This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE ICC 2010 proceedings

Scheduled Access Using the IEEE 802.15.4 Guaranteed Time Slots

M. Takaffoli, E. Elmallah
Department of Computing Science
University of Alberta
Edmonton, T6G 2E8, Canada
E-Mail: {takaffol,ehab}@cs.ualberta.ca

W. Moussa
Department of Mechanical Engineering
University of Alberta
Edmonton, T6G 2G8, Canada
E-mail: walied.moussa@ualberta.ca

***Abstract*—In this paper we consider the design of IEEE 802.15.4 wireless sensor networks (WSNs) where nodes belong to different priority classes. Each class is characterized by a specified average data transmission rate requirement, and the overall network forms a multi-level tree. We devise a framework for constructing TDMA schedules for solving the underlying rate differentiation problem using the GTS facility of the standard. Our framework defines a class of schedules, called *flow balanced* schedules, that are efficient in terms of the delay incurred by packets transmitted to the sink node, and the number of packets queued in each node. We identify two useful optimization aspects that help in constructing schedules with short cycle length. We then outline an algorithm, called GTS-TDMA, which integrates two algorithms that take advantage of the identified optimization aspects, and present simulation results that show the performance gains of the devised algorithm.**

I. Introduction

Many continuous monitoring applications of wireless sensor networks (WSNs) require the design of a network where the sensor nodes belong to different priority classes. In such networks, nodes in higher priority classes are required to collect and send data more frequently than nodes in lower priority classes. In some applications, such as networks for health monitoring of human and mechanical systems, the distances between the sensor nodes in a field justify the use of multi-level tree networks. Currently, the IEEE 802.15.4 standard for low-rate wireless personal area networks (LR-WPANs) [1] is the platform of choice for many commercial wireless sensor products. The standard specifies the physical (PHY) and medium access control (MAC) layers for such devices. The MAC layer defines an efficient CSMA-CA protocol. However, the protocol is not adequate for providing the required rate differentiation for such continuous monitoring applications. The standard, however, defines the Guaranteed Time Slots (GTSs) mechanism for serving devices that require dedicated bandwidth or low latency transmission.

Our main contribution in the paper is the development of an algorithm that utilizes the GTS feature of the standard to construct TDMA schedules for multi-level trees; the constructed schedules satisfy (whenever possible) the rate differentiation requirements of such networks. To position our contribution among existing IEEE 802.15.4 related results, we first remark that many of the existing work in the area can be roughly classified as either: (a) work on analyzing the performance of the protocols and procedures defined in the standard, (b) work on extending the standard to handle priority traffic, or (c) work on extending the standard to provide rate and/or delay differentiation among classes of nodes in a WSN.

Examples of work in the first direction include the simulation based studies of [2], [3], and the analytical modelling of the standard's CSMA-CA protocol under saturation and unsaturation conditions (e.g., [4]–[7]). Work in the second direction considers serving high priority packets in event monitoring networks. Examples of such work include non-GTS based approaches (e.g., [8], [9]), and GTS-based approaches (e.g., [10], [11]). Work in the third direction include [12] where the authors propose to extend the standard's CSMA-CA protocol by allowing each priority class to have its own backoff exponent (BE), and contention window (CW) values to achieve rate and delay differentiation among different classes of nodes.

None of the above work, however, addresses the construction of TDMA schedules as done here. The performance of TDMA schedules is well expected to exceed that achieved by any CSMA-CA based approach even for star networks with few nodes. To present our contribution, we review in the next section the main constraints regulating the use of GTSs in the standard; the review is followed by an outline of the main design aspects of the proposed scheduling algorithm.

II. Overview of the IEEE 802.15.4 GTSs

In this section, we summarize the main aspects and restrictions pertaining to the use of GTSs in the standard (see, e.g., [13], [14]). We first recall that in a beacon-enabled operation of a WSN, both the PAN coordinator and other relay nodes (acting as coordinators) can transmit beacon frames. Each coordinator can use a superframe structure composed of up to three types of periods: Contention Access Period (CAP), Contention Free Period (CFP), and inactive period.

