Technical Report

Energy efficient scheduling for cluster-tree Wireless Sensor Networks with time-bounded data flows: application to IEEE 802.15.4/ZigBee

Zdenek Hanzalek

Petr Jurcik

HURRAY-TR-100501

Version:

Date: 05-05-2010

Energy efficient scheduling for cluster-tree Wireless Sensor Networks with time-bounded data flows: application to IEEE 802.15.4/ZigBee

Zdenek Hanzalek, Petr Jurcik

IPP-HURRAY!

Polytechnic Institute of Porto (ISEP-IPP)

Rua Dr. António Bernardino de Almeida, 431

4200-072 Porto

Portugal

Tel.: +351.22.8340509, Fax: +351.22.8340509

E-mail:

http://www.hurray.isep.ipp.pt

Abstract

Cluster scheduling and collision avoidance are crucial issues in large-scale cluster-tree Wireless Sensor Networks (WSNs). The paper presents a methodology that provides a Time Division Cluster Scheduling (TDCS) mechanism based on the cyclic extension of RCPS/TC (Resource Constrained Project Scheduling with Temporal Constraints) problem for a cluster-tree WSN, assuming bounded communication errors. The objective is to meet all end-to-end deadlines of a predefined set of time-bounded data flows while minimizing the energy consumption of the nodes by setting the TDCS period as long as possible. Sinceeach cluster is active only once during the period, the end-to-end delay of a given flow may span over several periods when there are the flows with opposite direction. The scheduling tool enables system designers to efficiently configure all required parameters of the IEEE 802.15.4/ZigBee beacon-enabled cluster-tree WSNs in the network design time. The performance evaluation of thescheduling tool shows that the problems with dozens of nodes can be solved while using optimal solvers.

Energy efficient scheduling for cluster-tree Wireless Sensor Networks with time-bounded data flows

Zdeněk Hanzálek* and Petr Jurčík*†
*Department of Control Engineering, Faculty of Electrical Engineering
Czech Technical University in Prague, Czech Republic
†CISTER/ISEP, Polytechnic Institute of Porto, Portugal
{hanzalek, jurcikp}@fel.cvut.cz

Abstract

Cluster scheduling and collision avoidance are crucial issues in large-scale cluster-tree Wireless Sensor Networks (WSNs). The paper presents a methodology that provides a Time Division Cluster Scheduling (TDCS) mechanism based on the cyclic extension of RCPS/TC (Resource Constrained Project Scheduling with Temporal Constraints) problem for a cluster-tree WSN, assuming bounded communication errors. The objective is to meet all end-to-end deadlines of a predefined set of time-bounded data flows while minimizing the energy consumption of the nodes by setting the TDCS period as long as possible. Since each cluster is active only once during the period, the end-to-end delay of a given flow may span over several periods when there are the flows with opposite direction. The scheduling tool enables system designers to efficiently configure all required parameters of the IEEE 802.15.4/ZigBee beacon-enabled cluster-tree WSNs in the network design time. The performance evaluation of the scheduling tool shows that the problems with dozens of nodes can be solved while using optimal solvers.

Keywords

cluster-tree, TDMA, IEEE 802.15.4, ZigBee, cyclic scheduling, collision avoidance, energy efficiency, real-time

I. Introduction

To improve the functionality and efficiency of industrial monitoring and control systems (e.g. visual surveillance for automatic object detection [1]), the industries are looking toward Wireless Sensor Networks (WSNs) to provide sensing and actuating in hazardous and previously hard-to-reach areas [2]–[4] where very specialized and costly procedures must be adhered. Timeliness and energy efficiency are important requirements to be fulfilled in these systems. The use of WSNs in control applications has many advantages compared to wired solutions, which are dominant at the moment. WSNs may be installed for a fraction of the cost and time of an existing wired network. Since the wireless nodes are usually battery-powered, the network can be effectively used in environments where electricity is not available or some level of mobility is required (e.g. rotating parts of machines).

WSNs can be classified into two types, infrastructure-based and ad hoc (infrastructure-less) networks. The former is less flexible since it relies on the pre-deployed and structured topology, but provides better support of predictable performance guarantees. To ensure collision-free access to the shared wireless medium, the infrastructure-based networks employ the deterministic routing protocols (e.g. ZigBee's hierarchical routing [5]) and contention-free Medium Access Control (MAC) protocols. On the other hand, ad hoc network provides good flexibility to adaptive network changes, but at the cost of unpredictable performance. The non-deterministic routing protocols [6] and contention-based MAC protocols cause unpredictable performance bounds. Hence, when high predictability of performance guarantees is the objective, it is suitable to rely on infrastructure-based WSNs such as cluster-tree [7]. The cluster-tree network is supported by the IEEE 802.15.4/ZigBee [5], [8] standards which are leading technologies for low-cost, low-power and low-rate WSNs.

In this paper, we assume that the cluster-tree network has already been set up, i.e. each node knows its parent and child nodes (e.g using the ZigBee's tree addressing scheme [5]). All nodes may have sensing or/and actuating capabilities, therefore they can be sources or/and sinks of data flows. We consider periodic time-bounded flows (each given by parameters such as sink node, source nodes and end-to-end deadlines) which must be known in network design time.

In cluster-tree WSNs, the flows traverse different clusters on their routing paths from the source nodes to the sink nodes. The clusters may have collisions when they are in the neighborhood. Thus, the key problem solved in this paper is to find a periodic schedule which specifies when the clusters are active while avoiding possible inter-cluster collisions and meeting all flows' end-to-end deadlines. The fact that the cluster is active only once during the schedule period [5] leads to so called cyclic behavior of periodic schedule (i.e. time between the instant when a source sends the message and the instant when the sink receives this message spans over several periods) when there are the flows with opposite direction in a WSN. Since wireless nodes are usually battery-powered, the objective is also to minimize the energy consumption of nodes by maximizing the schedule period (consequently maximizing time when the nodes stay in low-power mode).

