An Efficient Multi Channel Query Scheduling In Wireless Sensor Networks

T.Saravanan,
Assistant Professor/CSE,
Selvam College of Technology, Namakkal.

Abstract- **In order to achieve high performance query services in wireless sensor networks, MDCQS (Mobile lightweight medium access control based Dynamic Conflict-free Query scheduling) a novel TDMA based MAC protocol used for dynamic conflict free scheduling with high query rate. Earlier this technique is used to achieve high data rate. By integrating MLMAC with DCQS makes the best use of the query performance through dynamic conflict-free transmission scheduling based on queries relating to time in wireless sensor networks. Then MLMAC accommodate to high workload without changing the transmission schedule. NS2 simulations demonstrate that MDCQS significantly outperforms a representative DCQS protocol in terms of query latency and throughput.**

***Keywords*- DCQS, MLMAC, sensor networks, Query scheduling.**

I. INTRODUCTION

Wireless sensor networks have recently come into prominence because they hold the potential to revolutionize many segments of our economy and life, from environmental monitoring and conservation, to manufacturing and business asset management, to automation in the transportation and health-care industries. The design, implementation, and operation of a sensor network requires the confluence of many disciplines, including signal processing, networking and protocols, embedded systems, information management, and distributed algorithms. Such networks are often deployed in resource-constrained environments, for instance with battery operated nodes running untethered [1][2]. These constraints dictate that sensor network problems are best approached in a holistic manner, by jointly considering the physical, networking, and application layers and making major design trade-offs across the layers. Consequently, for an emerging field such as sensor networks that involves a variety of different technologies, a student or practitioner often has to be versed in several disparate research areas before he or she can start to make contributions.

Low data rate applications on wireless sensor networks like habitat monitoring are used earlier that would not be suitable for high workloads [3]. Now, in the recent year's high data rate applications like structural health monitoring system have been used to achieve high performance query services [4]. The queries periodically collect data from the base station. The routing trees introduce precedence constraints while executing query instances among packet transmissions. DCQS establishes better performance than TDMA transmission scheduling technique designed for general purpose workloads [5].The scheduler runs on every node and makes scheduling decisions at runtime based on the start time and period of queries as exposed by the application and the plans constructed by the planner [6]. Some of the features of DCQS over TDMA are: 1) DCQS perform add/remove of queries without changing the transmission schedule. 2) DCQS accommodate to high workload better than the TDMA. 3) DCQS requires less memory requirement and minimum runtime whereas the TDMA requires high runtime. But the DCQS has the inefficient data collection, for that MDCQS that overcomes the problem mentioned earlier.

DCQS works as follows:

1. When a new query is submitted, DCQS identifies a plan for its execution. It is often the case that many queries can be executed using the same plan. When no plan may be reused, the planner constructs a plan for executing the query.
2. Next, the base station performs rate control to ensure that the total query rate remain within the maximum query rate under DCQS. If necessary, the rates of the queries are decreased proportionally to not exceed the maximum query rate.
3. The phase, period, and aggregation function of the query are disseminated to all nodes.
4. At runtime, the scheduler executes all query instances.

The hybrid protocols like Z-MAC: A Hybrid MAC for Wireless Sensor Networks and Funneling-MAC: A Localized Sink-Oriented MAC for Boosting Fidelity in Sensor Networks are used in improving the efficiency but it is impossible in real time query services [11][12]. So we

Manuscript received February 5, 2014
Manuscript revised February 20, 2014

integrate MLMAC with DCQS for the better efficiency under high workload.

The remainder of the paper is organized as follows: Section 2 describes the related work. Section 3 describes the query and network models. Section 4 details the protocol design of DCQS. Section 5 describes the performance evaluation. DCQS is compared with MDCQS and Section 6 concludes the paper.

II. RELATED WORK

LMAC: In [8][10] L.F.W. van Hoesel and P.J.M. Havinga introduced LMAC (lightweight medium access control), which is on a TDMA scheme. Time is divided into frames and slots. Each node reserves a slot in which it can send, this slot reoccurs every frame. Every Slot is used to send a control message followed by data payload.

