Message Efficient Time Synchronization in Wireless Sensor Networks

Presented by L. C. Chen

July. 12, 2007

Outline

- Introduction
- Related Works
- Network Model
- Time Synchronization Scheme
- Time Synchronization Scheme in heterogeneous network
- Simulation
- Conclusion

Introduction(1)

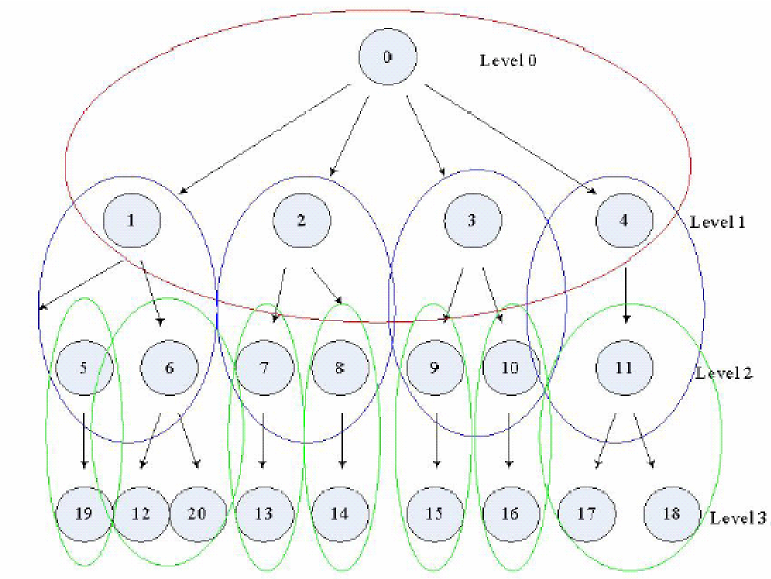
- Time synchronization is an important problem in wireless sensor networks.
- A lot of operation needs it.
  - Data aggregation
  - TDMA schedules
  - Synchronized sleep
  - Periods
- Most of time synchronization methods in traditional internet and wireless LAN do not consider the limited resourced.

Introduction(2)

- This protocol uses overhearing concept and multi-hop architecture suitable for WSN.
- We use bi-directional links to do two-way synchronization.
- Relative offset
  - Through every node maintains an individual clock, these clocks are not synchronized with respect to each other.
- Relative drift
  - Timers between two nodes are not exactly the same.

Related works-NTS[1]

- Through creating a spanning tree, all the nodes are connected with one another.
- The concept of subtree makes the whole synchronization process be divided into multiple repeated sub-processes.
- NTS didn't adapt to the situation of mobility nodes.



Related works-TPSN[2]

- TPSN is based on the simplistic approach of conventional sender-receiver time synchronization.
- TPSN need execute “level discovery phase” before initiating the “synchronization phase”.

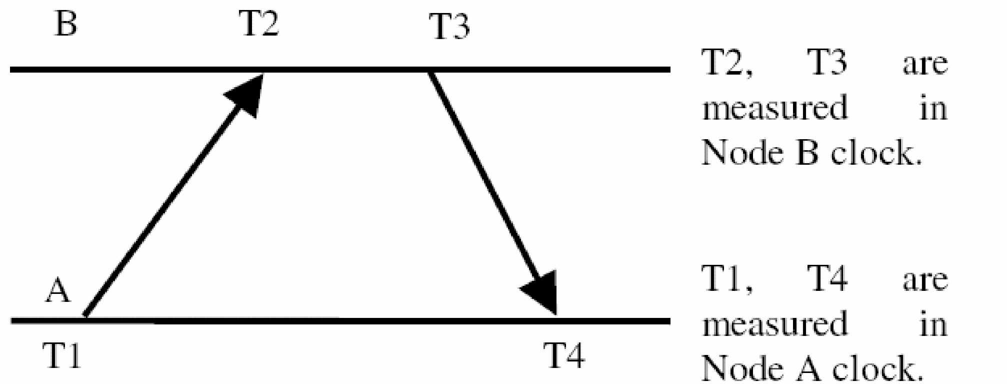
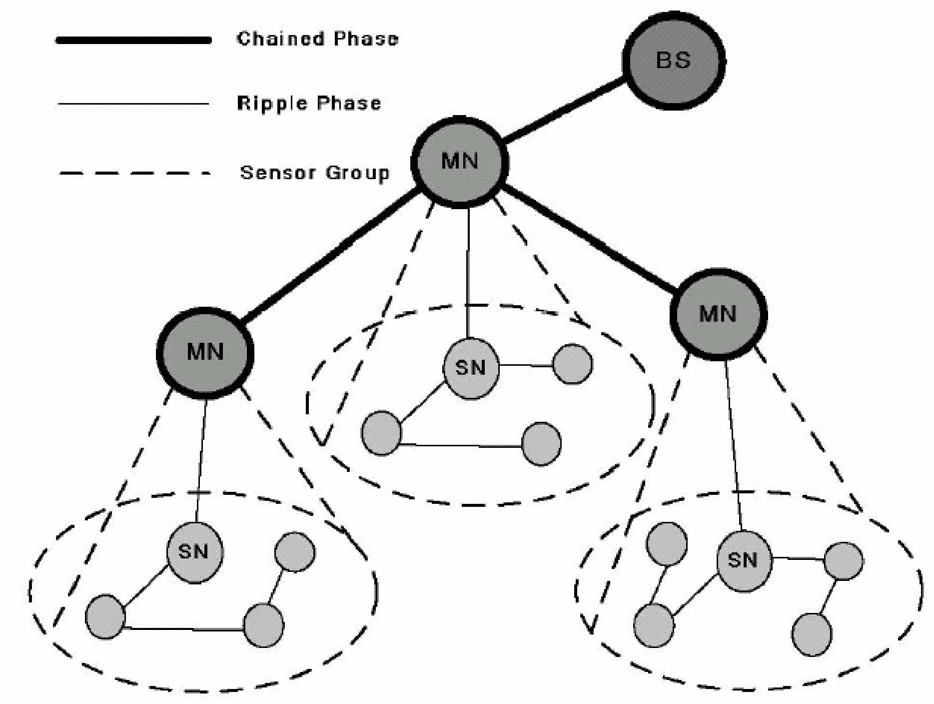