A. Related work

Since the wireless nodes are usually energy-constrained, energy efficiency is a important requirement in order to maximize the lifetime of the network. In [9], the authors have identified the major sources of energy waste in WSNs such as collisions, overhearing and idle listening. Our work eliminates the above mentioned sources of the energy waste by using the collision-free Time Division Cluster Schedule (TDCS) and dedicated Guaranteed Time Slots (GTS) mechanism.

There have been several research works dealing with the energy efficient routing protocols supporting QoS guarantees in WSNs. Real-time Power-Aware Routing (RPAR) protocol [10] integrates transmission power control and real-time routing for supporting energy efficient soft real-time communication in ad hoc WSNs. The protocol is based on the assumption that a higher transmission power results in higher speed. The transmission power is increased if the required speed is not satisfied, otherwise if the required speed is satisfied the transmission power is decreased (to improve energy efficiency). Another real-time routing algorithm minimizing the energy consumption was proposed in [11]. The authors have assumed a collision-free MAC protocol, and they have used multicommodity flow model to schedule the optimal flows' paths in terms of energy consumption while not exceeding links' bandwidths and flows' deadlines. The routing algorithm ensures polynomial-time complexity but no scheduling is considered. On the contrary, our work assumes a scheduling of cluster-tree WSNs where the flows' paths are unique and the routing decisions are simple and time efficient.

The approach based on the combination of an energy efficient topology management protocol with a non energy-aware real-time routing protocol has been proposed in [12]. The nodes, which must be location-aware, are divided into the clusters. To reduce the energy consumption of the nodes while introducing a bounded delay, time-driven data transmissions within each cluster are performed in a Time Division Multiple Access (TDMA) fashion. The relay nodes, which use the real-time routing protocol to forward data between clusters towards the sink, are elected in rotation among the nodes belonging to each cluster. The Frequency Division Multiple Access (FDMA) mechanism prevents the collisions between the nodes operating on different clusters. In [13], this topology management protocol has been extended to support even-driven data transmissions and dynamic cluster formation.

The probabilistic estimation of end-to-end path latency in ad hoc WSNs has been introduced in [14]. This approach approximates the end-to-end delay distribution of a routing path by performing statistical analysis of local information gathered at intermediate hops. It allows to estimate the probability that a sequence of messages is transmitted through a routing path within a time interval with a given confidence level.

To the best of our knowledge, so far no previous research has directly addressed the problem of energy efficient TDMA scheduling of time-bounded data flows in a cluster-tree WSN. Koubaa et al. [15] have proposed an algorithm for collision-free beacon/superframe scheduling in IEEE 802.15.4/ZigBee cluster-tree networks, using the time division approach. Note that the beacon frame scheduling problem comes back to a superframe scheduling problem, since each superframe starts with a beacon frame. The authors have proposed an algorithm based on the "pinwheel scheduling algorithm" [16], which performs the schedulability analysis of a set of superframes with different durations and beacon intervals, and provides a schedule if the set is schedulable. This problem becomes more complex and challenging when time-bounded data flows are assumed. Hence, our work addresses the problem of finding a collision-free superframe schedule that meets all data flows' end-to-end deadlines while minimizing the energy consumption of nodes.

B. Outline and contribution of this paper

The rest of the paper is organized as follows. First we provide a generic system model, encompassing the cluster-tree topology, the data flow model, the cyclic behavior of periodic schedule and the collision domains (Section II). In Section III, some of the most relevant aspects of the IEEE 802.15.4/ZigBee protocols are described. Then, in Section IV, a solution of scheduling problem and application to a IEEE 802.15.4/ZigBee cluster-tree WSN are provided. The problem of finding a feasible schedule is formulated as a cyclic extension of the Resource Constrained Project Scheduling with Temporal Constraints (RCPS/TC). A performance evaluation of the cluster scheduling mechanism takes place in Section V.

In particular, the paper presents the following contributions:

1) A formulation of the scheduling problem by a cyclic extension of RCPS/TC (Sections IV-B and IV-C). Using this formulation, the users are not restricted to a particular implementation but they can make a similar extension to any of the algorithms solving this type of problem.
2) A solution of cyclic extension of RCPS/TC by an Integer Linear Programming (ILP) (Section IV-D), where a grouping of Guaranteed Time Slots (GTS) leads to very efficient ILP model having a few decision variables.
3) An application of this methodology to a specific case of IEEE 802.15.4/ZigBee cluster-tree WSNs (Section IV-A).
4) A time complexity evaluation of the proposed TDCS algorithm implemented in Matlab while using the simplex-based GLPK solver (Section V).

Using our scheduling tool (available on [17]), system designers are able to derive the configuration parameters of each cluster (such as BO, SO and $StartTime$) in IEEE 802.15.4/ZigBee beacon-enabled cluster-tree WSNs. Furthermore, for

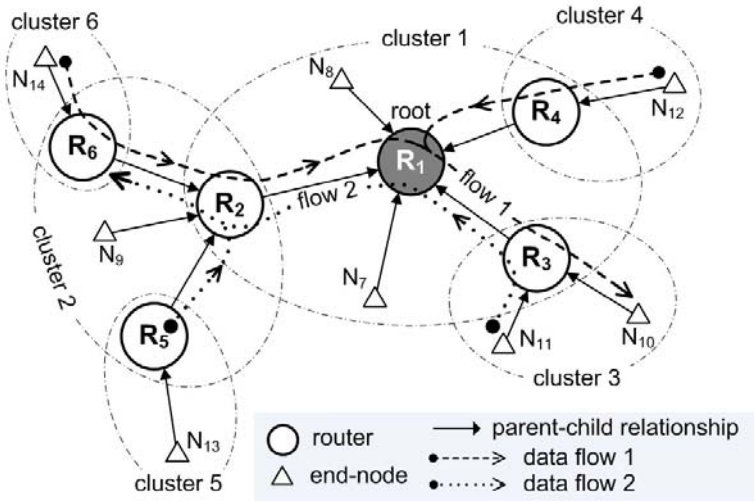


Fig. 1. Cluster-tree topology with two time-bounded data flows.

every cluster's superframe, the configuration parameters (GTS_params) [8] of each allocated GTS (such as *GTS device*, *GTS direction*, *GTS length* and *GTS starting slot*) can be obtained as well (Tab. II).

