Lightweight Time Synchronization for Sensor Networks

Jana van Greunen
University of California, Berkeley
janavg@eecs.berkeley.edu

Jan Rabaey
University of California, Berkeley
jan@eecs.berkeley.edu

WSNA'03, September 19, 2003, San Diego, California, USA

Cihat ÇETİNKAYA

Abstract

- Lightweight tree-based synchronization for sensor networks
- Single-hop synchronization
- Multi-hop synchronization
  - Centralized
  - Distributed

Outline

- Introduction
- Related Work
  - General Synchronization Techniques
  - Related Work in Sensor Network
- Pair-Wise Synchronization
- Multi-Hop Synchronization
  - Centralized Multi-Hop Synchronization
  - Distributed Multi-Hop Synchronization
- Simulation and Results
- Future Work
- Conclusion

Introduction

- Many applications of sensor networks depend on the time accuracy kept by nodes
- Events are timestamped with the node's local time
- Require synchronization, local time to a global time
- Traditional synchronization algorithms
  - (+) Minimizing the synchronization error
  - (+) Achieving maximum accuracy
  - (-) Computation and communication energy
- **In sensor network energy is a highly constrained**

Introduction (contd)

- In this paper,
  - **Argue** communication and computation requirements of synchronization can be significantly reduced by taking advantage of the relaxed accuracy constraints.
  - **Introduce** synchronization schemes that sacrifice accuracy by performing synchronization less frequently and between fewer nodes.

Introduction (contd)

- LTS algorithms
  - Designed to work with generic low-cost sensor nodes
  - Focus on minimizing overhead (energy) while being robust and self-configuring
  - Operate correctly in the presence of node failures, dynamically varying channels and node mobility.

Related Work
General Synchronization Techniques

- Classification of synchronization algorithms by Anceaume and Puaut[4].
  - Resynchronization event detection
    - identifies the time at which nodes have to resynchronize their clocks
  - Remote clock estimation
    - determine the local time of another node in a network
  - Clock correction
    - update the local time of a node when a resynchronization event has occurred

Related Work (contd)
General Synchronization Techniques

- General synchronization techniques
  - focus on achieving maximum accuracy.
- Our approach
  - the objective is to minimize complexity(and therefore energy) of the synchronization algorithm
  - The accuracy is given as a constraint.

Related Work
Sensor Network

- RBS (Reference Broadcast Synchronization)
- TINY/MINI-SYNC
- Level-based synchronization

Related Work(contd)
Sensor Network

- RBS (Reference Broadcast Synchronization)
  - synchronize the local time of two nodes
  - intermediate node transmits a “reference packet” to the two nodes.
  - The two nodes record the time that they received the packet.
  - Exchange this recorded time to find the difference.

Related Work(contd)
Sensor Network

- TINY/MINI-SYNC
  - Based on the assumption that the nodes' clock drifts are of the following linear form
    - $t_i = a_i t + b_i$
    - t_i : local clock of node i
    - a_i, b_i : drift parameters
    - t : real time
  - Nodes exchange timestamped packets to best-fit offset line between the two nodes

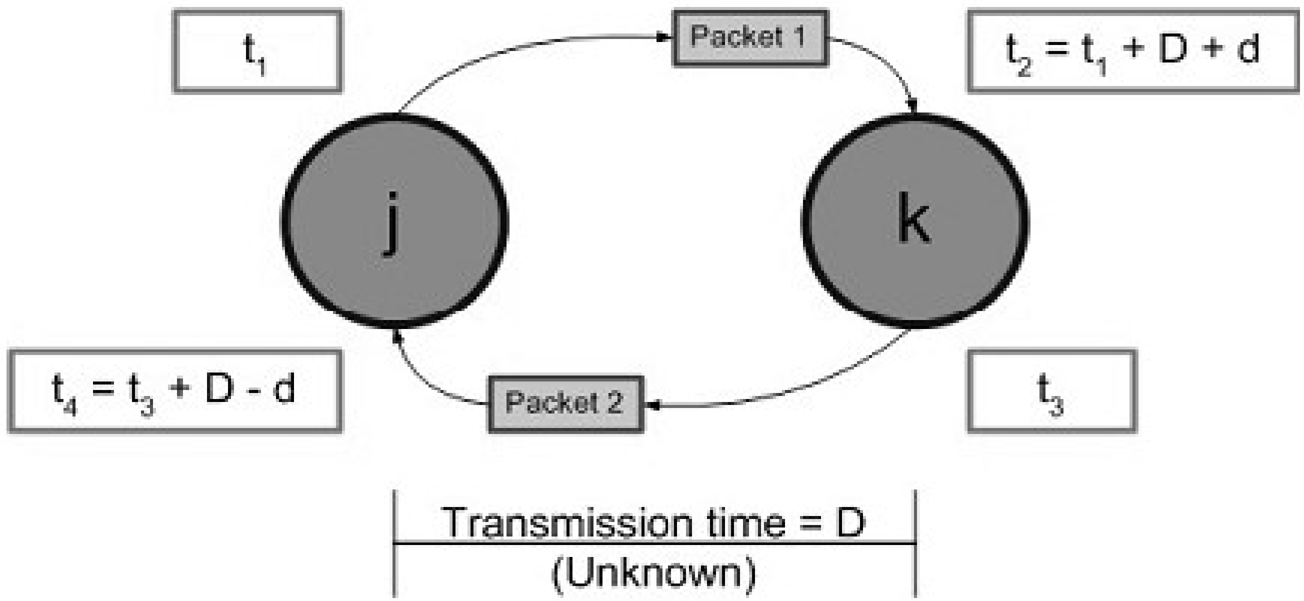
Related Work(contd)
Sensor Network

- Level-based synchronization
  - Introduces the pair-wise sync. used in this paper
  - Simple and computationally efficient
  - Accuracy is determined by the sensor's radio characteristics.