The IEEE standard requires that the active period of each superframe, known as the superframe duration (SD), of all coordinators in the network to have equal length. In contrast to the above equality constraint on the length of the SD intervals, the length of the beacon intervals (i.e., the lengths of the superframes) of different coordinators need not be equal.

978-1-4244-6404-3/10/$26.00 ©2010 IEEE

The active period of each superframe is divided into a fixed number of equal length slots that we denote by $n_{SD\_slots\_max}$. In the standard, $n_{SD\_slots\_max} = 16$ slots. The beacon frame always starts at the beginning of the first time slot. In each superframe, there can be up to a fixed number, denoted n_{GTS_max} of GTSs. In the standard, $n_{GTS\_max} = 7$. Each GTS can occupy one or more SD slots, e.g., the 16 slots of an SD interval can be divided into 2 GTSs serving 2 different nodes. The first GTS may occupy 15 slots, and the second GTS may occupy the remaining slot.

The beacon frame issued at the beginning of an SD interval determines the number of GTSs in the interval (at most n_{GTS_max}). In addition, the frame carries the following information: the node served by each allocated GTS, the start slot of each GTS, the length of each GTS, and the data transfer direction (uplink or downlink) of all GTSs in the SD.

In our design, nodes do not request GTSs, rather the devised algorithm determines a GTS allocation scheme. In a constructed schedule, each active period in each beacon interval contains the minimum allowed CAP, defined by the $aMinCAPLength$ period ($= 440$ symbols) in the standard, and the rest of the active period contains a CFP only. The design sets the length of the active periods of all superframes to the same value, as required by the standard. A constructed schedule is specified by describing its *cycle* structure that is repeated over the time (Fig. 2 illustrates an example schedule cycle).

III. An Example Network

To present our construction, we use the following example. We recall that two links have a *primary* interference if they share a node. Two links have a *secondary* interference if they do not share a node but some node on the first link lies within the interference range of some node on the second link.

Example 1: Fig. 1 illustrates a weighted tree with 6 coordinators, labelled c_0 to c_5, where c_0 is the PAN coordinator (the sink node), and c_1 through c_5 are relay nodes that don't generate packets of their own. The leaf nodes s_1 to s_{10} are sensor nodes that generate traffic. With each leaf node in the diagram there is an associated number of packets. We interpret that a leaf node periodically generates this number of packets every schedule cycle. For example, sensor node s_1 generates 3 packets every schedule cycle. Similar to the above property, with each non-leaf node in the diagram there is an associated number of packets that the node should periodically collect from its children. For example, relay c_3 should periodically collect 4 packets from relays c_4 and c_5. The solid lines are communication links. Not shown in the figure are the primary and secondary interference relations implied by the links of the tree. For example, the secondary interference relation between links (c_0, c_3) and (c_2, s_4) is not shown. Additional interference edges that are assumed to exist appear as dashed lines. ■

The weighted tree in the example forms an instance of the schedule construction problem. A valid solution is a schedule that satisfies the equality constraint on the SD intervals identified in the previous section, and serves each node x by transmitting its associated number $w(x)$ of packets in each cycle. If the cycle length of the constructed schedule is τ seconds then the data rate enjoyed by each node under the constructed schedule is $w(x)/\tau$ packets/second. Thus, maximizing the data rate enjoyed by each node amounts to minimizing the cycle length of the constructed schedule.

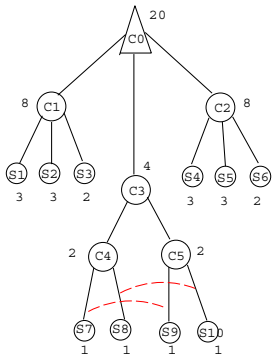


Fig. 1: An example WSN.

IV. Flow Balanced Schedules

We now introduce the concept of *flow balanced* TDMA schedules. A concrete example of the concept is presented in the next section.