II. System model

We consider a static deployment of wireless nodes which defines the physical topology of WSN given by the bidirectional wireless links between every pair of nodes that are within transmission range of each other. The logical topology, based on a physical topology, defines a subset of wireless links to be used for data transmission. In the rest of the paper, we will use notation *topology* while meaning logical topology.

A. Cluster-tree topology model

One of the WSN topologies suited for predictable and energy efficient behavior is a cluster-tree (Fig. 1) where the routing decisions are unique and nodes can enter low-power mode to save their energy. From the hierarchy point of view, the cluster-tree is directed tree (so called in-tree [18]) as depicted by solid arrows in Fig. 1. On the other hand, from the data transmission point of view, the cluster-tree is undirected tree (i.e. the wireless links are bidirectional). The hierarchy of the cluster-tree topology is defined by parent-child relationships, in the sense that each solid arrow in Fig. 1 leaves the *child* node and enters the *parent* node. Note that the in-tree has the following property: one node, called *root*, has no parent and any other node has exactly one parent.

The routers and end-nodes are two types of wireless nodes in cluster-tree WSNs. The nodes that can participate in multi-hop routing are referred to as *routers* (circle labeled by R_i). The nodes that do not allow association of other nodes and do not participate in routing are referred to as *end-nodes* (triangle labeled by N_i). In the cluster-tree topology, the nodes are organized in logical groups, called *clusters*. Each router forms a cluster and is referred to as its *cluster-head* (e.g. router R_2 is the cluster-head of cluster 2). All of its child nodes (e.g. end-node N_9 and routers R_5 and R_6 are child nodes of router R_2) are associated to the cluster, and the cluster-head handles all their transmissions.

This *cluster-tree topology* (Fig. 1) can be described by adjacency matrix $A = (a_{ij})$, where $a_{ij} = 1$ if router j is the parent router of node i, otherwise $a_{ij} = 0$. Remind [18] that A is a square matrix with dimension equal to the total number of nodes in a WSN ($total\_nodes$).

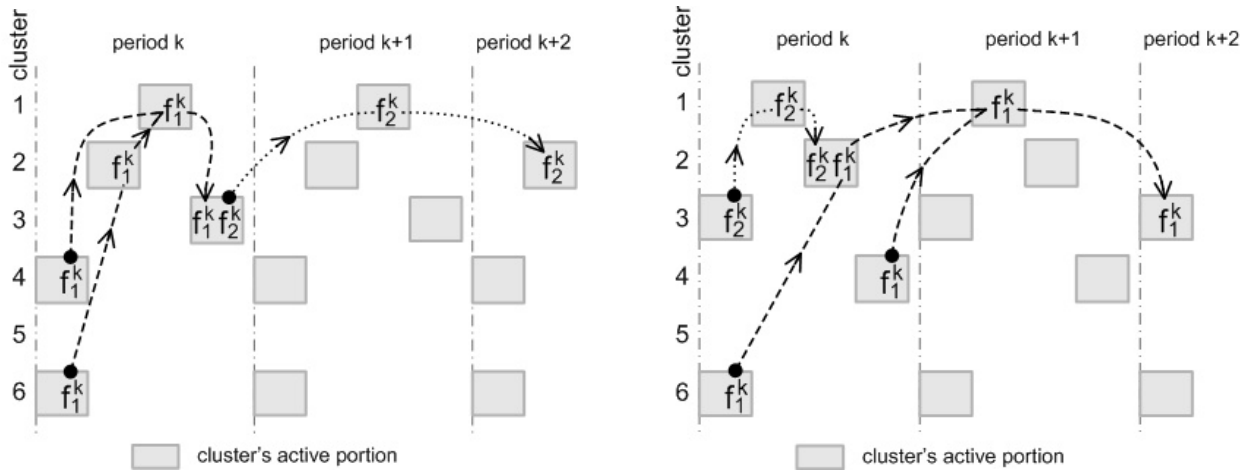
In the cluster-tree topology, the multi-hop communication is deterministic because each node only interacts with its pre-defined parent router and child nodes. Messages are forwarded from cluster to cluster until reaching the sink. The time behavior of each cluster is periodic and the period of each cluster is divided into two portions. *Active* portion, during which the cluster-head enables the transmissions inside its cluster, and subsequent *inactive* portion. Each router (except the root) belongs to two clusters, once as a child node and once as a cluster-head. Therefore, each router is awake whenever one of these two clusters is active, otherwise it may enter the low-power mode to save energy (see the example in Fig. 5).

B. Data flow model

The traffic is organized in the data flows (see user-defined parameters of the flows from Fig. 1 summarized in Tab. I). Each data flow has one or more sources and exactly one sink. In this paper, we assume that both routers and end-nodes can have sensing or/and actuating capabilities, therefore, they can be sources or/and sinks of data flows. A node regularly measures a

flow ID	sources	sink	$e2e\_deadline$		req_period	$sample\_size$	$sample\_ack$
			[sec]	[ptu]	[sec]	[bit]	
1	$\{N_{12}, N_{14}\}$	N_{10}	{0.05, 0.61}	{52, 635}	0.5	64	0
2	$\{R_5, N_{11}\}$	R_6	{0.01, 0.75}	{10, 781}	1	16	0

TABLE I
THE USER-DEFINED PARAMETERS OF THE DATA FLOWS FROM FIGURE 1 (ptu = processing time unit).



