Clock Synchronization in Delay Tolerant Sensor Networks

바르터쇠 야로홉스키*, 신승중*, 류대현*
*한세대학교 IT 학과

e-mail : bart@ihansei.com

Clock Synchronization in Delay Tolerant Sensor Networks

Bartosz Jarochowski*, Seung-Jeung Shin, Dae-Hyun Ryu
*Dept. of Information Technology, Hansei University

A b s t r a c t

For applications involving the monitoring of large areas, dense sensor networks are not practical. For such applications, delay tolerant networks which consist of disconnected clusters of sensors that are visited periodically by a mobile robot are implemented. Because clock synchronization is critical to any data collection endeavor, and because the structure of DTNs is unique, this paper examines various clock synchronization algorithms as they apply to DTNs. A simulation tool was developed to examine and evaluate the RBS clock synchronization algorithm for DTNs.

1. Introduction

Most previous research in the area of wireless sensor networks has been mostly focused on a rather dense network topology. In practical applications sensor networks are used to monitor large areas and thus must be implemented utilizing a sparse topology. In order to observe a very large area, these types of sensor networks may be linked together in a chain-type topology, without any redundant connections between nodes. Indeed, in some cases, the network may become disconnected and segmented into clusters of independently operating nodes. A number of research challenges arise because of these unique characteristics.

A Delay Tolerant Sensor Network (DTN) describes a network of sensors where nodes are distributed over a large area. These nodes may be grouped in a number of clusters that are disconnected from each other. Communication between clusters is facilitated by one or more mobile robots. These mobile robots independently traverse between clusters and perform various functions such as reprogramming nodes, transferring data, and location estimation. Each node cluster has a designated head which aggregates data received from its tails. Typically, clusters are randomly distributed throughout the environment which is being monitored.

DTNs have all of the limitations of standard wireless sensor networks in addition to the fact that real-time communication is not possible with the entire network at any given time. This paper seeks to address the issue of clock synchronization in DTNs.

2. Clock Synchronization and DTN

Clock synchronization is particularly challenging in the context of DTNs because clock synchronization algorithms are not typically delay tolerant. RBS, however, provides an appropriate basis for a clock synchronization algorithm applicable to delay tolerant networks

Reference broadcast synchronization is so named because it utilizes the ability of wireless networks to broadcast the same data to multiple nodes simultaneously. Therefore, two receivers which receive the same message will do so at approximately the same time. If each receiver records the time that the message arrived, all receivers can synchronize together attaining a high degree of accuracy by comparing their local clock values. A sequence of synchronization messages from a given sender is used to estimate the offset and skew of each node's local clock in relation to other nodes in the neighborhood. An advantage of RBS is that it reduces the potential variability in timing messages by having receivers synchronize with each other rather than directly with the sender. This allows two potential causes for inconsistency to be removed. The time spent by the sender to construct, transmit, and wait to access the transmit channel will be consistent for all receiving nodes thus will not affect synchronization.

Assuming one sender and two receivers, RBS uses the following algorithm to synchronize time:

1. The sender broadcasts a reference packet
2. Both receivers receive the packet and record the receive time according to their local clocks
3. Receivers exchange their respective observed times
4. The offset of the clocks are computed based on the difference of the local times at which the receivers received their respective messages

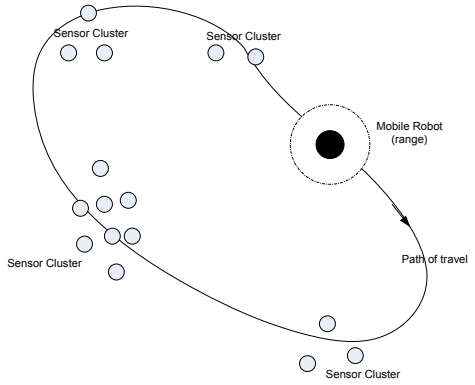

Figure1. DTNRBS System Overview

If each receiver is in the position to process the reference packet immediately, accuracy will be good. Unfortunately, this ideal case is not realistic. Most of the time, receivers are busy performing other tasks. In order to improve accuracy RBS sends a series of reference packets instead of just one. Therefore, receiver j will calculate its offset relative to another receiver i as the average of clock differences for each packet received by i and j:

$$\text{Offset}[i,j]=\frac{1}{m}\sum_{k=1}^{m}\left(T_{i,k}-T_{j,k}\right)$$

Parameters i and j represent two receiving nodes, while m is the number of reference packets sent. $T_{i,k}$ is node i's clock when it receives the broadcast packet k. Because all receivers compute their offsets with all other receivers, $O(n^2)$ offsets are processed. It has been shown in experimentation that in a standard RBS application, the more reference packets are sent, the more accurate the clock synchronization will become.