Definition 1: Given a weighted tree, a schedule that satisfies the IEEE 802.15.4 requirements on GTSs is called flow balanced if (1) in each cycle, each internal node collects from its children exactly the specified number of packets associated with the node, and (2) transmissions that are scheduled in one time slot are interference-free. ■

The structure of flow balanced schedules allows us to prove the following result.

Theorem 1: Let S be a flow balanced schedule for a tree network T. Denote the length of the beacon interval of the sink node by BI_{sink}. Then

1) If x is a node at distance d hops away from the sink node in T then the time required for all packets transmitted by x during one cycle to reach the sink is bounded by $d.BI_{sink}$.
2) If node x is scheduled to transmit $w(x)$ packets in each cycle then x needs to buffer at most $2.w(x)$ packets at any instant.

Proof. Part 1 follows since each internal node collects from its children all packets that need to be transmitted in one cycle. Thus, the schedule guarantees that all packets transmitted by a node during one cycle advance one hop towards the sink at the end of each cycle. Part 2 follows since at any instant, each node is required to buffer the incoming packets (if the node is a relay) and the outgoing packets. Thus, no more than $2.w(x)$ packets need to be buffered at any instant. ■

We remark that a flow balanced schedule can leave many slots unused (since it is only required to serve exactly the specified number of packets associated with each node). Nevertheless, by part (2) of the above theorem, the schedule ensures that no node becomes congested at any point of time. We also remark that packets generated by leaf nodes

(or received by non-sink coordinators) in one cycle may be forwarded up in the tree during the next cycle.

V. An Example Flow Balanced Schedule

We now present a flow balanced schedule for Example 1. The constructed schedule incorporates two optimization aspects, as mentioned below.

Example 2: Consider the weighted tree T of Example 1. For the purpose of illustrating the key features of the constructed schedule, we assume for simplicity that $n_{SD\_slots\_max} = 6$, $n_{GTS\_max} = 2$, and each slot accommodates one packet. Fig. 2 illustrates a valid flow balanced schedule for T. In particular, one may verify that (a) each SD has exactly $n_{SD\_slots\_max} = 6$ slots, (b) each SD is partitioned into at most $n_{GTS\_max} = 2$ GTSs (the nodes served by an SD are written below the SD in the diagram), (c) each internal node (a coordinator) collects exactly its associated number of packets in each cycle, and (d) transmissions that occur in one time slot are interference free. In the figure, each cycle is composed of 4 strides. The width of each stride equals the length of the beacon interval of the PAN coordinator. ■

Below, the notation BI_x refers to the length of the beacon interval of node x. We now draw the following remarks. First, the SDs of coordinators c_1 and c_2 are scheduled in *parallel*. That is, in each stride where the SDs of c_1 and c_2 occur simultaneously, they start form the same offset relative to the stride they occur in. In the example, $BI_1 = BI_2 = 2BI_0$.

Second, the SDs of coordinators c_3, c_4, and c_5 are scheduled in *alternation*. That is, in each stride, at most one SD of at most one of the three coordinators occurs in the stride, and all SDs start at the same offset relative to their respective strides. In the example, $BI_3 = BI_4 = BI_5 = 4BI_0$. In our development, we assume that the sink can afford to send as many beacon frames in one cycle as any other coordinator. The assumption allows the width of each stride in a typical cycle of the constructed schedule to be equal to the length of the beacon interval of the PAN coordinator, as in Fig. 2.

VI. Optimized Flow Balanced Schedules

In this section we define scheduling SDs in parallel and alternation formally, and present methods for recognizing each case. Below, T denotes a given weighted tree, S denotes a valid schedule serving the packets in T, and C denotes a given set of two, or more, coordinators.

A. Scheduling SDs in Parallel

Definition 2: The set of SDs of the coordinators in C are scheduled in parallel if (1) some strides of the schedule S contains two, or more, SDs belonging to two (or more) coordinators in C, (2) all SDs have the same offset relative to the beginning of their respective strides, and (3) each cycle contains all SDs required by the coordinators in C. ■

A sufficient condition for the SDs of C to be schedulable in parallel can be stated using the following definition.