(a) Minimized end-to-end delay of flow 1 (dashed line)

(b) Minimized end-to-end delay of flow 2 (dotted line)

Fig. 2. Schedules for data flows in Fig. 1.

sensed value (e.g. temperature, pressure, humidity) with the required period, called the *req_period*, and reports the acquired sensory data of a given size, called the *sample_size*, to a sink. Note that req_period defines the minimal inter-arrival time between two consecutive measurements, and a particular inter-arrival time has to be greater or equal to the req_period.

End-to-end (e2e) delay d_{ij}, given as a time between the instant when a source i sends the message and the instant when the sink j receives this message, is bounded by $e2e\_deadline_{ij}$ such that $d_{ij} \leq e2e\_deadline_{ij}$.

The communication errors such as message corruption or message loss come from unreliable and time-varying characteristics of wireless channels [19]. A corrupted or lost message can be detected by the simple checksum or acknowledgment techniques, respectively, and restored by the retransmission mechanism, for example. All of the above mentioned mechanisms are natively supported by the IEEE 802.15.4 protocol [8]. The messages of a given data flow can be transmitted without acknowledgment, i.e. parameter $sample\_ack = 0$, or with acknowledgment, i.e. $sample\_ack = 1$. Note that the maximum number of retransmissions must be bounded, otherwise, the analysis will not be possible.

C. Cyclic nature

In cluster-tree WSNs, the flows traverse different clusters on their routing paths from the source nodes to the sink nodes. One execution of the flow (i.e. complete data communication from the source node/nodes to the sink node) is called a *wave*, and the notation f_i^k is used to denote wave k of the flow i. The cluster is active only once during the period [5], therefore all the flows in a given cluster are bound together. For example, the gray rectangles on the first line of Figure 2 show active portions of cluster 1 during two consecutive periods accommodating flows 1 and 2 in each period. The key problem is to find a periodic schedule which specifies when the clusters are active while avoiding possible inter-cluster collisions and meeting all data flows' e2e deadlines. The schedule is characterized not only by the moments when the clusters become active within the period, but due to the cyclic nature of the problem it is also characterized by the index of the wave for each flow in a given cluster.

Figure 2 shows two possible schedules of the example in Fig. 1. Even if we relax on the lengths of transmitted messages and on resource constraints related to the cluster collisions, we have to deal with the precedence relations of the wave traversing different clusters. Since the flows have opposite directions in this example, the e2e delay minimization of the first flow is in contradiction with the the minimization of the second flow. Figure 2a shows the case, when e2e delay of the flow 1 is minimized, i.e. the ordered sequence of clusters' active portions is in line with the flow 1 (starting with clusters 4 and 6 and following with clusters 2, 1 and 3), and therefore one wave of this flow fits into one period. On the other hand, the wave of the flow 2 spans over 3 periods while going against the sequence of clusters. Figure 2b illustrates the opposite case, when e2e delay of the flow 2 is minimized (starting with cluster 3 and following with clusters 1 and 2), and consequently flow 1

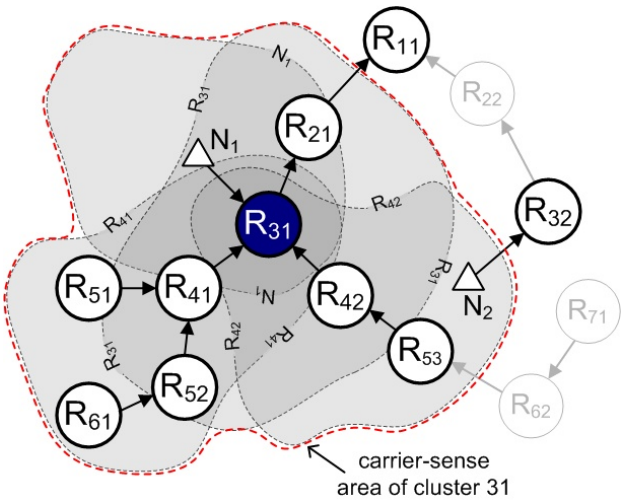


Fig. 3. The carrier-sense area and collision domain (set of bold routers) of cluster 31.

spans over 3 periods. It may happen that none of these schedules is feasible due to the deadline constraints (even if feasible schedule exists - see Fig. 10). Hence, proper order of the active portions of clusters is a subject of optimization even if the lengths of messages and collisions of clusters are not assumed.

The instances where the flows do not have opposite directions (please assume an instance containing one flow only, or try to exchange orientation of flow 2 in Fig. 1) can be easily scheduled within one period as the ordered sequence (i.e. phasing) of clusters' active portions following a topological order of a directed acyclic graph representing the flows.

D. Collision domains

According to the strength of the radio signal, the transmission range and the carrier-sensing range [20] can be defined around each transmitter. When a receiver is in the *transmission range* of a transmitter, it can receive and correctly decode messages from the transmitter. On the other hand, a node in the *carrier-sensing range* (also called the hearing range), but not in the transmission range, is able to sense the transmission (or even significant radio energy), but cannot decode the messages correctly. The carrier-sensing range is always larger than the corresponding transmission range [20]. In what follows, the topology is given by the transmission ranges while the collision domains depend on the carrier-sense ranges.

A *carrier-sense area* of a cluster is covered by the overlapping carrier-sense ranges of its cluster-head and associated child nodes. A *collision domain* of a cluster is a set of clusters, which compete for the same radio channel and, therefore, their active portions must be non-overlapping, i.e. only one cluster from a collision domain can be active at a given time instant. The collision domain of cluster i comprises the cluster j if and only if the carrier-sense area of cluster i comprises cluster-head or any of child nodes of cluster j. Hence the collision domain depends on the physical deployment of a WSN as well as on the topology (i.e. parent-child relationships).