Description	Size (bytes)
Identification	2
Current Slot Number	1
Occupied Slots	4
Distance to Gateway	1
Collision in Slot	1
Destination ID	2
Data Size (bytes)	1
Total	12

Fig. 1. Control message used in LMAC

Figure 1 shows the contents of a LMAC control message. Its total size amounts to 12 Bytes. It contains the identity of the sender and its slot number followed by the most important field Occupied Slots, which represents a Bitmask of Slots. An unused slot is represented by a 0 while a 1 represents an occupied one. Thus it is possible for every node to determine unoccupied slots by combining the control messages of its neighbors. This is done by performing a simple OR operation on the fields Occupied Slots of all received control messages. The distance to the Gateway is also transmitted, along with information of overheard collisions. Finally, the ID of the destination and the size of the data unit are given. The initialization of nodes is started by the gateway, which defines its own slot and is used for synchronization. After one frame, all direct neighbors of the gateway know its slot and choose their own ones. This information is transmitted to their neighbors who synchronize on these messages. After each frame, a new set of nodes with a higher hop distance from the gateway are synchronized until every node knows its slot. These slots only need to be locally unique, as the nodes only compete with others up to 2 hops distant. To conserve node energy, a node's transceiver is turned off for the remainder of the current slot when it is not addressed in the control message. As slots are computed only once in LMAC, this protocol is not suitable for mobile sensor networks, where nodes can enter and leave other nodes' radio neighborhood at any time [7].

CSMA/CA:

A network contention protocol that listens to a network in order to avoid collisions, unlike CSMA/CD that deals with network transmissions once collisions have been detected. CSMA/CA contributes to network traffic because, before any real data is transmitted, it has to broadcast a signal onto the network in order to listen for collision scenarios and to tell other devices not to broadcast.

As soon as a node receives a packet that is to be sent, it checks to be sure the channel is clear (no other node is transmitting at the time). If the channel is clear, then the packet is sent. If the channel is not clear, the node waits for a randomly chosen period of time, and then checks again to see if the channel is clear. This period of time is called the backoff factor, and is counted down by a backoff counter. If the channel is clear when the backoff counter reaches zero, the node transmits the packet. If the channel is not clear when the backoff counter reaches zero, the backoff factor is set again, and the process is repeated.

TDMA:

Time division multiple access is a channel access method for shared medium networks. It allows several users to share the same frequency channel by dividing the signal into different time slots. The users transmit in rapid succession, one after the other, each using its own time slot. This allows multiple stations to share the same transmission medium (e.g. radio frequency channel) while using only a part of its channel capacity. TDMA is used in the digital 2G cellular systems such as Global System for Mobile Communications (GSM), IS-136, Personal Digital Cellular (PDC) and iDEN, and in the Digital Enhanced Cordless Telecommunications (DECT) standard for portable phones. It is also used extensively in satellite systems, combat-net radio systems, and PON networks for upstream traffic from premises to the operator. For usage of Dynamic TDMA packet mode communication. TDMA is a type of Time-division multiplexing, with the special point that instead of having one transmitter connected to one receiver, there are multiple transmitters. In the case of the uplink from a mobile phone to a base station this becomes particularly difficult because the mobile phone can move around and vary the timing advance required to make its transmission match the gap in transmission from its peers.

A set of N users share the same radio channel, but each user only uses the channel during predetermined slots. A frame consists of N slots, one for each user.

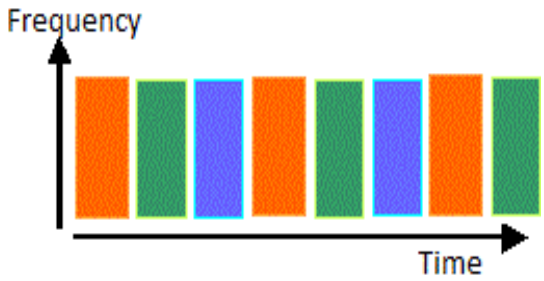

Fig. 2. CSMA/CA: Frequency vs. Time

Frames are repeated continuously. The transmit bandwidth is N times the bandwidth that would be needed to accommodate a single user. Thus the receiver can be built with broader filters, which are less expensive and smaller than those required for FDMA operation. Mostly, TDMA is combined with Time Division Duplex (TDD), in which transmission and reception do not occur simultaneously, but during different slots. This obviates the need for costly duplex filters.