Figure 1: Two-way message exchange between pair of nodes

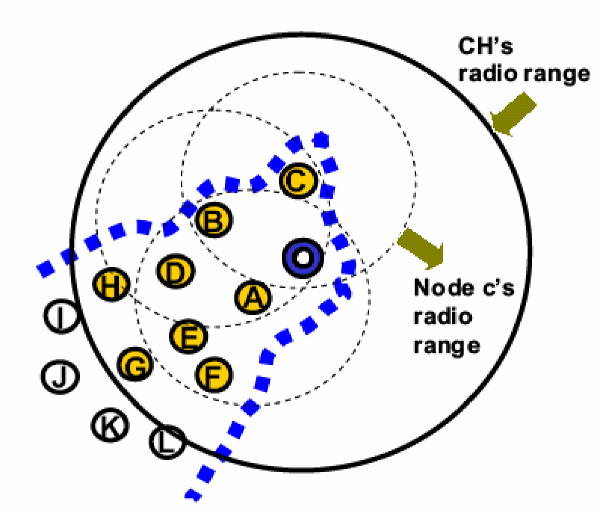
Related works-CRIT[3]

- CRIT contributes in the accurate hierarchical and multi-hop time synchronization with low error-rate and efficient clock offset.
- They didn't discuss the detail about how to build the architecture.



Related works-CHTS[4]

- CHTS introduced the hierarchical clustering protocol to decrease the synchronization error by reducing the average hop count from the reference node to each node.
- They didn't discuss the detail about the architecture of "CH layer".

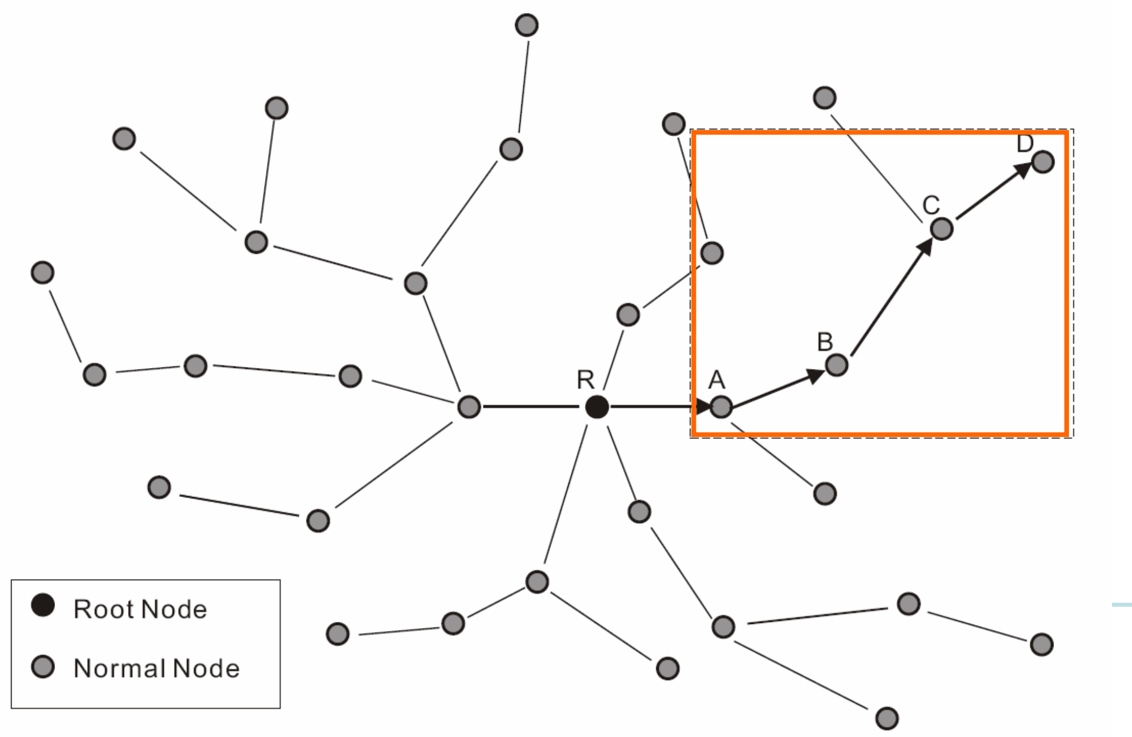


Network Model

- There are N sensor nodes in a rectangle area.
- Every sensor node has a location (x,y).
- We determine that the transmission range is a constant R.
- There are three states for nodes：
  - General state
  - Preparative state
  - Synchronized state