Let us consider the illustrative example in Fig. 3. The carrier-sense area of cluster 31 (gray region in Fig. 3) is covered by the carrier-sense ranges of cluster-head R_{31} and its child nodes (i.e. routers R_{41}, R_{42} and end-node N_1). Hence, the collision domain of cluster 31 comprises the clusters 31, 41, 42, 51, 52, 53, 61, 21 whose cluster-heads are inside the carrier-sense area (i.e. R_{31}, R_{41}, R_{42}, R_{51}, R_{52}, R_{53}, R_{61}, R_{21}), and the clusters 11 and 32 whose child nodes are inside the carrier-sense area (i.e. R_{21} and N_2).

The collision domains of a WSN are defined by collision matrix $C = (c_{ij})$, where $c_{ij} = 1$ if cluster j is within the collision domain of cluster i, otherwise $c_{ij} = 0$. Then, C is a square matrix with dimension equal to the total number of clusters in a WSN ($total\_routers$). In the example of Fig. 1, we assume that the clusters 4, 6 and clusters 4, 5 can be active at the same time (i.e. the collision domain of cluster 4 does not comprise clusters 5 and 6).

III. Overview of IEEE 802.15.4/ZigBee

The IEEE 802.15.4 [8] standard specifies the physical and data link layers while the network and application layers are defined by the ZigBee specification [5]. The MAC layer supports the beacon-enabled or non beacon-enabled modes that may be selected by a central controller of the WSN, called *PAN coordinator*. This paper only considers the beacon-enabled mode, since it supports cluster-tree topology and enables the provision of the collision-free access to the wireless medium through the *Guaranteed Time Slot* (GTS) mechanism.

While IEEE 802.15.4 in the beacon-enabled mode supports only the star-based topology, the ZigBee specification has proposed its extension to the cluster-tree topology. In the particular case of ZigBee cluster-tree WSNs, a PAN coordinator is

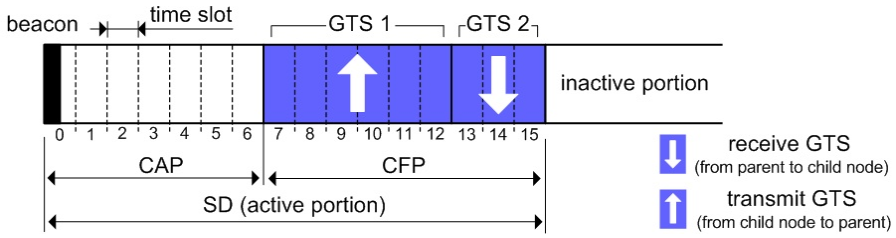


Fig. 4. Superframe structure.

identified as the root of the tree and forms the initial cluster. The other routers join the cluster-tree in turn by establishing themselves as cluster-heads, starting to generate the beacon frames for their own clusters. Beacon frames are periodically sent at *Beacon Interval* (BI) to synchronize the child nodes that are associated with a given cluster-head and to define a superframe structure (Fig. 4).

Each cluster's period, corresponding to BI, is divided into an active and an inactive portions. The active portion, corresponding to *Superframe Duration* (SD), is divided into 16 equally-sized time slots, during which the data transmission is allowed. These time slots are further grouped into a *Contention Access Period* (CAP) using slotted CSMA/CA for the best-effort data delivery, and an optional *Contention Free Period* (CFP) supporting the time-bounded data delivery. Within the CFP, the cluster-head can allocate Guaranteed Time Slots (GTSs) to its child nodes. The CFP supports up to 7 GTSs and each GTS may contain one or more time slots. Each child node may request up to one GTS in the *transmit direction*, i.e. from the child node to the parent router, and/or one GTS in the *receive direction*, i.e. from the parent router to the child node. Note that a node to which a GTS has been allocated can still transmit the best-effort data within the CAP. During the inactive period, each associated node may enter a low-power mode to save energy.

Durations of the cluster's period (BI) and the cluster's active portion (SD) are defined by two parameters, the *Beacon Order* (BO) and the *Superframe Order* (SO) as follows:

$$\begin{aligned} BI &= aBaseSuperframeDuration \cdot 2^{BO} \\ SD &= aBaseSuperframeDuration \cdot 2^{SO} \end{aligned} \tag{1}$$

where $0 \leq SO \leq BO \leq 14$ and *aBaseSuperframeDuration* = 15.36 ms (assuming the 2.4 GHz frequency band and 250 kbps of bit rate) and denotes the minimum duration of active portion when $SO = 0$. Note that the ratio of the active portion (SD) to the whole period (BI) is called the *duty-cycle*.

Note that due to the cluster-tree topology, each router (except the root) belongs to two clusters, once as a child node and once as a cluster-head. Hence, router r has to maintain the timing between the active portion (SD) of its parent's cluster (in which a beacon and the data frames from the parent router are received, and the data frames to the parent router are sent) and its own active portion (in which a beacon and the data frames are sent to the associated child nodes, and the data frames from child nodes are received). Router r acts as a child node in the former active portion while in the latter active portion it acts as a cluster-head. The relative timing of these active portions is defined by the *StartTime* parameter [8]. The illustrative example is shown in Fig. 5, where the router R_2 acts as child node in cluster 1 (shaded rectangle) and as cluster-head in cluster 2 (solid rectangle), for example.