Pair-wise Synchronization

- Single-hop synchronization that exchange of 3 messages
- Nodes j and k synchronization procedure:
  - Node j transmits the first packet with a timestamp t1 with respect to its local time.
  - Node k records the time t2 when it receives the first packet.
    - $t2 = t1 + D + d$
      - D : transmission time(unknown)
      - d : offset between j and k's clock
  - Node k transmits a second packet(including t1 and t2) with a timestamp t3
  - Node j receives the second packet at time t4 and calculates d
    - $t4 = t3 + D - d$

Pair-wise Synchronization(contd)



- The offset d can be calculated at node j by subtracting t_4 from t_2.
  $\boldsymbol{t_2 - t_4 = t_1 - t_3 - D + D + 2d}$
  $\mathbf{d} = 0.5*(\boldsymbol{t_2 - t_4 - t_1 + t_3})$
- The two nodes are synchronized once node j has calculated the offset d. However, a third message is required if the offset d must also be communicated to node k.

Pair-wise Synchronization(contd)

- Underlying Assumption D1 = D2
  - Transmission time is same from j to k and k to j
- Ofcourse D1 and D2 are not exactly equal and this introduces some error in synchronization.
- Kopetz and Schwabl [10] have divided the transmission time (D) into four parts:
  - Send Time
  - Propagation Time
  - Receive Time
  - Access Time

Pair-wise Synchronization(contd)

- Send Time
  - The time spent assembling the message at the sender
  - Includes processing and buffering time.
  - The message is timestamped after the send time has completed
- Propagation Time
  - The time for the signal to propagate across the physical medium between the two nodes
  - Function of the distance between the nodes

Pair-wise Synchronization(contd)

- Receive Time
  - The processing time required for the receiver to receive a message from the channel and notify the host of its arrival.
- Access Time
  - The delay associated with accessing the channel including carrier sensing

Multi-hop Synchronization

- Extension of the pair-wise synchronization.
- A group of n nodes requires n^2 pair-wise synchronizations.
- Due to the relatively low accuracy requirements of our sensor network, we perform pair-wise synchronization only along network edges that form a spanning tree structure
- There are several important considerations.

Multi-hop Synchronization (contd)

- Global Reference
  - We assume that at least one node in the network has access to a global time reference.
- Selective Synchronization
  - Multi-hop synchronization can aim to keep all nodes synchronized at all times, or we can perform selective synchronization
- Resynchronization Rate
  - Due to clock drift, the nodes will periodically need to be resynchronized.

Multi-hop Synchronization (contd)

- Error Estimation & Limitation
  - The synchronization algorithm itself should keep track of accuracy performance and the errors produced by clock drift among nodes.
- Robustness
  - There should not be a single point of failure in the system.
- Mobility
  - Synchronization should work for both stationary or mobile nodes

Centralized Multi-hop LTS

- Simple linear extension of the single-hop synchronization
- The basis of the algorithm :
  - Construction (either offline or dynamic) of a low-depth spanning tree T comprising the nodes in the network
  - Pair-wise synchronizations are performed along the edges of T.

Centralized Multi-hop LTS (contd)

- In order to synchronize nodes in tree
  - The reference node
    - initates the sync. by synchronizing with all immediate (single-hop) children in T.
  - Each child of reference node
    - Synchronizes with their subsequent children
  - Terminates when all the leaf nodes have been synchronized.

Centralized Multi-hop LTS (contd)

- Analysis Of Errors
  - Sync. error increases along each branch
  - Low-depth tree is efficient
- Creating a Spanning Tree
  - Construct a spanning tree that maximizes the sync. accuracy.
  - Optimal tree is one with minimum depth.
  - To minimize running time sync. should occur in parallel. (BFS)
  - BFS has higher communication overhead .
  - DDFS developed by Awerbuch[12]

Centralized Multi-hop LTS (contd)

- Efficiency
  - Communication cost = Spanning Tree Const. + Pair-Wise Sync along tree's n-1 edges.
  - Pair-Wise Sync has fixed overhead of 3 messages total of 3n-3
  - DDFS has has overhead of 4*m
  - Total = 3n-3 + 4m per network synchronization

Distributed Multi-hop LTS

- Performs node synchronization in a distributed fashion
- Does not make use of an overlay spanning tree to direct the pair-wise synchronizations
- Moves resynchronization responsibility from the reference node to the nodes themselves

Distributed Multi-hop LTS (contd)

- When a node j determines that it needs to be resynchronized
  - send a resynchronization request to the closest reference node
  - All nodes along the routing path will be synchronized in a pair-wise fashion

Distributed Multi-hop LTS (contd)

- Avoiding Cycles
  - When the node at the head of the sync. chain requests sync. from a node that is lower down in the same request chain
  - Cycles cause deadlock.
  - Apprroach for avoiding cycles
    - Send sync. Request to neighbor and start timer
    - If timer expires before a sync. response from neighbor arrives, send sync. to different neighbor
    - Does not prevent cycles, reduces impact at an overhead cost of additional synchronizations

Simulations and Results

- Simulation Setup
  - Omnet++ and C++
- Implementation Details
  - 500 node
  - 120m*120m rectangular area
  - Radio range 10m
  - Single reference node that placed in the center of the rectangular area, keeps accurate time
  - All nodes are aware of their own locations, location of reference node and single-hop neighbor.
  - Location information is used only to construct tree

Simulations and Results (contd)

- The success probability for a packet transmission is Bernoulli with parameter p
- p is either 0.95 or 0.65
- Required accuracy = 0.5 seconds
- Drift of clocks = 50 ppm
- Simulation execution time = 36000 seconds

Simulations and Results (contd)