Definition 3: For each coordinator c_x in C, denote by E_x the set of communication links between x and its children in the tree T. The conflict graph of the coordinators in T, denoted $G_{cc}(T)$, is a graph on the set of coordinators of T where two coordinators c_x and c_y are adjacent in the conflict graph if at least one link in E_x interferes with at least one link in E_y. ■

As can be verified, the coordinators in C can be scheduled in parallel if no two coordinators are adjacent to each other in the conflict graph $G_{cc}(T)$. We utilize a greedy algorithm for determining coordinators that can be scheduled in parallel.

B. Scheduling SDs in Alternation

Definition 4: The set of SDs of the coordinators in C are scheduled in alternation if (1) each stride of the cycle has at most one SD of at most one of the coordinators in C, (2) all SDs have the same offset relative to the beginning of their respective strides, and (3) each cycle contains all SDs required by the coordinators in C. ■

We now present the basic ingredients of an algorithm for deciding whether a given set of coordinators can be scheduled in alternation, given the following input parameters:

- $n_{strides}$: the number of strides in a typical cycle of the schedule under construction.
- $(n_i : i = 1, 2, \cdots, k)$: n_i is the number of SDs required by the ith coordinator.
- $offset$: the offset at which each of the alternating SDs is scheduled to start relative to the beginning of its respective stride.

The main ingredients follow. (1) If the ith coordinator considered by the algorithm requires n_i SDs in each cycle to satisfy its requests, then the algorithm must allocate at least n_i' SDs, $n_i' \geq n_i$, to serve that particular coordinator. (2) The allocation of these n_i' SDs must be evenly spaced within a cycle so that the SDs appear cyclically in the schedule. That is, given that the cycle has $n_{strides}$ strides, the distance $\ell_i = n_{strides}/n_i'$ must be an integer. (3) The use of the distance ℓ_i makes the length of the beacon interval of the ith coordinator, denoted BI_i, equal to $\ell_i.BI_0$, where BI_0 is the length of the beacon interval of the PAN coordinator. (4) ℓ_i must be a power of 2 so as to give a valid length of the BI_i interval. Based on the above observations, we implemented an iterative algorithm that determines whether a given coordinator can be scheduled in alternation along with a given set of coordinators

C. The Main Algorithm

Fig. 3 presents the overall algorithm. The main loop in step 1 iterates over all possible values of the SO parameter in the IEEE standard (form 0 to 14). Each iteration tries to construct a schedule with a short cycle length. At the end of each iteration, step 1.5 keeps the best valid schedule found thus far. For a given SO value (and hence a given SD slot length) steps 1.1 and 1.2 compute the largest number of SDs required by each node, and set the number of strides in the constructed schedule ($n_{strides}$) to this value. Step 1.3 initializes the schedule by assigning the SDs of the PAN coordinator (the sink) to the beginning of each stride (i.e., $offset = 0$). The loop in step 1.4 iterates over all non-sink coordinators sequentially level by level from top to bottom.