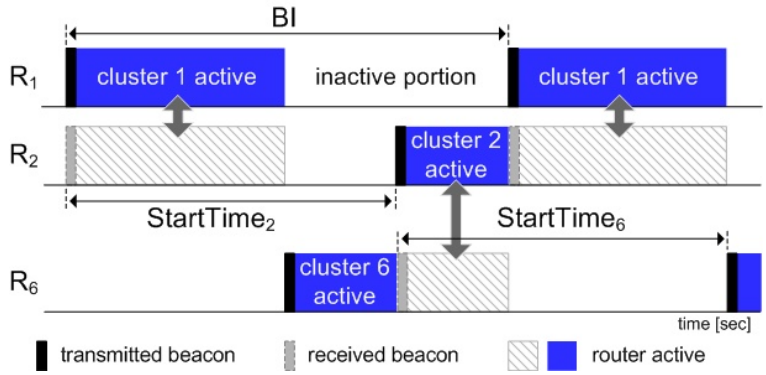


Fig. 5. Timing among clusters 1,2 and 6 from Figure 1.

IV. Time Division Cluster Scheduling and its Application to IEEE 802.15.4/ZigBee

The key idea of this paper is to formulate the problem of finding a feasible *Time Division Cluster Schedule* (TDCS) as a cyclic extension of the RCPS/TC (Resource Constrained Project Scheduling with Temporal Constraints) problem [21], so that the users are not restricted to a particular implementation but they can make a similar extension to any of the algorithms solving this problem.

The objective is to minimize the energy consumption of the nodes by maximizing the TDCS period, corresponding to BI, while avoiding possible inter-cluster collisions (i.e. resource requirements) and meeting all flows' end-to-end deadlines (i.e. temporal requirements). All clusters have equal BI, defined by BO, but various SD (Section IV-A), defined by SO, (i.e. various duty-cycle) to ensure efficient bandwidth utilization. The BI should be set as long as possible to minimize clusters' duty-cycle and consequently to minimize the energy consumption of the nodes. Hence, the TDCS algorithm (see pseudo code in Fig. 6 where the content of the `init` function is explained in Sections IV-A, IV-B, IV-C, the `ilp_solve` function is in Section IV-D:1 and the `config_params` function is in Section IV-D:2) is iterating from the minimum BI up to the maximum BI. The maximum BI, given by BO_{max}, is rounded down to the nearest BI (Eq.(1)) towards the shortest *req_period* among all of the flows. The minimum BI, given by BO_{min}, is rounded up to the nearest BI towards the duration of all clusters' SDs when assuming that non-interfering clusters overlap. If a feasible TDCS is found for a given BI, BO is increased by 1 and next iteration is called with new BI. This procedure is repeated until $BO = BO_{max}$ or a feasible TDCS is not found. Then, the last feasible TDCS meets all the resource and temporal requirements while minimizing the energy consumption of the nodes.

```
(BO,SO,StartTime,GTS_params) = TDCS(C,A,flows)
01 begin
02   (BO_min,BO_max,p,SO,V,W,GTS_params) = init(C,A,flows)
03   BO = BO_min
04   while BO ≤ BO_max
05     (ŝ,q̂,feasible) = ilp_solve(V,W,BO,p)
06     if feasible
07       BO = BO + 1
08     else
09       break
10     end
11   end
12   /* calculate the StartTime parameter of each cluster */
13   StartTime = config_params(ŝ,BO)
14 end
```

Fig. 6. Pseudo code of the TDCS algorithm.

A. Duration of the cluster's active portion in IEEE 802.15.4/ZigBee

The duration of the cluster's active portion (SD) is given by the amount of data traffic traversed through a given cluster. Each SD is divided into a CAP and CFP (Fig. 4). Within the CFP, a given router allocates the GTSs in transmit or receive directions for the data received from or transmitted to associated child nodes, respectively. To reduce the resource requirements of the routers and end-to-end delays, we introduce the following priority rule: "*When a cluster-head handles several GTSs in opposite directions, the transmit GTSs (i.e. communication from child-node to cluster-head) are allocated before the receive GTSs (i.e. communication from cluster-head to child-node)*". Using this rule, the end-to-end delay of a flow can be reduced by one period of TDCS at a router which allocates both receive and transmit GTSs for a given flow. For example, in Fig. 1, router R_1 allocates one transmit GTSs and one receive GTS for flow 1. In order to reduce the computational complexity, the GTSs inside the transmit or receive group are in an arbitrary order and are not the subject of the optimization.

The length of each GTS is given by the amount of transmitted data. Each GTS includes effective data transmission and overheads (i.e. inter-frame spacing and eventual acknowledgment and retransmissions). Consecutive frames are separated by inter-frame spacing (IFS). The IFS is equal to a SIFS (Short Inter-Frame Spacing) or a LIFS (Long Inter-Frame Spacing) according to the length of MAC frame.

IEEE 802.15.4 protocol supports acknowledgment and retransmission mechanisms to minimize the communication errors coming from the unreliable and time-varying characteristics of wireless channels. In the case of acknowledged transmissions (i.e. $sample\_ack = 1$) the sender waits for the corresponding acknowledgment frame at most *macAckWaitDuration* [8]. If an

acknowledgment frame is received within *macAckWaitDuration*, the transmission is considered successful. Otherwise, the data transmission and waiting for the acknowledgment are repeated up to a maximum of *macMaxFrameRetries* [8] times. If an acknowledgment frame is not received after *macMaxFrameRetries* retransmissions, the transmission is considered failed. Note that each retransmission decreases the effective throughput, increases the communication delay and energy consumption such that a fair trade-off between reliability, energy consumption and timeliness must be found.

The whole data transmission, including the data frame, IFS and eventual acknowledgment and retransmissions, must be completed before the end of the current GTS. Otherwise, it must wait until the GTS in the next superframe.

The duration of a GTS required for the whole data transmission is expressed as:

$$T_{GTS} = \sum_{i=1}^{e} \left(\begin{array}{c} (macMaxFrameRetries \cdot sample\_ack_i + 1) \cdot \\ (frm\_size_i/rate + macAckWaitDuration \cdot sample\_ack_i) + \Delta IFS_i \end{array} \right) \tag{2}$$

where frm_size is the size of transmitted frame including the data payload, MAC and PHY headers; $rate$ is the data rate equal to 250 kbps; ΔIFS is equal to SIFS or LIFS depending on the length of MAC frame; and e is the number of flows in the transmit or receive direction belonging to a given child node.