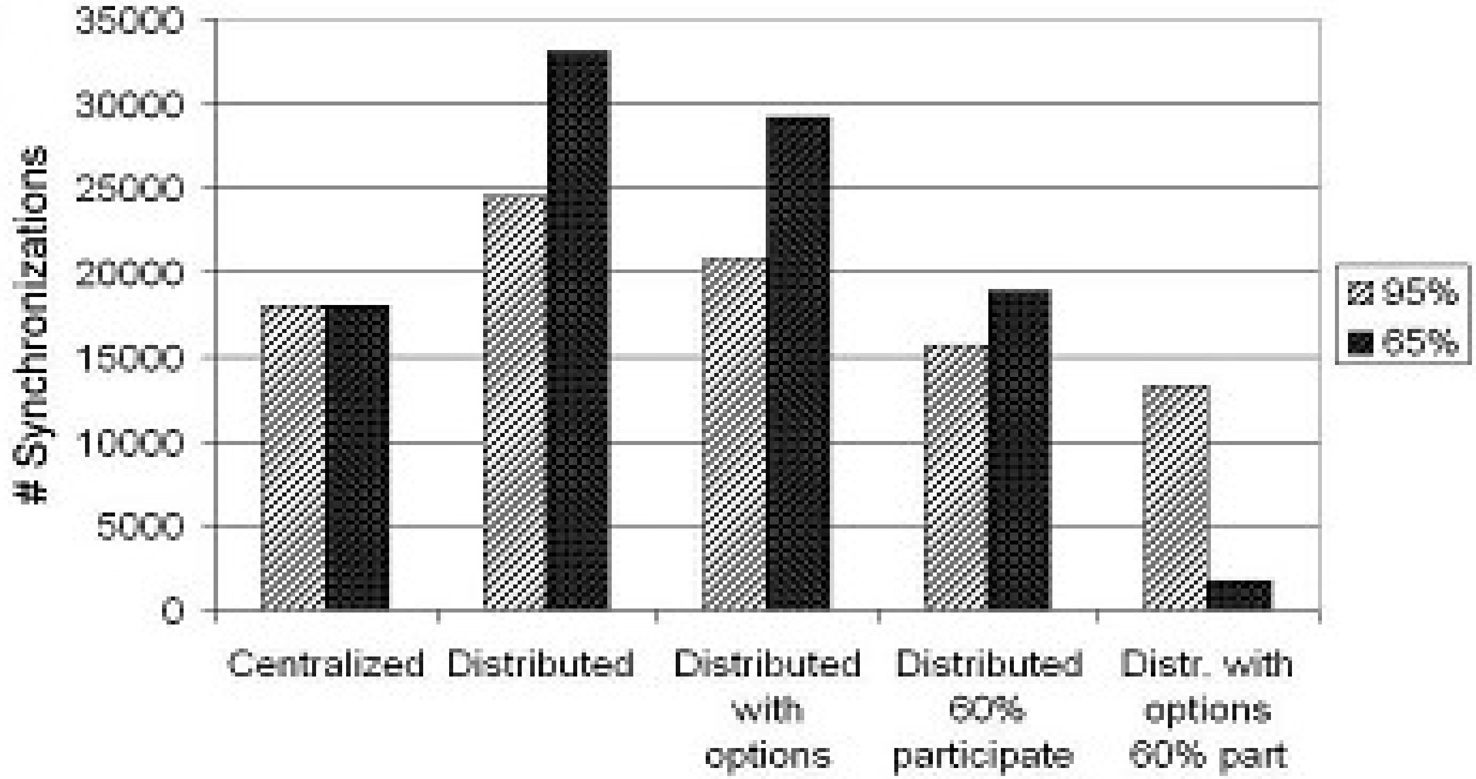


Figure 4: Number of synchronizations for different algorithms and channel quality.

Simulations and Results (contd)