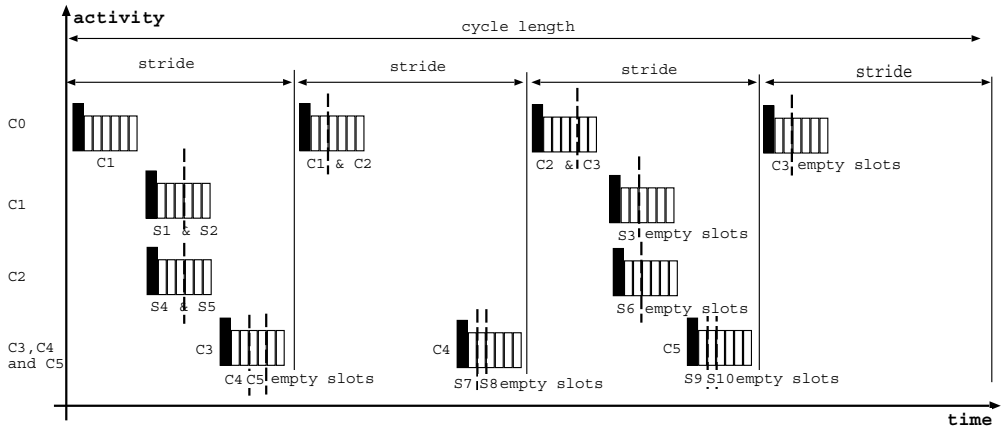


Fig. 2: An example flow balanced schedule

Algorithm GTS-TDMA
Input: a weighted tree T and its conflict graph $G_{cc}(T)$
Output: a valid schedule S (if possible)

1. for $SO = 0, 1, \cdots, 14$ {
   - 1.1 Traverse T from bottom to top. Use the current SO value to compute for each node x its required number of SDs.
   - 1.2 Set $n_{strides}$ to the largest number of SDs required by any node in the tree.
   - 1.3 Allocate $n_{strides}$ SDs to the sink, each SD starts at $offset = 0$ in its respective stride.
   - 1.4 Let $c_1, c_2, \cdots$ be the non-sink coordinators in T when the tree is traversed from top to bottom.
     for $i = 1, 2, \cdots$ {
     - 1.4.1 Determine if c_i can be scheduled in parallel along with a subset of coordinators that have been already processed. If such a subset exists, schedule c_i accordingly.
     - 1.4.2 Repeat the above step with respect to scheduling in alternation.
     - 1.4.3 If neither parallelism nor alternation is possible then increment the offset value and allocate SDs to the coordinator so as to maximize the length of its beacon interval.

     }
   - 1.5 If the length of the beacon interval of the PAN coordinator is valid (i.e., the corresponding $BO \leq 14$) then store the schedule if it has the shortest cycle length encountered thus far.

}

Fig. 3: Pseudo-code of Algorithm GTS-TDMA

Each coordinator is checked to see if it can be scheduled in parallel, or alternation along with some group of coordinators that the algorithm has already processed. Checking for parallel scheduling uses the precomputed coordinator conflict graph $G_{cc}(T)$.

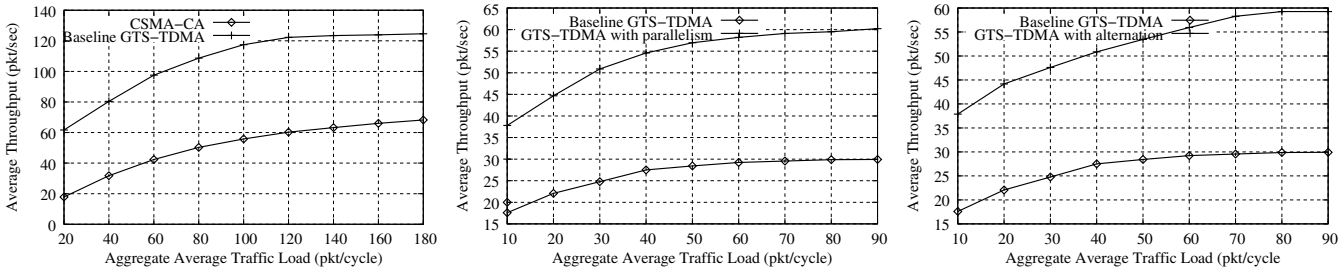
Running Time. If n is the number of nodes in the tree T then the overall running time of the algorithm is determined by the execution of steps 1.4.1 and 1.4.2 which is $O(n^2)$.