In a downlink (base to mobile), TDMA is simple to implement: it is just a matter of multiplexing N user signals. In the uplink (mobile to base), TDMA is more difficult: the signals from all users have to be aligned in time. Often this is achieved through a feedback loop with timing information. Relatively fast power-up and power-off times are needed to avoid that signals from users interfere with signals in other slots.

III. SYSTEM MODELS

The System model comprises of the Query model and the network model.

Query model:

A query service works as follows: a user issues a query to a sensor network through a base station, which disseminates the query description to all nodes. To facilitate data aggregation, each nonleaf node waits to receive the data reports from its children, produces a new data report by aggregating its data with the children's data reports, and then sends it to its parent. We assume that there is a single routing tree that spans all nodes and it is used to execute all query instances. This assumption is consistent with the approach adopted by existing query services [4]. During the lifetime of the application, the user may issue new queries, delete queries, or change the period of existing queries. DCQS is designed to support such workload dynamics efficiently.

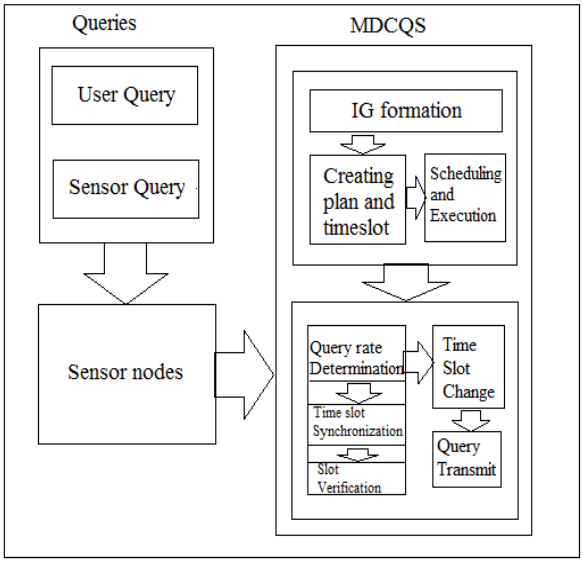

Fig. 3. Working process of MDCQS.

Network Model: The IC graph is based on the SNIR model. Empirical studies validating the accuracy of the SNIR model on 802.15.4 [9], [10], [8] and 802.11 [11] radios have already been performed. Moreover, MAC protocols which take advantage of the SNIR model have already been proposed and their performance validated empirically [12]. These previous studies on real hardware indicate that the IC graph is a realistic assumption. The IC graph was studied in [9], which presented a realistic approach for constructing IC graphs based on the SNIR model and RSS measurements. It should also be noted that the IC graph model adopted by our algorithm is more realistic than models often adopted by earlier scheduling algorithms (e.g., unit disk models and ignore interference).

Interference is inherently probabilistic and time-variant. It is important to note that the SNIR threshold controls the temporal stability of the IC graph. A conservative SNIR threshold would lead to a more stable IC graph at the cost of reduced throughput. We recognize that even when using conservative SNIR threshold, packets may be still corrupted asa result of intermittent interference. We address these issues through retransmissions and multipath routing.

The IC graph is built conservatively to improve temporal stability, over time the interference relations may change significantly. Detect changes in IC graph by monitoring the reliability of data collection over time. If the reliability falls below a user-set threshold then the IC graph is rebuilt. The IC graph also needs to be updated when nodes are added or removed.

IV. PROTOCOL DESIGN

Slot Synchronization:

The node that wants to send a packet first starts the synchronization. This removed the necessity to use the

field Distance to Gateway for synchronization. Even when it is not used for synchronization, the field Distance to Gateway could be used to support routing decisions in stationary sensor networks.