The number of allocated time slots for a given GTS is then equal to:

$$N_{GTS} = \left\lceil \frac{T_{GTS}}{TS} \right\rceil \tag{3}$$

where TS is the duration of a time slot and is equal to SD/16. The number of time slots, N_{GTS}, is calculated for each allocated GTS in a given superframe. The remaining time slots of SD are utilized for the best-effort traffic within the CAP. The allocated GTSs cannot reduce the length of the CAP to less than *aMinCAPLength* [8].

The superframe duration (SD) is then computed iteratively starting from $SO = 0$. If the number of time slots required for all allocated GTSs in a given superframe is greater than $16 - \lceil aMinCAPLength/TS \rceil$, the SO is increased by 1 and the length of each GTS (Eq. (3)) is recalculated. This procedure is repeated until all allocated GTSs fit into a given SD.

To ensure efficient bandwidth utilization, the SD of the clusters handling a higher amount of data traffic should be longer than the ones handling less amount of data traffic. Thus, the adequate SD is computed for each cluster such that for each cluster k, we get SO_k and the configuration parameters (GTS_params) [8] of each allocated GTS, i.e. *GTS device*, *GTS direction*, *GTS length* and *GTS starting slot*. In case of the example in Fig. 1, we get $SO_1 = 1$ and $SO_2 = SO_3 = SO_4 = SO_6 = 0$. All clusters have the same BO equal to 5, which gives the longest possible BI minimizing the energy consumption of the nodes (see Section V). Table II presents the output of TDCS tool [17], namely the configuration parameters of clusters from the simulation scenario (Fig. 1) assuming unacknowledged transmission (i.e. $macMaxFrameRetries = 0$).

cluster	**BO**	**SO**	**StartTime** [sec]	**GTS device**	**GTS length**	**GTS direction**	**GTS starting slot**
cluster 1	5	1	0.0	R_2	1	transmit	10
				R_3	1	transmit	11
				R_4	1	transmit	12
				R_2	1	receive	13
				R_3	1	receive	14
cluster 2	5	0	0.04608	R_5	2	transmit	8
				R_6	2	transmit	10
				R_6	4	receive	12
cluster 3	5	0	0.03072	N_{11}	2	transmit	10
				N_{10}	4	receive	12
cluster 4	5	0	0.47616	N_{12}	2	transmit	14
cluster 6	5	0	0.43008	N_{14}	2	transmit	14

TABLE II
THE CONFIGURATION PARAMETERS OF CLUSTERS OBTAINED BY THE TDCS TOOL.

The above described algorithm for the calculation of the duration of clusters' active portions, given by SO, is illustrated in Fig. 7.

```
01 for each cluster i
02   SO_i = −1
03   repeat
04     for each child node j of cluster i
05       calculate N^T_{GTS,j} for all flows in transmit direction
06       calculate N^R_{GTS,j} for all flows in receive direction
07     end
08     SO_i = SO_i + 1
09   until Σ_j N^T_{GTS,j} + Σ_j N^R_{GTS,j} ≤ 16 − ⌈aMinCAPLength/TS⌉
10   /* processing time of cluster-task T_i */
11   p^T_i = Σ_j N^T_{GTS,j}      p^R_i = Σ_j N^R_{GTS,j}      p^CAP_i = 16 − (p^T_i + p^R_i)
12 end
```

Fig. 7. Pseudo code of the calculation of clusters' SOs and the processing times of cluster-tasks.

B. TDCS formulated as a cyclic extension of RCPS/TC

The concept of *(general) temporal constraints* (also called *minimum and maximum time lags*) have been classified by Brucker et al. [22]. The problem was studied by the operations research community, but similar principles have also appeared in the optimization of compilers for multiprocessor machines [23] and in symbolic representation of states in timed automata [24].

1) General description of RCPS/TC: The set of n tasks $\mathcal{T} = \{T_1, \dots T_i, \dots T_n\}$ with temporal constraints is given by a graph of communication tasks G (see Fig. 8), where the vertices correspond to the tasks and the directed edges represent the temporal constraints between the tasks. The scheduling problem is then defined as searching for such a *feasible schedule* $(s_1, s_2, \dots s_n)$, which satisfies the *temporal constraints* and *resource constraints* while minimizing the objective criterion. Note that in RCPS/TC terminology, s_i is the start time of task T_i related to the beginning of the schedule (i.e. time 0), but in IEEE 802.15.4/ZigBee terminology, the parameter $StartTime$, is related to the moment, when the beacon frame from the parent router was received (Fig. 5). Parameter $StartTime$ can be easily derived from start time s and matrix A, since parameter $StartTime$ of the root is equal to 0 (see Eq. (13) for more details).

Each edge from vertex T_i to vertex T_j is labeled by a weight $w_{i,j}$ which constrains the start times of the tasks T_i and T_j by the inequality $s_j - s_i \geq w_{i,j}$. There are two kinds of edges: the edges with positive weights and the edges with negative weights. The edge, from vertex T_i to vertex T_j with a positive weight $w_{i,j}$ (giving the minimum time lag), indicates that s_j, the start time of T_j, must be at least $w_{i,j}$ time units after s_i, the start time of T_i (i.e. $s_j \geq s_i + w_{i,j}$). We use the positive weights $w_{i,j}$ to represent the *precedence constraint*, i.e. $w_{i,j} = p_i$ and therefore T_j starts after completion of T_i, where p_i is processing time of T_i.