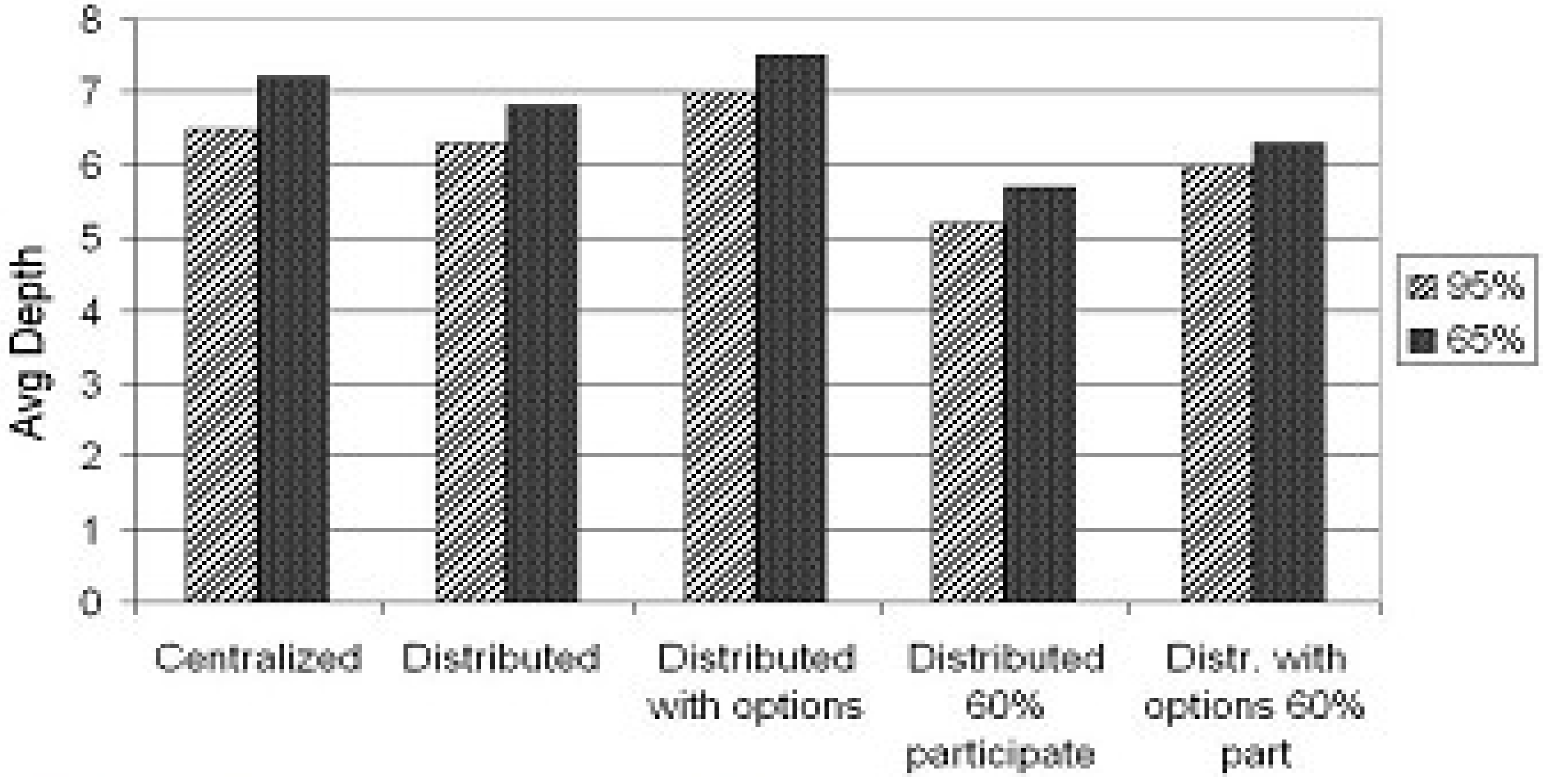


Figure 5: Average depth of synchronization tree for different algorithms

Simulations and Results (contd)

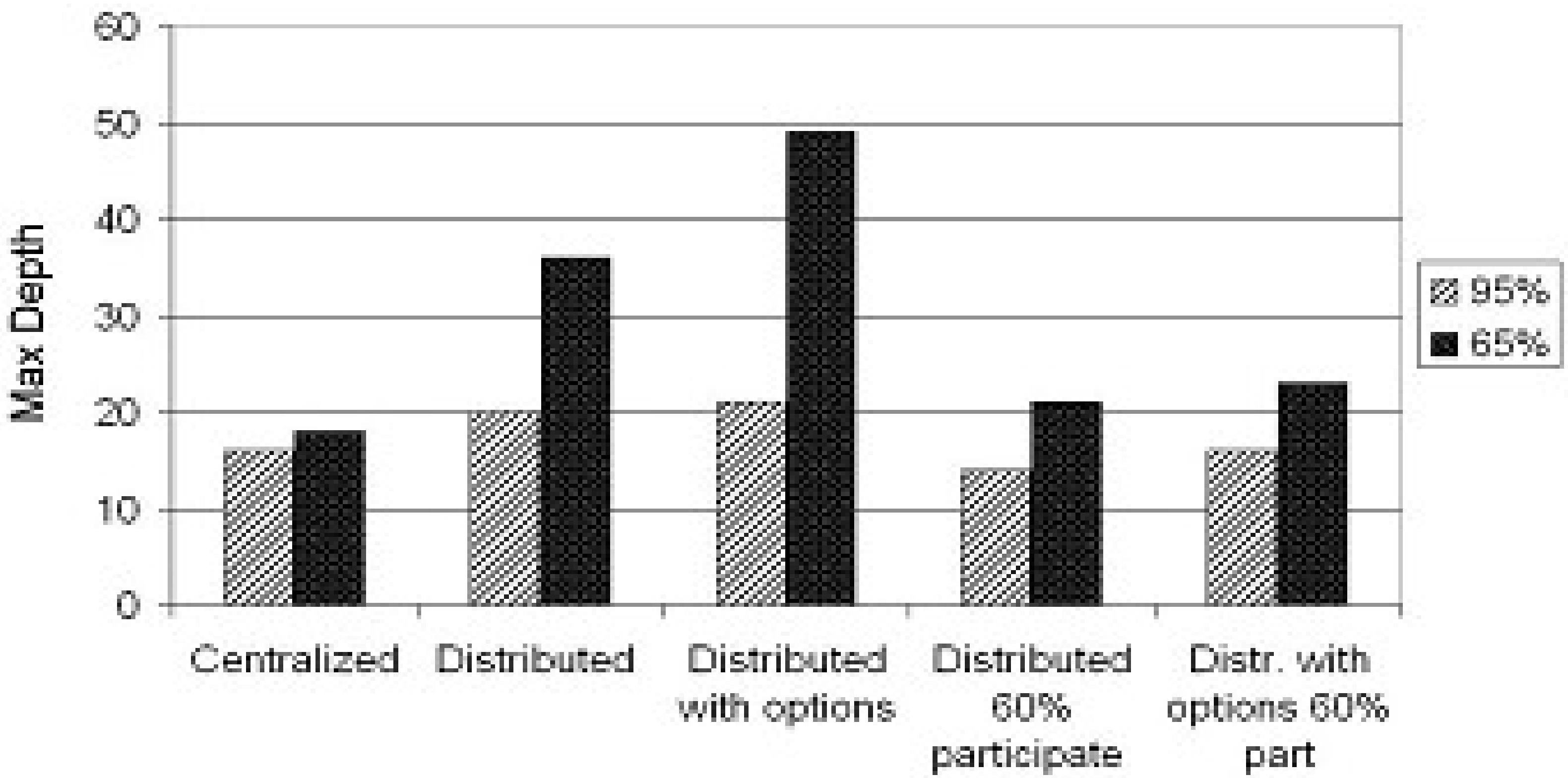


Figure 6: Maximum depth of synchronization tree for different synchronization algorithms.