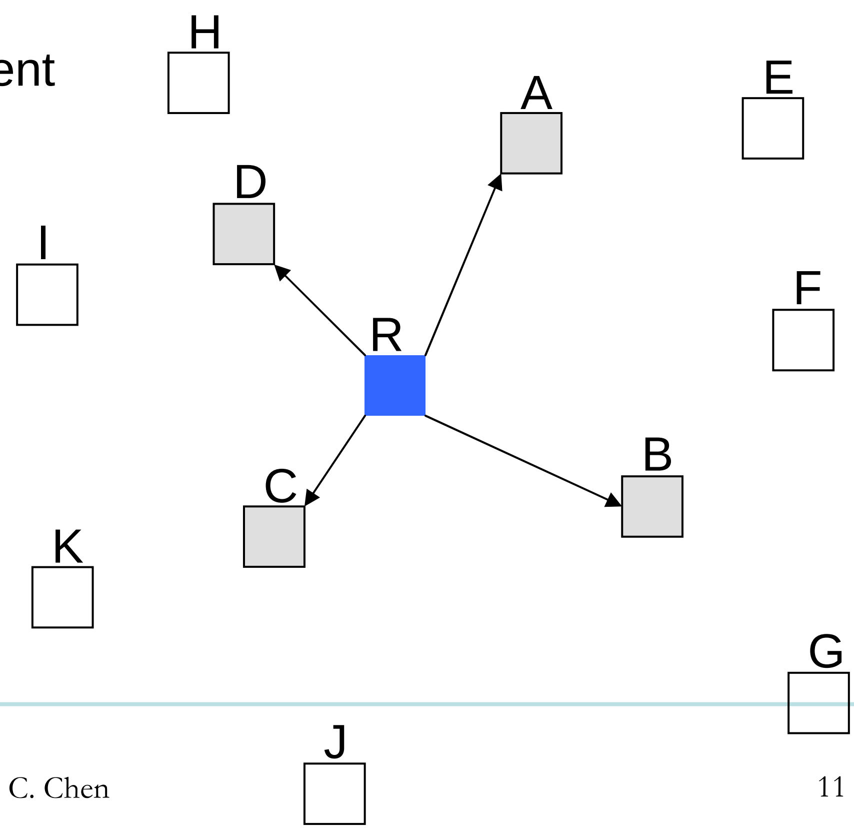
Network Model

- The reference node like sink in network manages the time synchronization for all of nodes in environment acted as the gateway between the sensor network and the external world.
- After the wireless sensor network is deployed, root node floods a beacon message periodically.



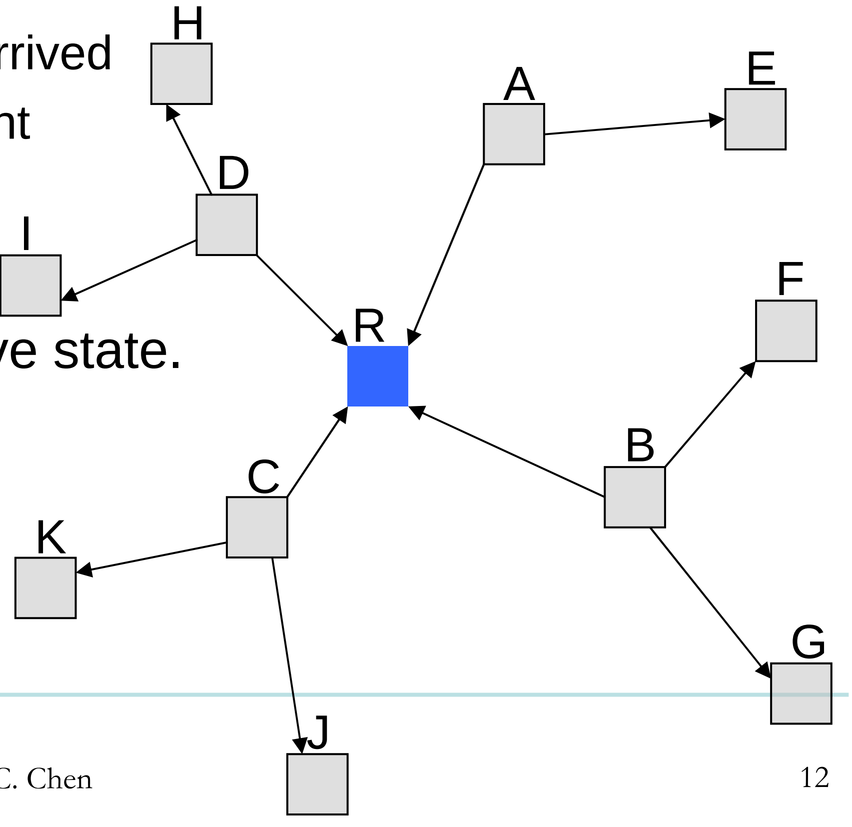
Time Synchronization Scheme

- At the beginning of a round, all nodes are in the general state except for reference node.
- The reference node floods a *snotify_packet* to begin a round.
  - ID
  - the timestamp after this packet sent
  - HopCount