In mobile sensor networks however, this distance could not be determined only once and saved for further use, as the mobility of nodes will lead to a change in topology after a certain time. This time depends on the range of the transceivers and on the speed of the nodes of course, but eventually the change in topology will take place. The field Distance to Gateway is removed from the header of MLMAC. The fields Destination ID and Collision in Slot were not used either, because of the used hardware.

There was no way to shut down the transceivers and the radio module used a built in checksum to discard faulty packets. Thus, collisions could not be detected directly and this part of LMAC was not implemented. As this decision is based solely on the used radio modules, it could be revoked in the future, when a different platform is used.

Description	Size (bytes)
Identification	1
Slot number and Status	1
Occupied Slots	1-2
Identity of the Synchronization	1
Age of Synchronization	1
Total	5-6

Fig. 4. Control message used in MLMAC.

MLMACs control message format is shown in table II. This control message is quite different from the one used in LMAC. Due to our small sensor network, we reduced the field containing the identity of the sender to one Byte. Slot number and Status contains 5 Bit for the senders slot and 3 Bit for its status. The field Occupied Slots is used exactly as in LMAC, only its size is reduced.

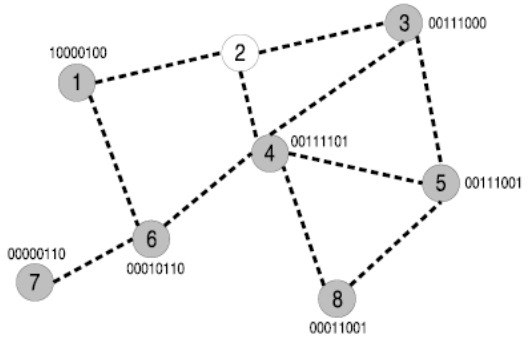

Fig. 5. Occupied slots as seen by each node.

The field Occupied Slots in the control message of a node contains the used slots of all its neighbors and itself. In the case of node 4 for example the slots 3, 4, 5, 6 and 8 would be marked as used, which results in a representation as 00111101. Note that in this example the third bit from the left represents the third slot. Figure 1 shows how slots are chosen with a simple example containing only eight nodes. In this example you can see that node 2 is not synchronized yet. It receives the control messages from its neighbors and combines them. 10000100 (from node 1) — 00111000 (from node 3) — 00111101 (from node 4) = 10111101 (seen on node 2).
This means that node 2 can choose between slots 2 and 7. If it chooses slot 2, its control message would contain the Bitmask 11110000 in the field Occupied Slots, as node 2 receives messages from nodes 1,3, and 4 and adds its own choice. If it chooses slot number 7 the field Occupied Slots would contain the Bitmask 10110010. Note that this method solves the hidden station problem. The number of slots can be chosen between 3 and 16 in our implementation, thus the size of Occupied Slots varies between 1 and 2 Byte. If more slots per frame are needed, the size of the field Occupied Slots grows. Thus far, the choice of slots is done in the same way as in LMAC, except for the fact that the synchronization started by a node rather than by the gateway.

Scheduling:

The network can experience unexpected link or node failures, traffic bursts, and topology changes, and there are no probabilistic assumptions describing these time varying events. Performance of our scheduling algorithm is compared against an ideal T-slot lookahead policy that can make optimal decisions based on knowledge up to T-slots into the future [17]. A simple non-anticipating algorithm that provides network throughput-utility that is arbitrarily close to (or better than) that of the T-slot lookahead policy, with a tradeoff in the worst case queue backlog kept at any queue. The same policy offers even stronger performance, closely matching that of an ideal infinite lookahead policy, when ergodic assumptions are imposed [13][14].

The scheduler improves throughput by overlapping the transmissions of multiple query instances concurrently while enforcing a conflict-free schedule.