Simulations and Results (contd)

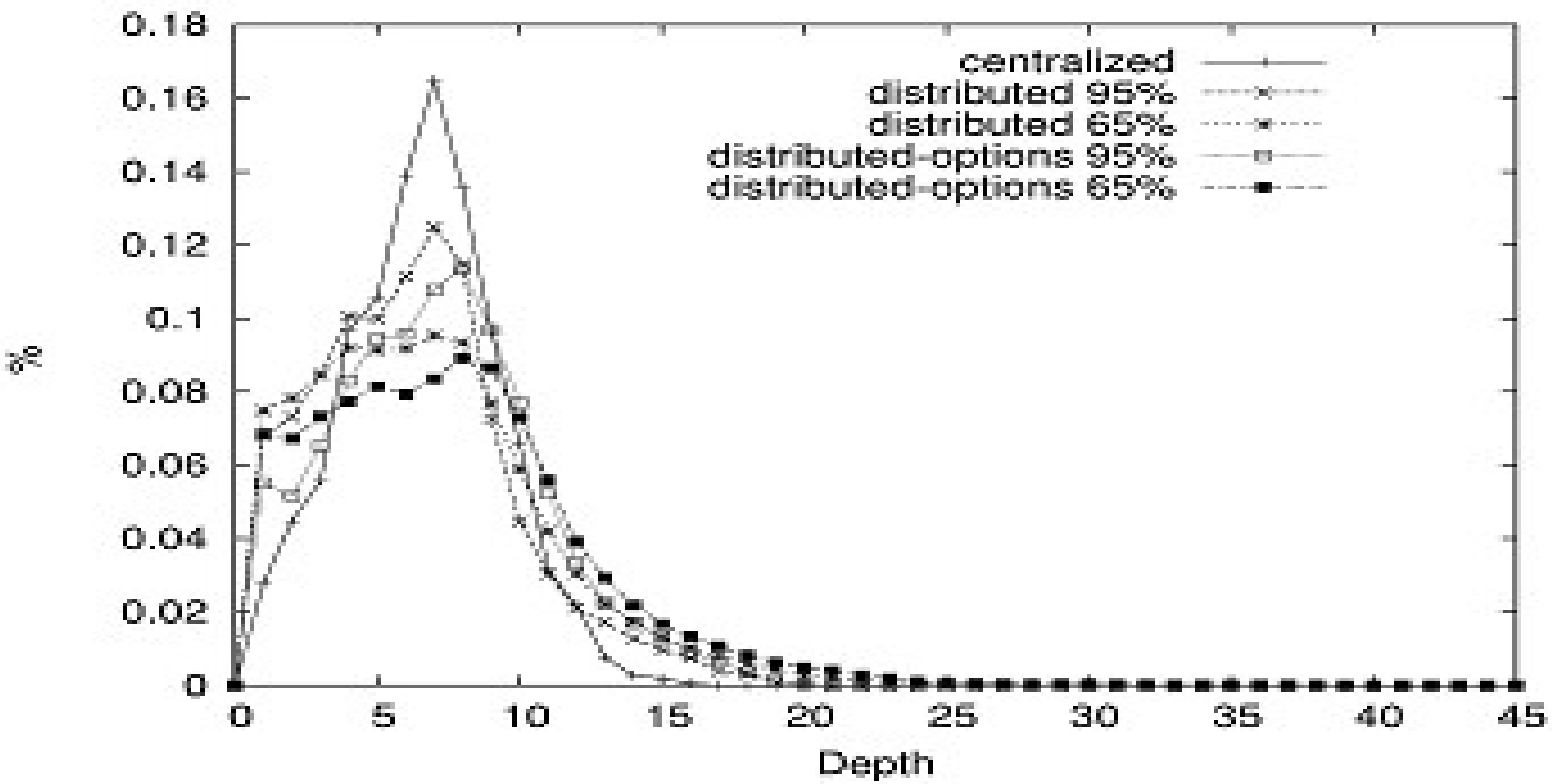


Figure 7: Percentage of synchronizations as a function of depth in synchronization tree

Simulations and Results (contd)