Time Synchronization Scheme

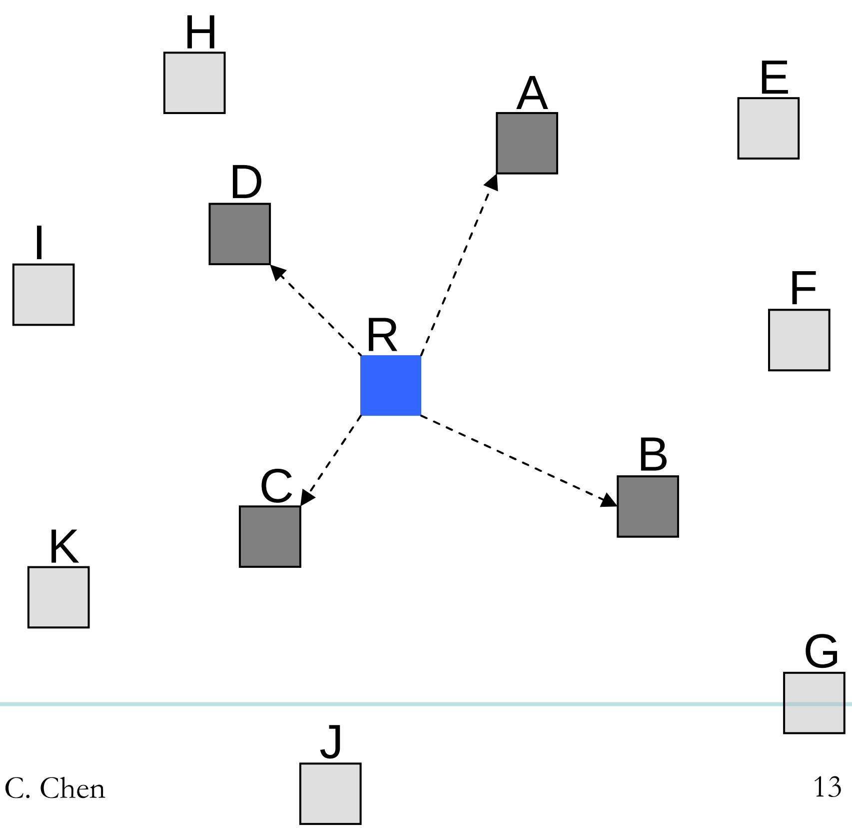
- After the node receives this packet, the node also floods its *snotify_packet*.
  - ID
  - parent node's ID
  - the timestamp when last packet arrived
  - the timestamp after the packet sent
  - HopCount
- The node goes into the preparative state.



Time Synchronization Scheme

- When the parent node receives the *snotify_packet* from the nodes, it will calculate the offset for the node.

- The parent node is in the synchronized state, it will waits for some time to collect all the information from the children nodes and then floods a *soffset_packet*.
  - ID
  - all of the children node's ID
  - all of the children node's offset



Time Synchronization Scheme

- If the node receives the *snotify_packet* from the node when it is in the **synchronized state**, it can calculate the offset for the children node by equation (1).

(1) Offset(X) = {(T4(X) - T3(X))-[T2(X) -(offset(parent's ID) + T1(X))]}/ 2

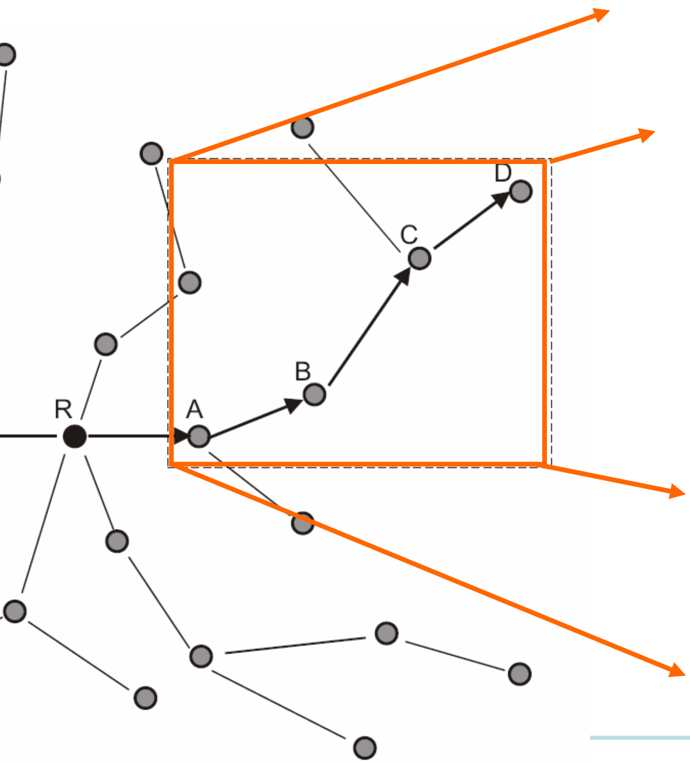
- If the node receives the *snotify_packet* from the node when it is in the **preparative state**, we will calculate the offset by equation (2).