Given multiple reference broadcasts, the skew between the clocks of neighboring nodes can be computed using a linear least squares method.

In DTN networks, to facilitate reliable monitoring and clock synchronization, clusters should contain at a minimum two nodes. When the mobile robot comes within range of two or more nodes, clock synchronization may occur. Clock synchronization continues as lon as there are nodes with unsynchronized clocks. The robot is externally synchronized by way of GPS, which it also requires for navigation. Since all nodes are synchronized to the mobile robot, it is essential that the robot maintains an accurate and consistent clock.

3. Simulation

An agent-based simulation was constructed in order to demonstrate the application of RBS to DTN networks. The simulation mimics the behavior of RBS as applied to DTN networks. The simulation is initialized with the following parameters: number of nodes, number of reference messages sent per synchronization attempt, transmission noise, robot speed, robot range, and message cycle length. In addition, synchronization error, performance, and the number of nodes synchronized are plotted.

The robot follows a circular path, visiting clusters of nodes and synchronizing their clocks as it comes within range of the head node. If the entire cluster cannot be seen by the robot at once, the robot will synchronize subsequent nodes as they come into range. In this simulation, clock synchronization is visually represented by flashing the nodes and robot at the end of their clock cycle. The starting point of each node's clock is initialized randomly. Therefore, the synchronization algorithm will have completed when all nodes share the same flash sequence as the robot.

4. Evaluation

Network convergence only occurs once the robot has visited all clusters at least once. Clusters must contain two or more nodes. Algorithm performance is highly dependent on cluster size and the number of reference packets sent. While small clusters are largely unstressed by clock synchronization, large clusters may require a reduction in the number of reference packets sent by the robot, thereby reducing the accuracy of the algorithm. While there may potentially be a large number of reference packets sent, actual processing time is quite manageable, even for modestly powerful nodes. Nodes need only perform simple arithmetic and store a lookup table and need not perform the expensive operation of changing the local clock. The only point at which clock synchronization occurs is when the robot is in range of a particular cluster. At other times, the node is free to collect data or sleep, thereby making this algorithm quite energy efficient. Because the robot is constantly in motion, it is imperative that clock synchronization and data transfer are completed before the robot moves out of range. Therefore there is a definite constraint on the maximum cluster size a robot can effectively process based on robot and node transmission time, number of reference packets sent, robot speed, node cluster size, robot transmission range, and robot CPU processing ability. Despite these constraints, RBS is appropriate for most practical applications of DTNs with maximum cluster sizes of fewer than 20 nodes.

5. Conclusion

Applying RBS to delay tolerant networks using mobile robots for inter-cluster communication is an effective way to perform clock synchronization. RBS was found to have been especially effective when applied to networks with small cluster sizes. The implemented algorithm is quite energy efficient because nodes only perform clock synchronization when in range of the robot. At other times, nodes are free to perform other functions. When implementing a DTN, attention must be paid to the dynamics of the robot vis-à-vis the node clusters. The speed of the mobile robot must be such that there is sufficient time to perform clock synchronization and data transfer functions for all clusters. Depending on the application, the RBS algorithm may be adjusted to favor performance over clock accuracy.

[1] K.Fall, "A delay-tolerant network architecture for challenged internets," Intel Research Berkeley, Tech. Rep. IRB-TR-03-003, February 2003.
[2] J. Elson, L. Girod, and D. Estrin. "Fine-Grained Network Time Synchronization using Reference Broadcasts", *Proc. Fifth Symposium on Operating Systems Design and Implementation (OSDI 2002),* Vol 36, pp. 147–163, 2002.