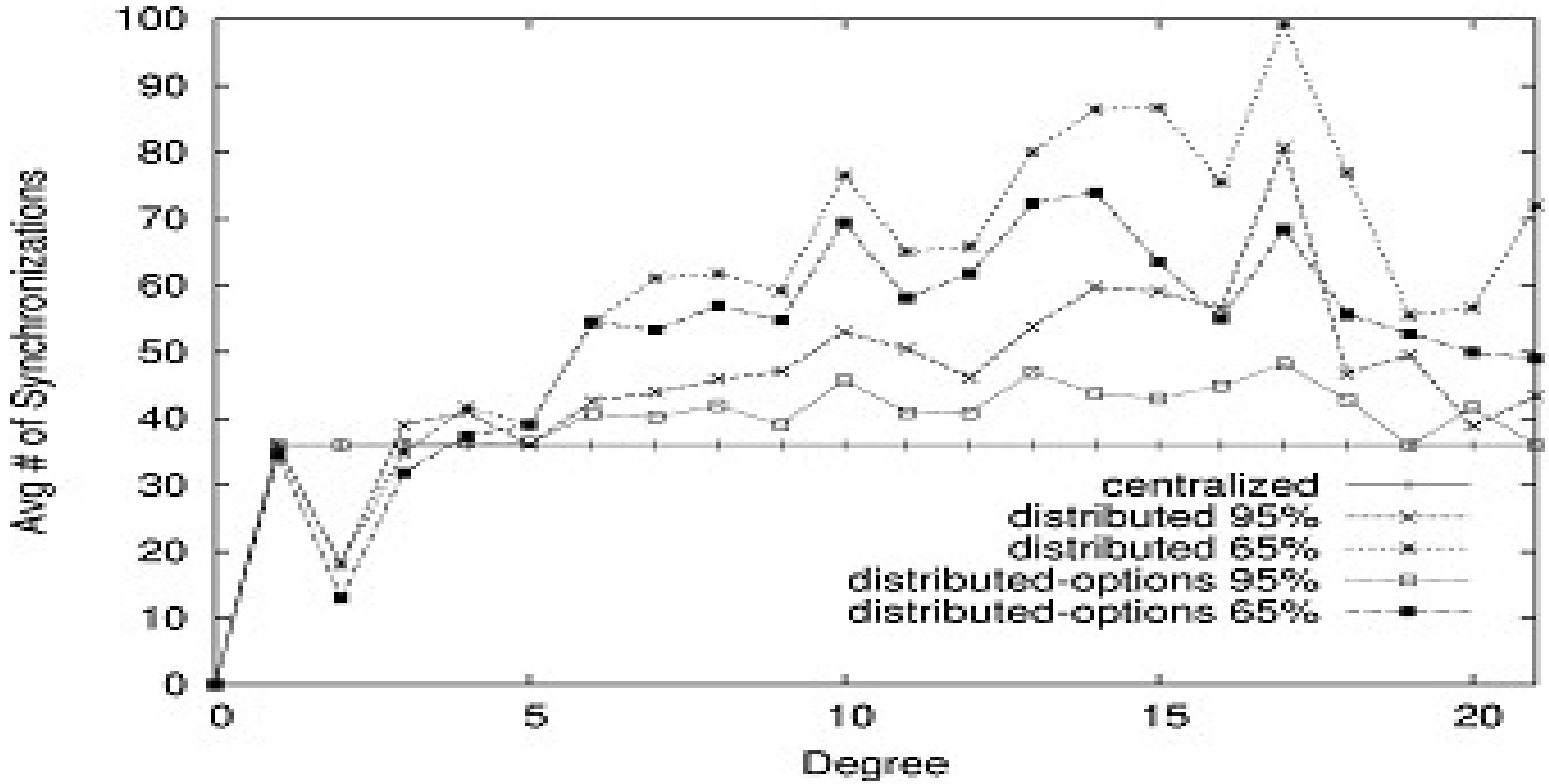


Figure 8: Average number of synchronizations as a function of node degree

Simulations and Results (contd)

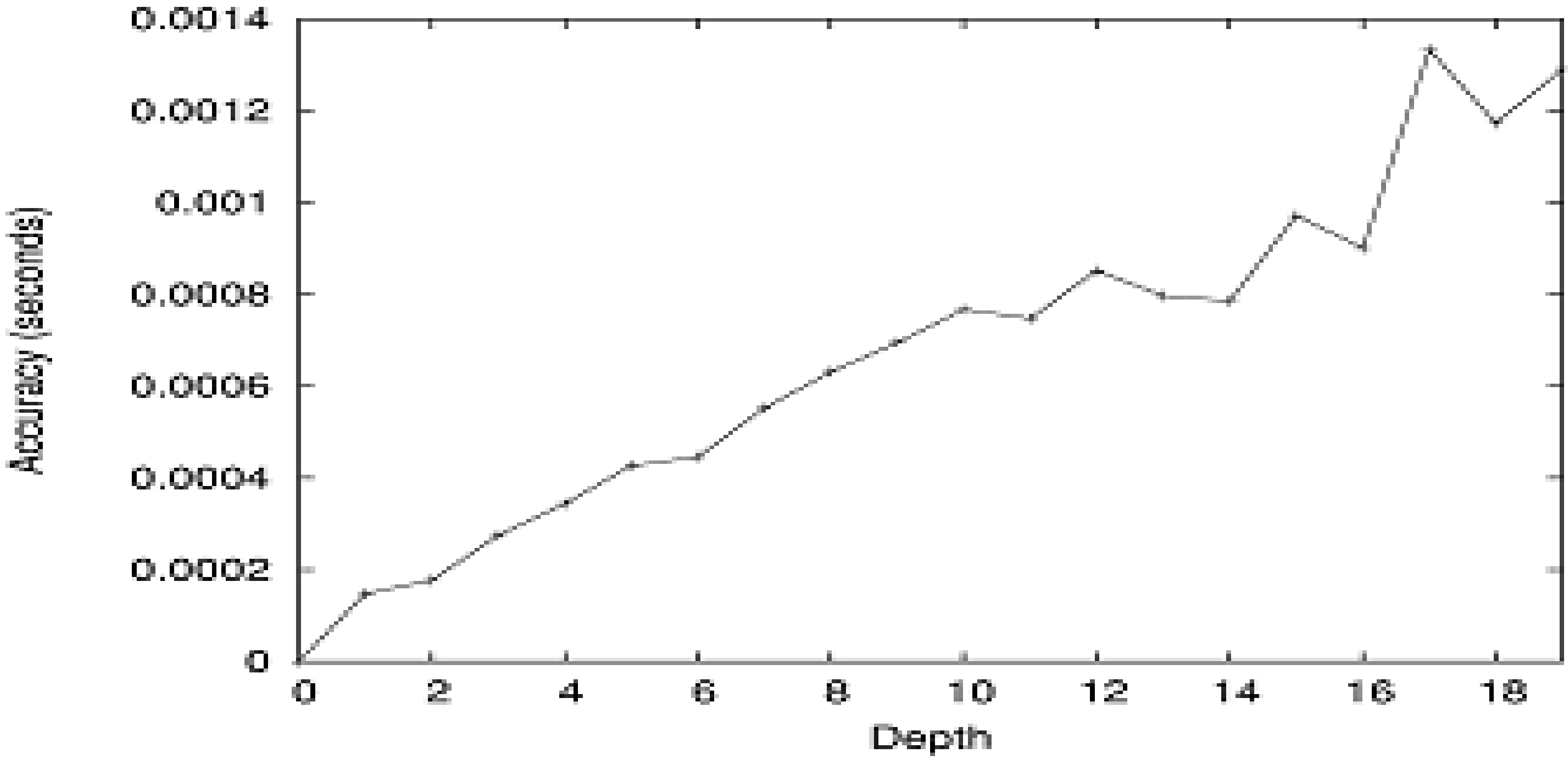


Figure 9: Accuracy of synchronization as a function of node depth in the tree

Simulations and Results (contd)