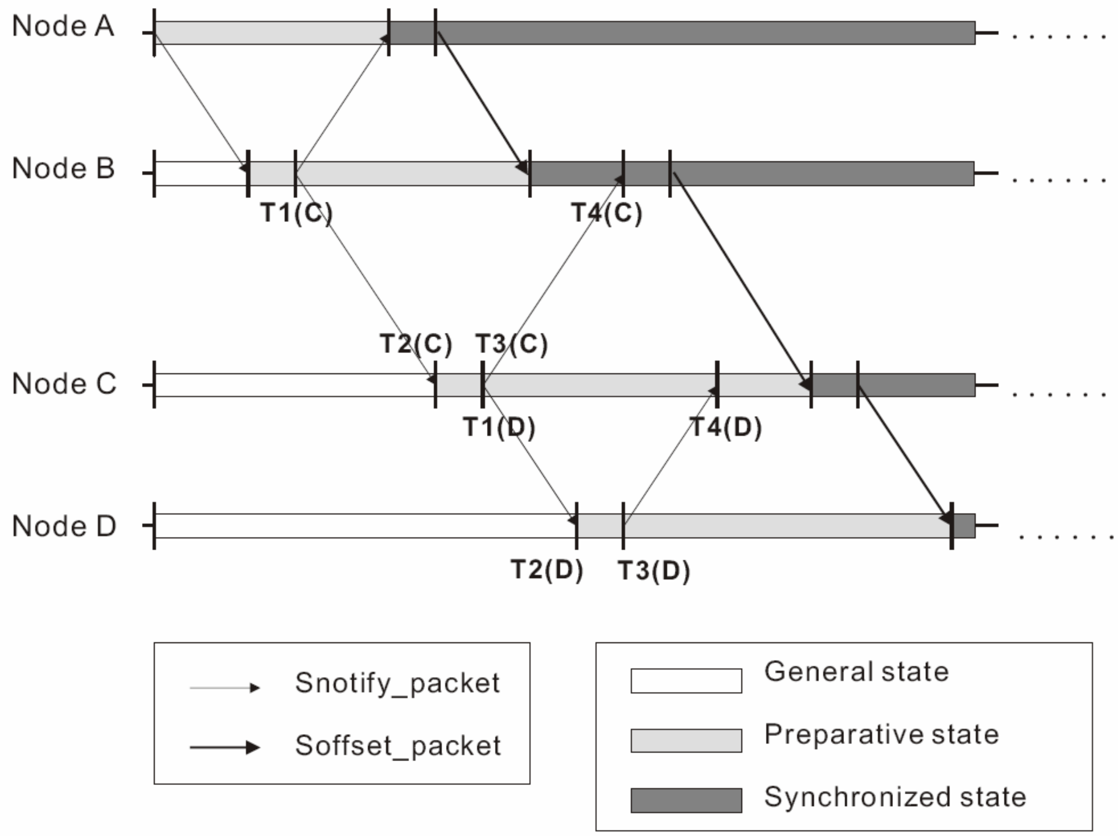
(2) Offset(X) = [(T4(X) - T3(X))-(T2(X) - T1(X))]/ 2 + offset(parent's ID)

Time Synchronization Scheme

Offset(C) = {(T4(C) - T3(C))-[T2(C) -(offset(B) + T1(C))]}/ 2

Offset(D) = [(T4(D) - T3(D))-(T2(D) - T1(D))]/ 2 + offset(C)

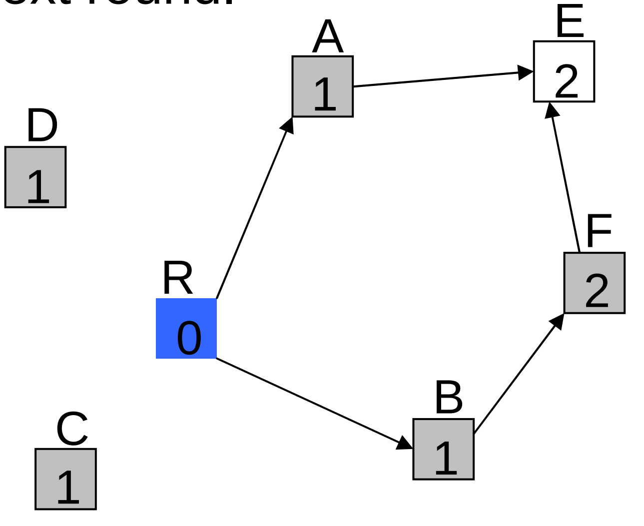




Time Synchronization Scheme

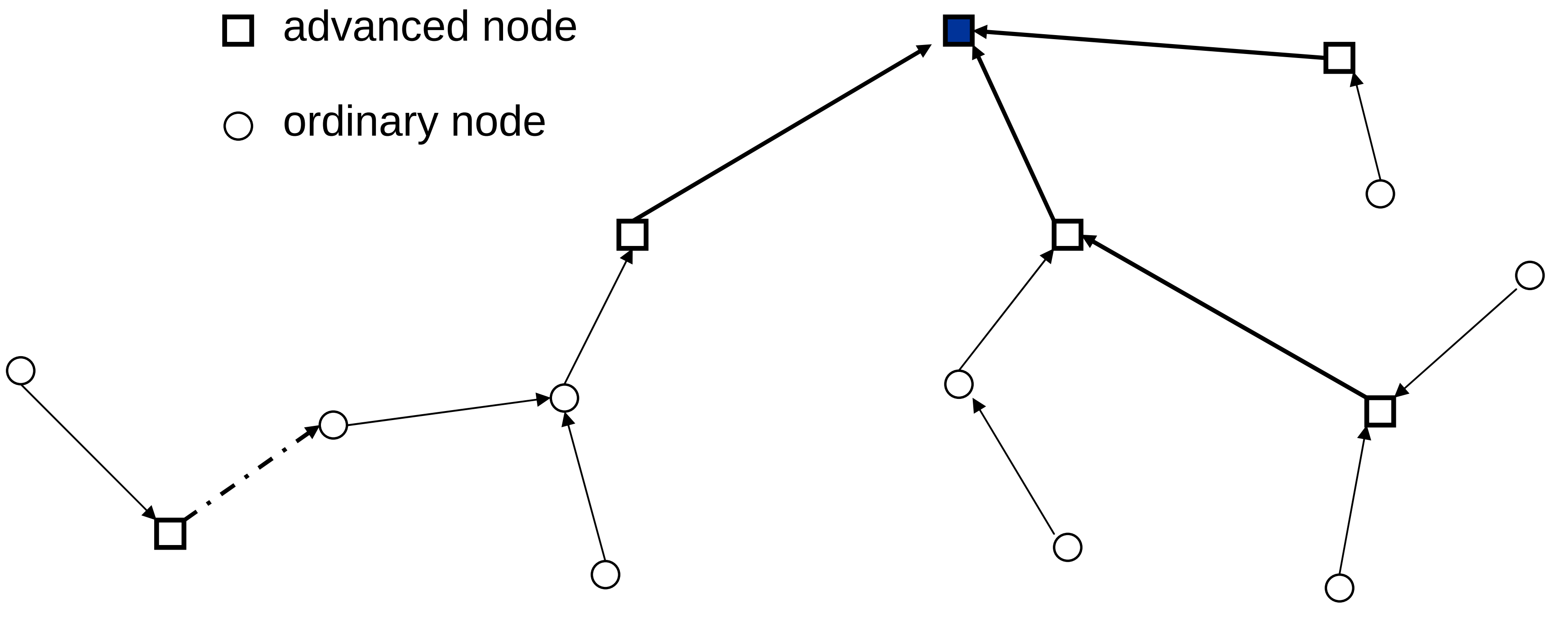
- If a node in the preparative state or the synchronized state receives the second *snotify_packet*, we will compare these two *HopCounts*.