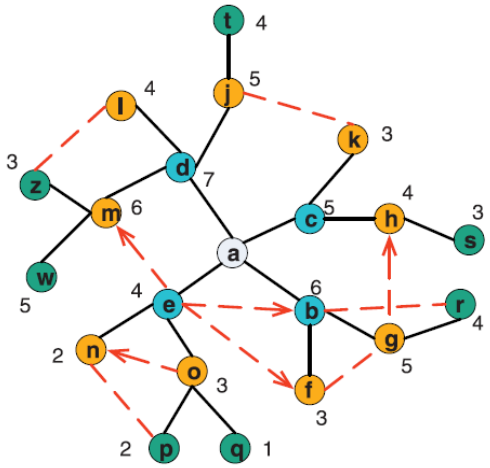

Fig. 6. Scheduling process in MDCQS

Each node employs a local scheduler that schedules the transmissions of all instances. The state of the scheduler includes: the start time and period of all queries, the plan's length, and the minimum interrelease time. Note that if all nodes have a consistent view of these parameters, they will construct independently the same schedule. The scheduler also knows the steps in which the host node transmits or receives. However, the scheduler does not need to know the specific steps in which any other nodes transmit or receive [15][16]. The scheduler has two FIFO queues: a run and a release queue. The release queue contains all instances released but not being executed. The run contains the instances to be executed in slot s. we first describe how to construct a global conflict-free schedule. We then present an efficient local scheduling algorithm. For clarity, in this section, we assume that all queries are executed according to the same plan, i.e., they belong to the same query class. Each instance is executed according to the plan of its query class.

Slot Changes:

The second difference is the fact that MLMAC stays adaptive even after slots are chosen. The last two fields of a control message are needed, because every node can start synchronization. Due to this fact, it is possible that two distant nodes start synchronization separately, as both of them assume that they are the first to send. Their neighbors would synchronize with them and increase the Age of Synchronization by one before retransmitting. In this case, two different synchronizations would be flooding the net and meet somewhere in between the two starter nodes. At this point, nodes would realize that some of their neighbors use a different synchronization by comparing the field Identity of the Synchronization. Now the field Age of Synchronization is compared. If the received value is equal to or higher than the local value saved in a node, this node becomes unsynchronized again and tries to find a new slot.

Due to the mobility of nodes, a node X may leave the radio range of node Y. Both nodes then realize that they do not receive any more control messages from each other and remove the other one from the neighbor list. When X moves into the radio range of another node Z which knows a different node W which uses the same slot as X, the control messages of X and W collide at Z. Therefore, Z does not receive any more control messages in that slot and marks it as unused. Nodes X and W receive the control message from Z and realize that there must have been a collision of control messages. After this, they give up their current slot and try to find a different one.

Process of MLMAC:

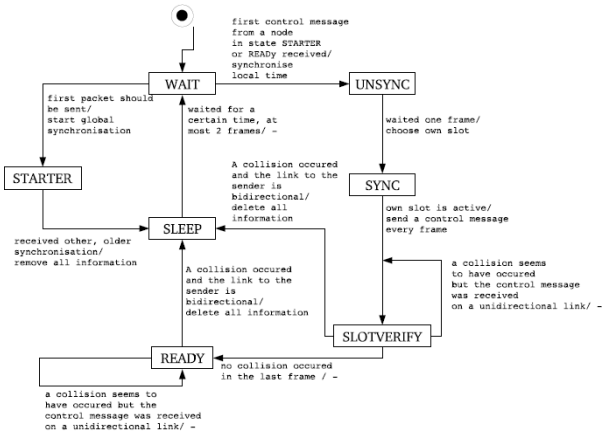

Fig. 7. Working process of MLMAC

MLMACs state machine is shown in figure 2. The rectangles represent states, the arrows transitions between them. The text on the arrows describes the necessary event for that transition and the action that is taken separated by a slash.

Initially, all nodes begin in the Wait-state. When they want to send a message without having received a control message yet, they change into the Starter-state. When only one node switches to starter, this is a stable state and the node remains there. If another node switched to the Starter-state earlier, this node gains knowledge of that fact after some time and switches to the Sleep-state from which it will return into the Wait-state after a certain time.

If a node received a control message from another node in Starter- or Ready-state while in the Wait-state it synchronizes its local time with that of the originator of the control message and switches into the Unsync-state. After waiting one frame to overhear all transmitted control messages and calculate used slots, it chooses its own one and transitions into the Sync-state. The next time, that node's slot is active, it starts to transmit its own control messages in every frame and changes to state Slotverify. This state is used to verify that no other node has chosen the same slot during the last frame. This would be

indicated by a collision of control messages in this slot and lead to a change into the Sleep-state.