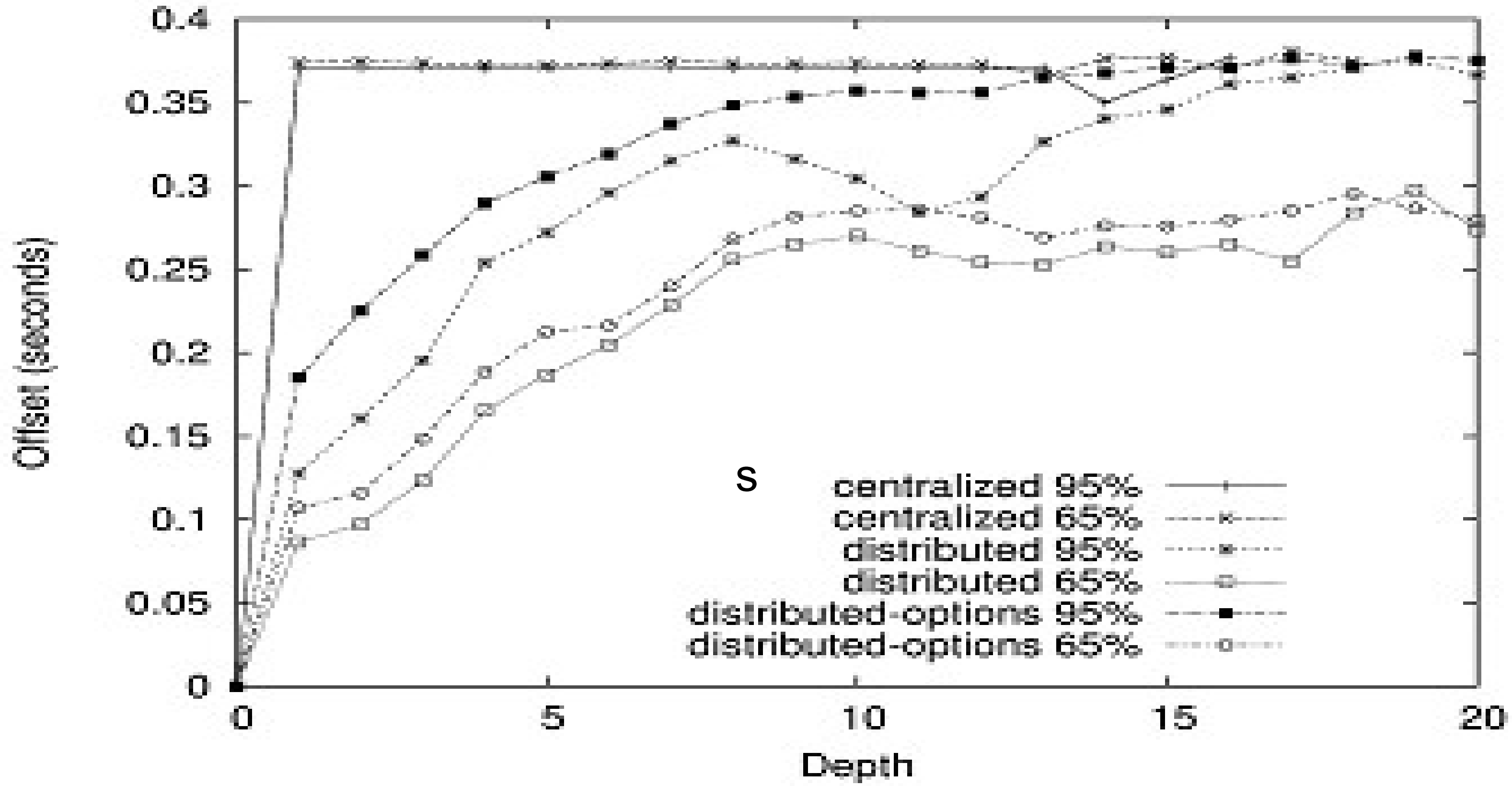


Figure 10: Average time offset before synchronization as a function of node depth in tree

Simulations and Results (contd)