- If the *HopCount* in the snotify_packet is less than that in the node, the node will record the information of the *snotify_packet* serving as the reference for next round.



Time Synchronization Scheme in heterogeneous network

- In order to reduce the error, we provide an algorithm for decreasing hop counts through the clustering of nodes.



Time Synchronization Scheme in heterogeneous network

- We add a parameter *Advanced_HopCount* to make advanced nodes as close to the reference nodes as possible to achieve the result of decreasing the hop count.

- When an advanced node wants to flood its *snotify_packet*, it will increase an *Advanced_HopCount* to record the hop counts among those advanced nodes in its packet.

Time Synchronization Scheme in heterogeneous network

- snotify_packet for advanced nodes:
  - ID
  - parent node's ID
  - the timestamp when last packet arrived
  - the timestamp after the packet sent
  - HopCount
  - *Advanced_HopCount*

- snotify_packet for ordinary node:
  - ID
  - parent node's ID
  - the timestamp when last packet arrived
  - the timestamp after the packet sent
  - HopCount

Time Synchronization Scheme in heterogeneous network

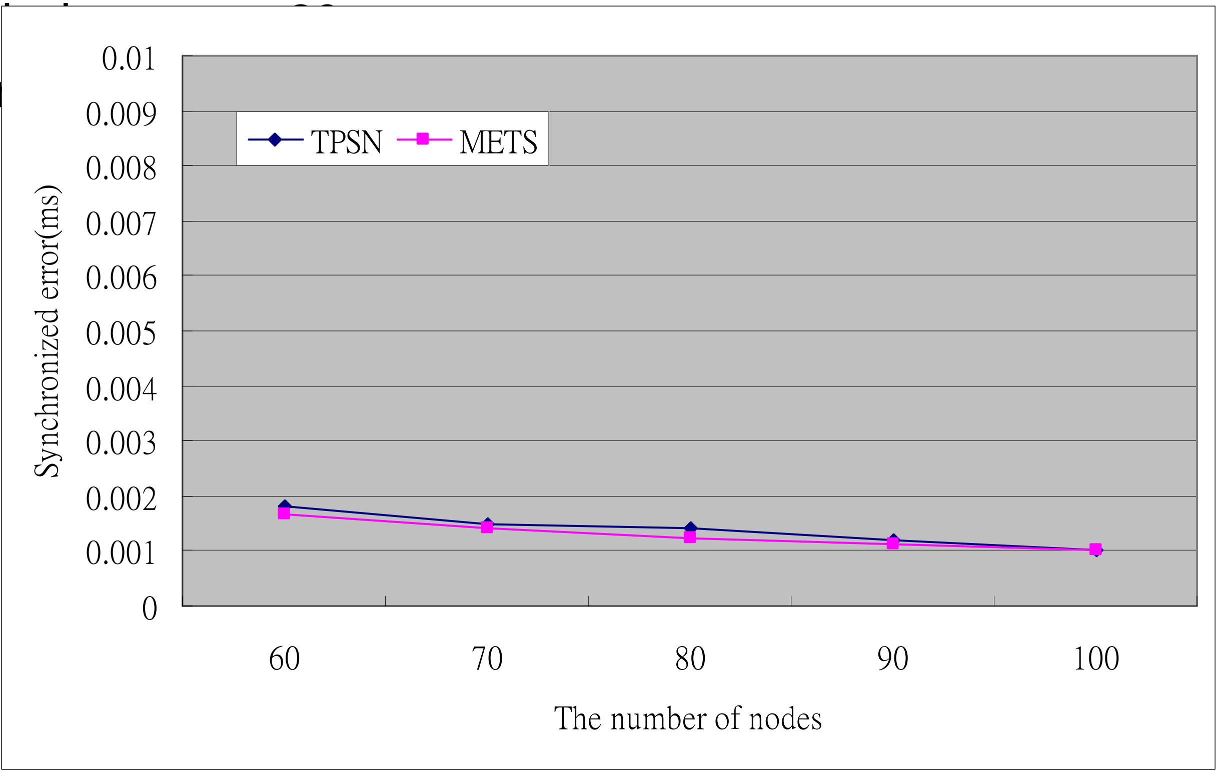
- If a node receives a *snotify_packet* excluding the *Advanced_HopCount* parameter, it will execute according to the scheme of the above.

- If we both have *Advanced_HopCount* and *HopCount* information, we compare the value of *Advanced_HopCount* first. The node will select the node of smaller *Advanced_HopCount to* be its parent.