As said before, the used hardware does not enable us to detect collisions directly. Rather, a node X that transmitted a control message can determine if a collision occurred by listening to its neighbors' control messages. If no collision occurred, the neighboring nodes have added X's slot to the field Occupied Slots in their control messages. Otherwise they did not. When X receives control messages containing its slot, it knows that no collision occurred because no other node has chosen the same slot. Therefore, it switches into the Ready-state.

Like the Starter-state, this is a stable state as long as no collision occurs. If a collision occurs, there must have been a mistake in the process of choosing slots, and the node returns into the Sleep- and finally into the Wait-state to start over again.

Note that MLMAC also distinguishes between collisions on unidirectional and bidirectional links. If a collision on a unidirectional link occurred on a node in Slotverify- or Ready-state, this node stays in the same state.

V. PERFORMANCE EVALUATION

We implemented the distributed version of MDCQS in NS2.We used a two-ray propagation model at the physical layer. Since DCQS is targeted at high data rate applications such as structural health monitoring we configured our simulator according to the DCQS settings. An overview of those deployments can be found in [17]. In our simulations, the bandwidth is 2 Mbps and the communication range is 125 m.

We set the slot size to 8.16 ms which is sufficient to transmit 2 KB of data. A simulation run takes 200s. Unless otherwise mentioned, each data point is the average of five runs and the

90 percent confidence intervals are plotted.

In the beginning of the simulation, the routing tree is constructed. The node closest to the center of the topology is selected as the base station. The base station initiates the construction of the routing tree by flooding setup requests. A node may receive multiple setup requests. The node selects as its parent the node that has the highest RSS among those with smaller depth than itself.

*Performance comparison when executing queries and their base rate is varied:*While comparing between wireless LAN and DCQS, the DCQS produces less latency but the data collection is not efficient. By the integration of MLMAC with DCQS produces less latency with efficiency.

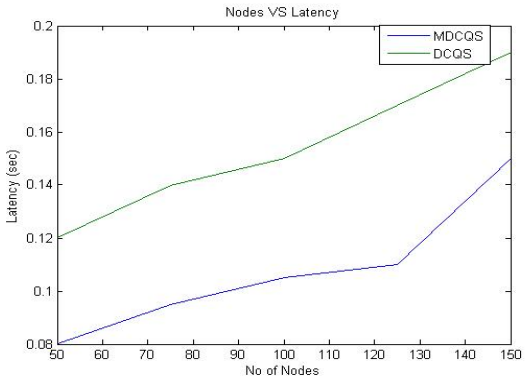

Fig. 8.Comparision of Query latency with DCQS and MDCQS.

Same as Query latency, the Query throughput while comparing between wireless LAN and DCQS, the DCQS produces high throughput but the data collection is not efficient. By the integration of MLMAC with DCQS produces high throughput with efficiency.

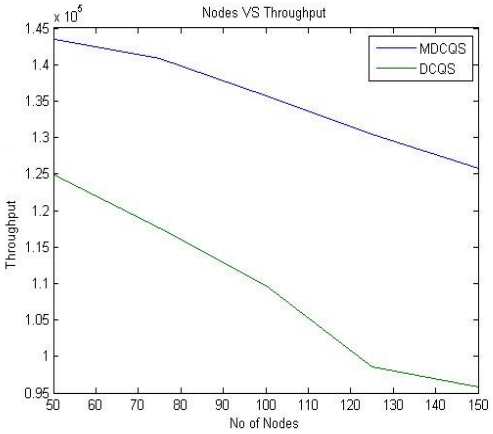

Fig. 9.Comparision of Query throughput with DCQS and MDCQS.

VI. CONCLUSION

This paper presents MLMAC based DCQS, a novel TDMA based MAC protocol transmission scheduling technique specifically designed for query services in wireless sensor networks. MDCQS features a planner and a scheduler. The planner reduces the latency while transmitting the queries. The scheduler increases the throughput. Mobile LMAC in each node can spontaneously establish a TDMA schedule on demand or join/leave existing schedules while nodes are moving. By that the high channel throughput can be achieved for mobile sensor nodes even in heavy load situations.