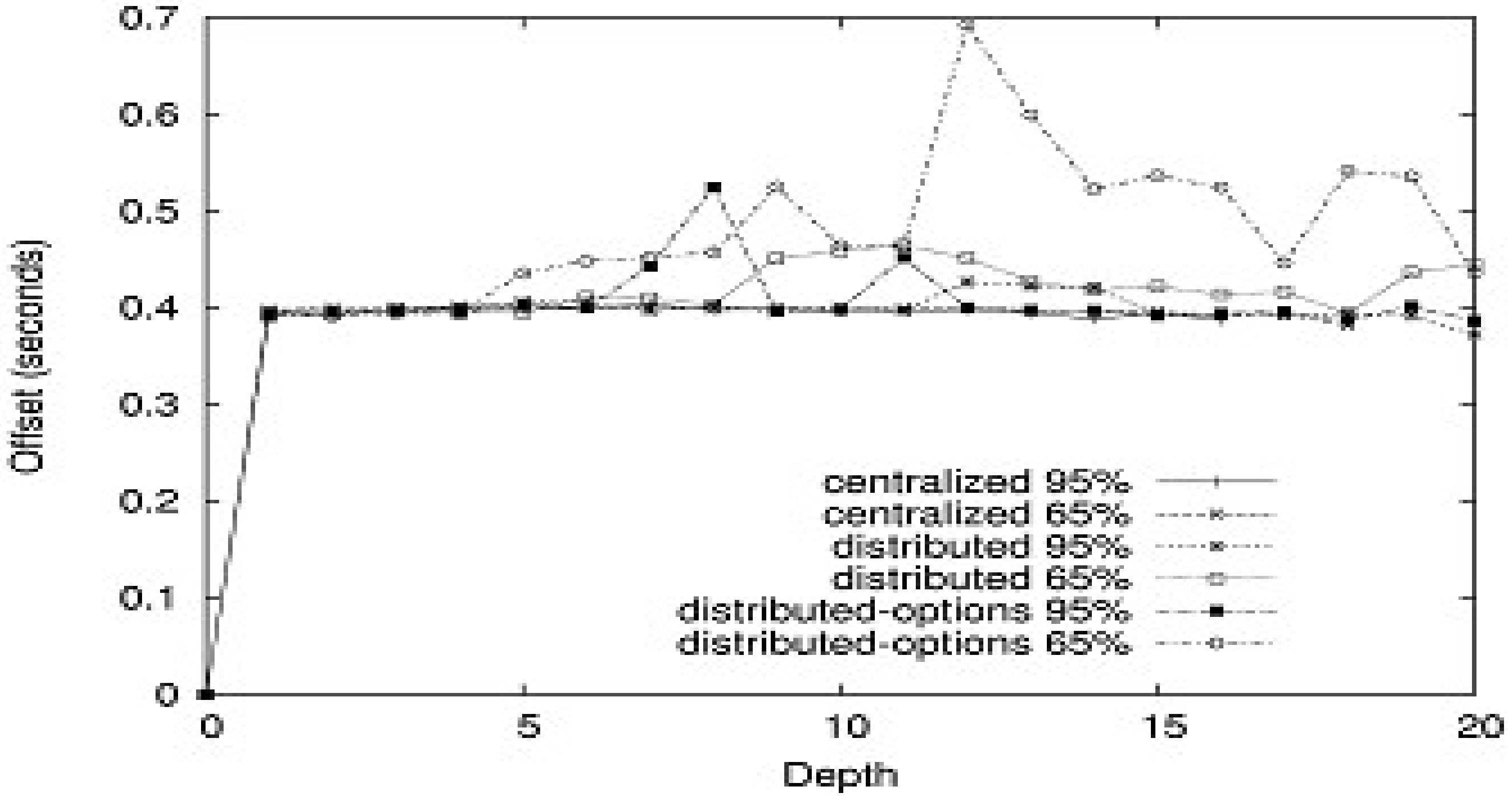


Figure 11:Maximum time offset before synchronization as a function of node depth in tree

Future Work

- The LTS schemes presented in this paper rely on the reliability and correctness of information from all nodes along the path to the reference node.
- The synchronization will fail if there are
  - Byzantine faults
  - Clock failure
  - Malicious misinformation
- LTS algorithms may be updated to function correctly in the presence of these malicious faults.

Conclusion

- The required time accuracy of most sensor network applications is relatively low.
- The LTS scheme is an effective way to give up accuracy for gains in energy efficiency.
- Centralized vs Distributed
  - When all nodes participate => Centralize
  - When portion of nodes => Distributed

References

- [1] J. Rabaey, J. Ammer, T. Karalar, S. Li, B. Otis, M. Sheets, T. Tuan, PicoRadios for Wireless Sensor Networks: The Next Challenge in Ultra-Low-Power Design in Proceedings of the International Solid-State Circuits Conference, San Francisco, CA, 2002.
- [2] B. Hofmann-Wellenhof, H. Lichtenegger, and J. Collins GPS Theory and Practice, SpringerWienNewYork, 1997.
- [3] D. Mills, Network Time Protocol (Version 3) Specification, Implementation and Analysis, from http://www.faqs.org/ftp/rfc/rfc1305.pdf.
- [4] E. Anceaume and I. Puaut , A Taxonomy of Clock Synchronization Algorithms, Research report IRISA,NoPI1103, July 1997.

References

- [5] J. Elson, L. Girod, and D. Estrin, Fine-Grained Network Time Synchronization using Reference Broadcasts, Proceedings of the Fifth Symposium on Operating systems Design and Implementation, Boston, MA. December 2002.
- [6] M.L. Sichitiu and C. Veerarittiphan, Simple, Accurate Time Synchronization for Wireless Sensor Networks. IEEE Wireless Communications and Networking Conference, WCNC 2003
- [7] Saurabh Ganeriwal, Ram Kumar, Sachin Adlakha and Mani Srivastava, "Network-wide Time Synchronization in Sensor Networks," Technical Report UCLA, April 2002.
- [8] S. Mitra and J. Rabek, Power Efficient Clustering for Clock Synchronizarion in Dynamic Multi-hop Sensor Networks,from http://theory.lcs.mit.edu/~mitras/courses/6829/project/project_main.html.

References

- [9] J. Elson and K. Römer, Wireless Sensor Networks: A New Regime for Time Synchronization, Proceedings of the First Workshop on Hot Topics In Networks (HotNets-I), Princeton, New Jersey. October 28-29 2002.
- [10] H. Kopetz, W. Schwabl. Global time in distributed real time systems. Technical Report 15/89, Technishe Univesität Wien, 1989.
- [11] Warneke, B. Atwood, K.S.J. Pister, Smart Dust Mote Fore-runners, Proceedings of the Fourteenth Annual Interna-tional Conference on Microelectromechanical Systems (MEMS 2001), Interlaken, Switzerland, January 21-25, 2001, pp. 357-360.
- [12] B. Awerbuch, A new distributed depth first search algo-rithm, Inf. Proc. Lett. 20 (1985), 147-150.

References

- [13] A. Boukerche, C. Tropper, A Distributed Graph Algorithm for the Detection of Local Cycles and Knots, IEEE Trans. Parallel and Distributed Systems, 1998, pp. 748-758
- [14] A. Varga, “The OMNeT++ Discrete Event Simulation System,” in European Simulation Multiconference (ESM’2001), Prague, Czech Republic, June 2001.
- [15] C. Guo, L. C. Zhong and J. M. Rabaey, "Low Power Distributed MAC for Ad Hoc Sensor Radio Networks", Proceedings of IEEE GlobeCom 2001, San Antonio, November 25-29, 2001