- Only if two packets have the same value of *Advanced_HopCount*, we choose the value of *HopCount* to decide the parent.

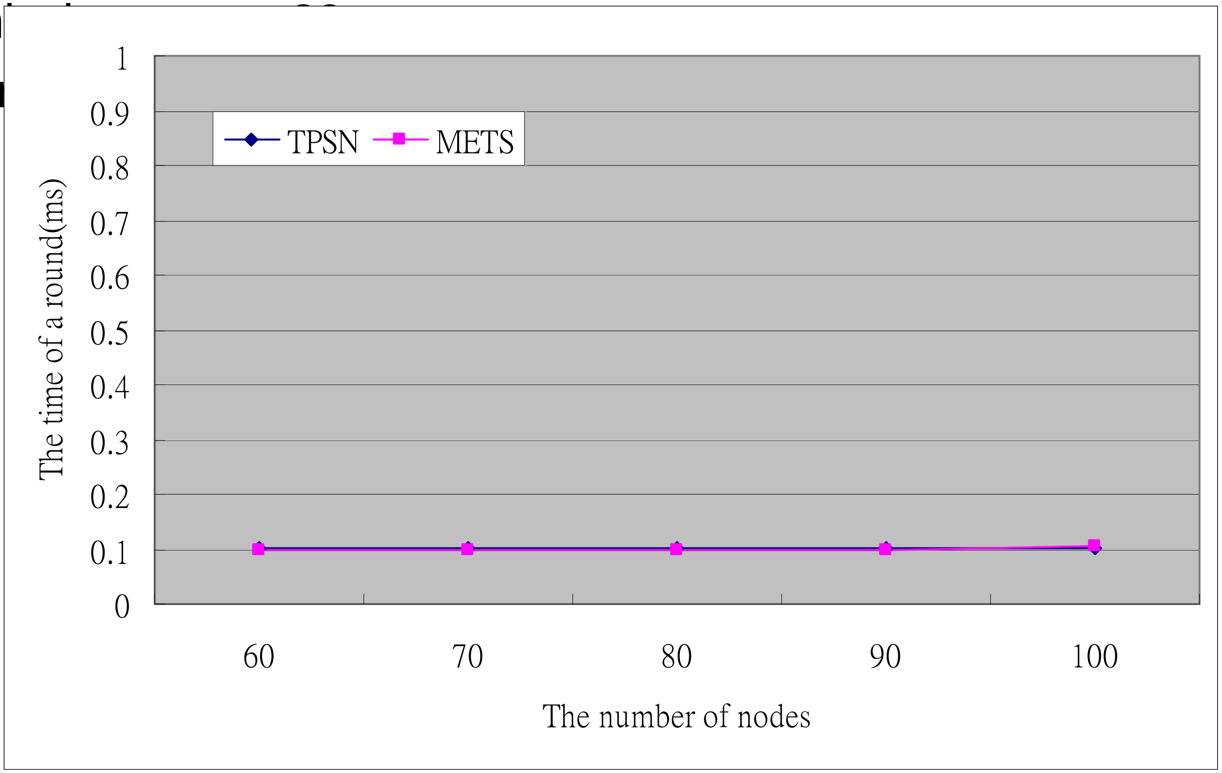
Simulation(1)

- Area: 100m*100m
- Transm
- Numbe
- Round:



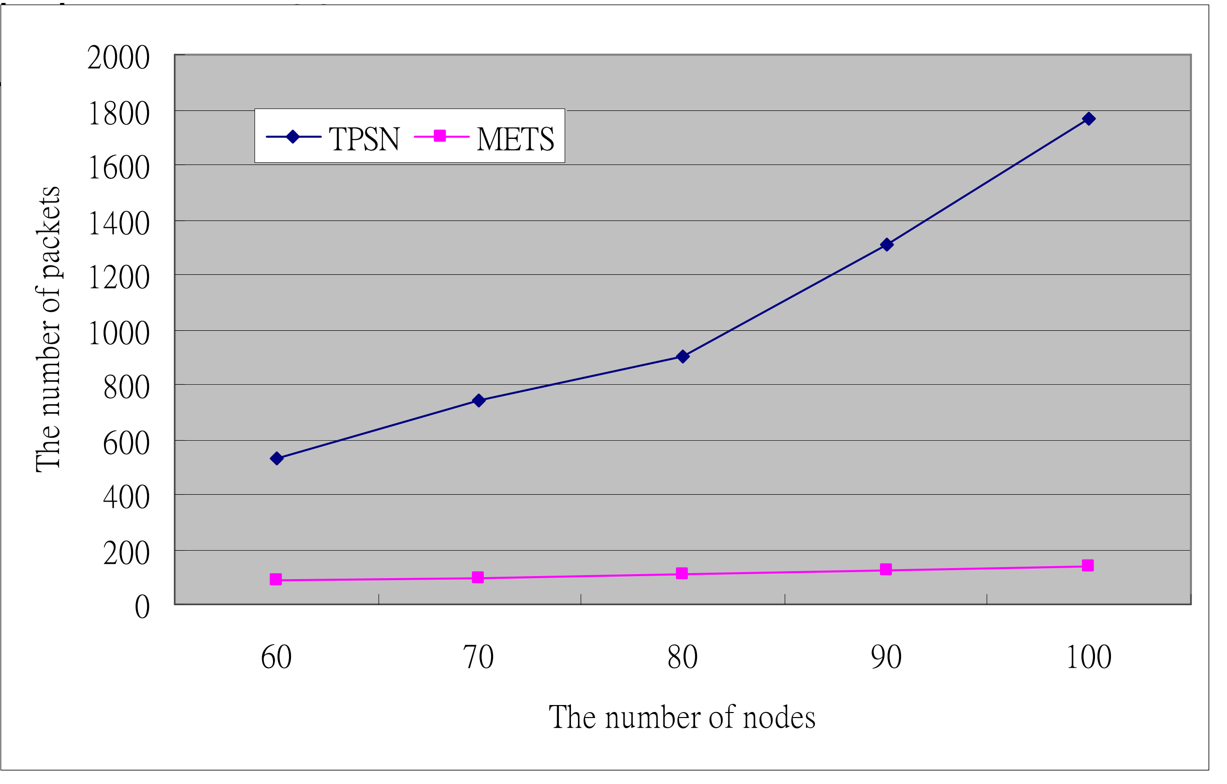
Simulation(2)