VII. Performance Evaluation

We investigate the performance of three versions of the algorithm: (a) a baseline version that does not use either scheduling SDs in parallel nor alternation, (b) a baseline version that uses scheduling SDs in parallel only, and (c) a baseline version that uses scheduling SDs in alternation only. Our objective is to gain insight into the impact of using each optimization method. Performance of the baseline version is studied against a modified version of the standard's CSMA-CA algorithm [12] that assigns specific backoff exponent (BE) and contention window (CW) to each priority class so as to achieve rate differentiation among various classes of nodes. We implemented the modifications in QualNet 4.0 network simulator [15]. Below, we present only comparisons of the baseline algorithm when the WSN has only one class of sensors. Hence, the comparison uses the standard's CSMA-CA algorithm.

Simulation Parameters. Traffic from the sensor nodes to their serving relay nodes is made of fixed size packets of length $L = 1376$ bits each (120 bytes data plus 52 bytes header). To evaluate the GTS-TDMA algorithms, each leaf node generates packets according to the Poisson distribution with a specified average number of packets per cycle. The x-axes in Figures 4a to 4c show the aggregate average number of packets generated by tree nodes per cycle. The algorithms use such averages to compute a schedule and determine its cycle length. The y-axes in Figures 4a to 4c show the observed average throughput (packets/sec) of packets delivered to the sink. Thus, schedules with short cycle length result in higher average throughput. Simulation of the CSMA-CA protocol generates Poisson traffic with specified average packets/sec. In addition, for CSMA-CA, the simulation sets the length of the active part of each beacon interval using a superframe order $SO = 3$ so that each slot of the active period accommodates one packet. The network operates at 250 Kbps (the standard channel rate when operating in the 2.4 GHz ISM band). Simulation duration of the CSMA-CA protocol is set to 3000 seconds of which the first 1000 seconds allow the nodes to associate with the PAN coordinator, and the remaining time is used for sending application traffic.

Baseline GTS-TDMA versus CSMA-CA. Fig. 4a illustrates the average throughput achieved by our baseline GTS-TDMA algorithm and the standard's CSMA-CA algorithm. We use a star network with $n = 40$ sensor nodes connected to the sink



(a) Throughput of Baseline GTS-TDMA Algorithm (b) Effect of Scheduling SDs in Parallel (c) Effect of Scheduling SDs in Alternation

Fig. 4: Simulation results

node. Each sensor node is placed $d = 10$ meters away from the sink. The transmission radius R_T of each node is set to 10 meters. The aggregate traffic generated by the n nodes and destined to the sink node is Poisson with average in the interval $[40, 180]$ packets/cycle (for CSMA-CA traffic, the cycle length is one second). For such scenarios, Fig. 4a indicates that the algorithm achieves approximately twice the throughput of the CSMA-CA algorithm.

Effect of Scheduling SDs in Parallel. Here, the WSN forms a tree of depth 2, where the sink is connected to $n_1 = 2$ relay nodes (coordinators) at level 1 of the tree. Each relay node is placed $d_1 = 15$ meters away from the sink. The two relays and the sink lie on a straight line. Each relay is connected to $n_2 = 20$ sensor nodes at level 2 of the tree. Each sensor node is placed $d_2 = 10$ meters away from its serving relay. The transmission radius of each node is set to $R_T = 15$ meters. The aggregate traffic generated by all sensor nodes and destined to the sink is Poisson with average in the interval $[10, 90]$ packets/cycle. The results shown in Fig. 4b indicate a notable increase in the achieved throughput by the optimized algorithm. This increase is attributed to the ability of the algorithm to schedule the SDs of the two relays at level 1 in parallel; thus, reducing the schedule cycle length.

Effect of Scheduling SDs in Alternation. We use the same tree network of depth 2, and offered traffic load as above. However, we set $R_T = 50$ meters for each node so that scheduling SDs in parallel cannot be used. The results shown in Fig. 4c indicates a notable increase in the achieved throughput by the optimized algorithm. The increase is attributed to the ability of the algorithm to schedule the SDs of the two relays at level 1 in alternation. where each relay is required to collect half the packets destined to the sink.