REFERENCES

[1] F. L. Lewis, “Wireless Sensor Networks”, This research was supported by ARO Research Grant DAAD 19-02-1-0366.

[2] R. Karnapke and J. Nolte. Copra, “A communication processing architecture for wireless sensor networks”, In Euro-Par 2006 Parallel Processing, pages 951–960. Springer, 2006.

[3] Mainwaring, D. Culler, J. Polastre, R. Szewczyk, and J. Anderson, “Wireless Sensor Networks for Habitat Monitoring,” Proc. First ACM Int’l Workshop Wireless Sensor Networks and Applications (WSNA), 2002.

[4] K. Chintalapudi, J. Paek, O. Gnawali, T. Fu, K. Dantu, J. Caffrey, R. Govindan, and E. Johnson, “Structural Damage Detection and Localization Using NETSHM,” Proc. Fifth Int’l Conf. Information Processing in Sensor Networks (IPSN), 2006.

[5] I. Rhee, A. Warrior, J. Min, and L. Xu, “DRAND: Distributed Randomized TDMA Scheduling for Wireless Ad Hoc Networks,” Proc. ACM MobiHoc, 2006.

[6] Octav Chipara, Chenyang Lu,” Dynamic Conflict-Free Transmission Scheduling for Sensor Network Queries” IEEE Trans. Mobile Computing, vol. 10, no. 5, May 2011.

[7] Stephan Mank, Reinhardt Karnapke and Joerg Nolte,” An adaptive TDMA based MAC Protocol for Mobile Wireless Sensor Networks”, International Conference on Sensor Technologies and Applications, 2007.

[8] P. H. S. Chatterjea, L.F.W. van Hoesel. Ai-lmac: An adaptive, information-centric and lightweight mac protocol for wireless sensor networks.

[9] Octav Chipara, Chenyang Lu, Gruia-Catalin Roman, “Real-time Query Scheduling forWireless Sensor Networks”, 28th IEEE International Real-Time Systems Symposium.

[10] L. van Hoesel and P. Havinga. A lightweight medium access protocol (lmac) for wireless sensor networks: Reducing preamble transmissions and transceiver state switches. In INSS, Japan, Jun 2004.

[11] I. Rhee, A. Warrier, M. Aia, and J. Min, “Z-MAC: A Hybrid MAC for Wireless Sensor Networks,” Proc. Third Int’l Conf. Embedded Networked Sensor Systems (SenSys), 2005.

[12] G.-S. Ahn, S.G. Hong, E. Miluzzo, A.T. Campbell, and F. Cuomo, “Funneling-MAC: A Localized, Sink-Oriented MAC for Boosting Fidelity in Sensor Networks,” Proc. Fourth Int’l Conf. Embedded Networked Sensor Systems (SenSys), 2006.

[13] R. Ramaswami and K.K. Parhi, “Distributed Scheduling of Broadcasts in a Radio Network,” Proc. IEEE INFOCOM, 1989.

[14] L. Bao and J.J. Garcia-Luna-Aceves, “A New Approach to Channel Access Scheduling for Ad Hoc Networks,” Proc. ACM MobiCom, 2001.

[15] V. Rajendran, K. Obraczka, and J.J. Garcia-Luna-Aceves, “Energy- Efficient Collision-Free Medium Access Control for Wireless Sensor Networks,” Proc. First Int’l Conf. Embedded Networked Sensor Systems (SenSys), 2003.

[16] I. Chatzigiannakis, S. Nikoletseas, and P. G. Spirakis. Efficient and robust protocols for local detection and propagation in smart dust networks. Mob. Netw. Appl., 10(1-2):133–149, 2005.

[17] M. Maroti, B. Kusy, G. Simon, and A. Ledeczi, “The Flooding Time Synchronization Protocol,” Proc. Second Int’l Conf. Embedded Networked Sensor Systems (SenSys), 2004.