O. Saeki, “Real-Time Power System Monitoring at Demand Sides by Campus Wide Area Measurement System,” in *Proceedings of the 35th Annual Conference of the IEEE Industrial Electronics Society*, pp.3609 - 3614, November 2009.

10.Z. Zhong, C. Xu and B. J. Billian, “Power System Frequency Monitoring Network (FNET) Implementation,” *IEEE Transactions on Power Systems*, Vol. 20, No. 4, pp. 1914-1921, November 2005.

11.E. W. Dijkstra, "A Note on Two Problems in Connexion with Graphs," *Numerische Mathematik* 1,pp. 269–271, 1959

12.] M. Lixia, C. Muscas, and S. Sulis, “Application of IEEE 1588 to the Measurement of Synchrophasors in Electric Power Systems,” in *Proceedings of International IEEE Symposium on Precision Clock Synchronization for Measurement, Control and Communication Brescia* (ISPCS 2009), pp.1-6, October 2009

13.Carta, N. Locci, C. Muscas, F Pinna, and S Sulis, “GPS and IEEE 1588 Synchronization for the Measurement of Synchrophasors in Electric Power Systems,” *Computer Standards & Interfaces*, Vol. 33, No. 2, pp. 176-181, February 2011

14.“IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems,” *IEEE 1588-2008 Standard*, July 2008.