VIII. Concluding Remarks

In this paper we devise a framework that utilizes the IEEE 802.15.4 GTS facility to realize TDMA schedules that provide rate differentiation among classes of sensor nodes in multi-level tree networks. Our framework implements two optimization methods to synthesize schedules with short cycle length. The two methods are referred to as scheduling SDs in parallel, and scheduling SDs in alternation. For future work we propose investigating rate differentiation problems in mobile WSNs where some nodes may be required to associate with different relay nodes as they move along a given path. Finally, we remark that the IEEE 802.15 WPAN Task Group 4e is currently working on improving the standard to better serve applications with QoS requirements. Evaluating the newly proposed mechanisms is a topic of interest for future work.

Acknowledgment. This research is supported by NSERC Canada. We thank the reviewers for their valuable comments.

References

[1] IEEE Computer Society, "PART 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (LR-WPANs)," *IEEE P802.15.4a/D5*, 2006.

[2] G. Lu, B. Krishnamachari, and C. S. Raghavendra, "Performance evaluation of the IEEE 802.15.4 MAC for low-rate low-power wireless networks," *IEEE International Performance Computing and Communications Conference (IPCCC)*, April 2004.

[3] J. Zheng and M. J. Lee, "A Comprehensive Performance Study of IEEE 802.15.4," *IEEE Press Book*, 2004.

[4] J. Misic and V. B. Misic, "Access delay for nodes with finite buffers in IEEE 802.15.4 beacon enabled PAN with uplink transmissions," *Computer Communications*, vol. 28, no. 10, pp. 1152 – 1166, 2005.

[5] T. R. Park, T. H. Kim, J. Y. Choi, S. Choi, and W. H. Kwon, "Throughput and energy consumption analysis of IEEE 802.15.4 slotted CSMA/CA," *IEEE Electronics Letters*, vol. 41, pp. 1017 – 1019, 2005.

[6] J. V. Misic, S. Shafi, and V. B. Misic, "Performance limitations of the MAC layer in 802.15.4 low rate WPAN," *Computer Communications*, vol. 29, no. 13-14, pp. 2534–2541, 2006.

[7] I. Ramachandran, A. K. Das, and S. Roy, "Analysis of the contention access period of IEEE 802.15.4 MAC," *ACM Transactions on Sensor Networks (TOSN)*, vol. 3, March 2007.

[8] T. Kim and D. Lee and J. Ahn and S. Choi, "Priority toning strategy for fast emergency notification in IEEE 802.15.4 LR-WPAN," *in Proceedings of the 15th Joint Conference on Communications & Information (JCCI)*, April 2005.

[9] A. Koubaa, M. Alves, B. Nefzi, and Y. Q. Song, "Improving the IEEE 802.15.4 slotted CSMA/CA MAC for time-critical events in wireless sensor networks," in *Workshop on Real Time Networks (RTN)*, July 2006.

[10] Y.-K. Huang, A.-C. Pang, and H.-N. Hung, "An adaptive GTS allocation scheme for IEEE 802.15.4," *IEEE Transactions on Parallel and Distributed Systems*, vol. 19, no. 5, pp. 641–651, 2008.

[11] C. Na, Y. Yang, and A. Mishra, "An optimal GTS scheduling algorithm for time-sensitive transactions in IEEE 802.15.4 networks," *Computer Networks*, vol. 52, no. 13, pp. 2543 – 2557, 2008.

[12] E. Kim, M. Kim, S. Youm, S. Choi, and C. Kang, "Priority-based service differentiation scheme for IEEE 802.15.4 sensor networks," *International Journal of Electronics and Communications(AEU)*, vol. 61, pp. 69–81, February 2007.

[13] E. Callaway, P. Gorday, L. Hester, J. Gutierrez, M. Naeve, B. Heile, and V. Bahl, "Home networking with IEEE 802.15.4: a developing standard for low-rate wireless personal area networks," *IEEE Communications Magazine*, vol. 40, pp. 70–77, August 2002.

[14] S. Farahani, *ZigBee Wireless Networks and Transceivers*. Oxford, UK: Elsevier Inc, 2008.

[15] Scalable Networks Technologies, "Qualnet 4.0." http://www.scalable-networks.com/.