- Area: 100m*100m
- Transm
- Numbe
- Round:



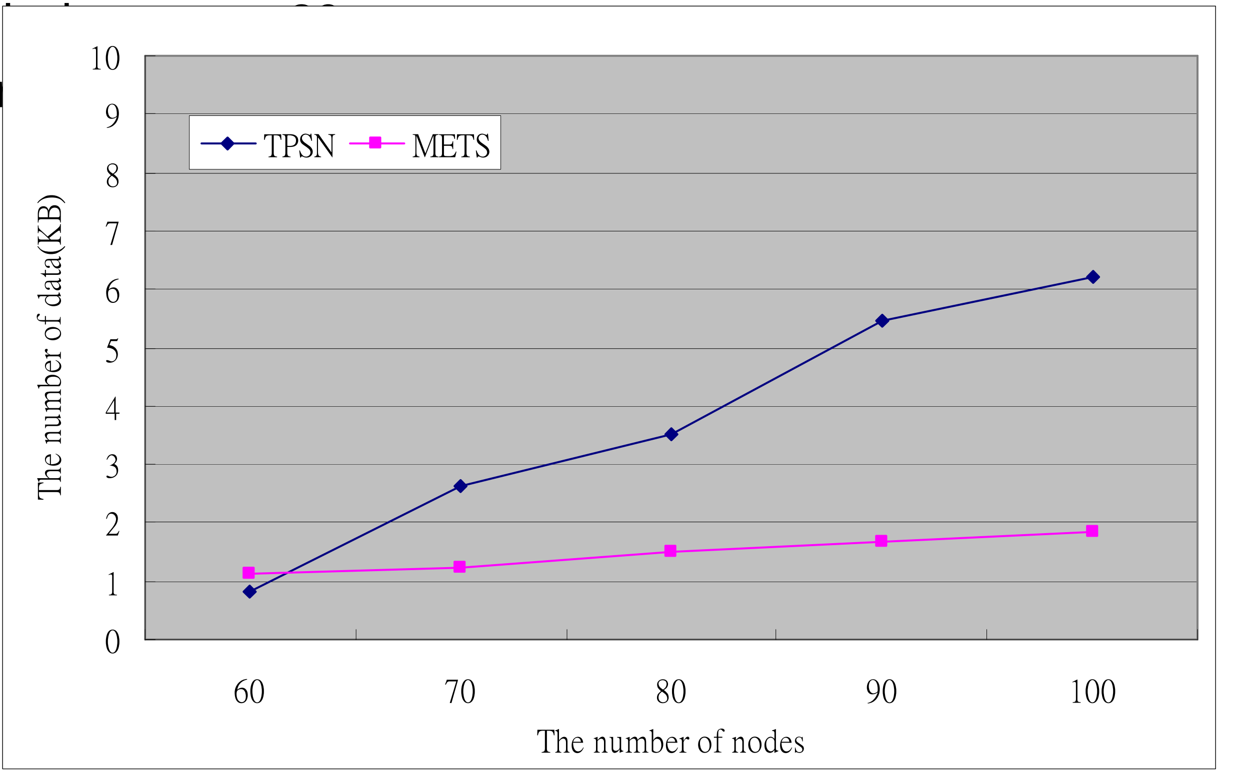
Simulation(3)

- Area: 100m*100m
- Transm
- Number
- Round:



Simulation(4)

- Area: 100m*100m
- Transm
- Numbe
- Round:



Conclusion

- We introduce a simple, flexible, energy-efficient, and always-on time synchronization called METS for wireless sensor networks.
- METS utilizes the concept of overhearing to decrease the transmission message for calculating the clock values.
- Through the simulations, METS certainly save a lot of power through.
- The algorithm also adapts to heterogeneous wireless sensor network.

References

- Liming He, and G.S. Kuo, “A Novel Time Synchronization Scheme in Wireless Sensor Networks”, VETECS, 2006.
- S. Ganeriwal, R. Kumar, and M. Srivastava, “Timing-Sync Protocol for Sensor Networks”, *The First Int. Conf. on Embedded Networked Sensor Systems*, Los Angeles, California, November 2003.
- K.Y. Shin, K.Y. Lee, and K. Lee, “CRIT: A Hierarchical Chained-Ripple Time Synchronization in Wireless Sensor Networks”, ICNSC,April 2006, pp. 23-25.
- H. Kim, D. Kim, and S.E. Yoo, “Cluster-based Hierarchical Time Synchronization for Multi-hop Wireless Sensor Networks”, *Proceedings of the 20th International Conference on Advanced Information Networking and Applications*, 2006.

Thank you!