The edge, from vertex T_j to vertex T_i with a negative weight $w_{j,i}$ (giving the maximum time lag), indicates that s_j must be no more than $|w_{j,i}|$ time units after s_i (i.e. $s_j \leq s_i + w_{j,i}$). Therefore, each negative weight $w_{j,i}$ represents the *relative deadline* of task T_j in relation to task T_i. Consequently, when T_j is the last task of the flow and T_i is the first task of the same flow, the edge with a negative weight may be conveniently applied for e2e deadline such that the value of e2e deadline is equal to $|w_{j,i}| + p_j$.

2) Problem complexity: Positive and negative edges may be used to represent many other time constraints that occur frequently in practice [21], [25]. The necessary condition for finding a feasible schedule is a graph G without cycle of positive length [18]. Moreover, the solving algorithms have to consider the *resource constraints* which ensure that two tasks with a potential conflict are not executed at once. From the time complexity point of view, the problem is NP-hard, due to the resource constraints leading to a non-convex representation of the set of feasible solutions. The ILP formulation of the problem (shown in Section IV-D) gives exact answer to the schedulability problem (i.e. feasible time-triggered schedule is found for each schedulable instance and no solution is found for each non-schedulable instance). There are several polynomial time heuristics [21], [25], [26] that are able to handle RCPS/TC problems with up to one thousand of tasks in a few seconds. These algorithms usually include lower/upper bound estimation on the length of the period. With respect to the schedulability problem, heuristics may be pessimistic while evaluating an instance as non-schedulable even if there exist a feasible schedule (but we do not know a polynomial-time algorithm able to solve this problem, since already decision problem upon schedulability of non-preemptive tasks with fixed release dates and deadlines on monoprocessor is NP-complete).

3) Task model of clusters and flows: Any feasible TDCS has to respect the resource constraints related to the collision domains of clusters and the temporal constraints of the flows. Hence, the set of tasks $\mathcal{T}$ will consist of two disjoint subsets: a set of cluster-tasks and a set of dummy-tasks reflecting the temporal constraints only. The duration of a task T_i is given by its processing time p_i.

The *cluster-task* T_i is created for each cluster i. Note that the clusters which do not route any data flow have $p_i = 0$ (i.e. cluster-task 5 is not shown in Fig. 8). In the case of an active cluster-task (i.e. the one routing at least one data flow) the processing time is equal to the cluster's SD (i.e. p_i = SD computed in Section IV-A) and includes all the communications handled by the given cluster. That means, for each cluster-task, we define the duration of the CAP as p_i^{CAP}, the duration of all GTSs in the transmit direction as p_i^T and the duration of all GTSs in the receive direction as p_i^R, i.e. the processing time of cluster-task is given as $p_i = p_i^{CAP} + p_i^T + p_i^R$. The unit of processing time, called *ptu*, is equal to the length of a time slot when $SO = 0$ (i.e. 1 ptu = $aBaseSuperframeDuration/16 = 0.96$ ms).

Each dummy-task has a processing time equal to 0 since they are used to handle temporal constraints of different flows.

Let us consider the illustrative example of cluster-tree WSN in Fig. 1, where periodic time-bounded traffic is sent using two data flows. Within the first data flow, messages are sent from source nodes N_{12} and N_{14} to the sink node N_{10}. In the second case, nodes R_5 and N_{11} send messages to the sink router R_6. The user-defined parameters of the flows are summarized in Tab. I. Thus, cluster-tasks T_1, T_2, T_3, T_4, T_5, T_6 are associated with clusters 1, 2, 3, 4, 5 and 6. The processing time of each cluster-task is equal to its associated cluster's SD as follows: $p^{CAP} = [20, 8, 10, 14, 0, 14]$, $p^T = [6, 4, 2, 2, 0, 2]$ and $p^R = [6, 4, 4, 0, 0, 0]$. Cluster 5 does not route any flow, thus its processing time is equal to 0. Note that since Superframe Order of cluster 1 is equal to 1 ($SO_1 = 1$), the processing time p_1 was doubled.

The collisions among the routers are represented as conflicts among the cluster-tasks due to the shared resources. We define $\mathcal{M}$ as a set of couples of the cluster-tasks having a potential conflict. Consider two cluster-tasks T_i and T_j. The potential conflict between T_i and T_j is a couple $\{i, j\}$ derived from the collision matrix C as follows: we say that $\{i, j\} \in \mathcal{M}$ if and only if $C_{i,j} = 1$, $p_i > 0$ and $p_j > 0$.

Precedence constraints of each flow are represented by an in-tree of dummy-tasks connected by positive edges. The leaves correspond to the source clusters, where the source nodes are associated, and the root to the sink cluster, where the sink is associated. In our particular example in Fig. 8, the in-tree of dummy-tasks T_{11}, T_{10}, T_9, T_8, T_7 corresponds to flow 1, and the in-tree of dummy-tasks T_{14}, T_{13}, T_{12} corresponds to flow 2. Each dummy-task represents a given flow in a given cluster (e.g. T_{11} represents flow 1 in cluster 6).

4) Cyclic extension: The messages transmitted periodically over the network can be considered as a periodic execution of task-set $\mathcal{T}$. As mentioned in Subsection II-C, one wave of a given flow may go over several periods and, therefore, we are faced with a cyclic scheduling problem.

Let $\hat{s}_i$ be the start time within the period, i.e. remainder after division of s_i by BI, and let $\hat{q}_i$ be the index of the period, i.e. the integer part of this division. Then start time s_i can be expressed as follows:

$$s_i = \hat{s}_i + \hat{q}_i \cdot \text{BI} \qquad \text{for} \quad \hat{s}_i \in \langle 0, \text{BI} - 1 \rangle \,,\ \hat{q}_i \geq 0. \tag{4}$$

This notation divides s_i into segment $\hat{q}_i$ and offset $\hat{s}_i$. Hence, two tasks T_i and T_j within one period may have a different $\hat{q}_i$ and $\hat{q}_j$, since the pieces of data related to these tasks correspond to the different waves (this notion used in cyclic scheduling [27], [28] is identical to the modulo scheduling or SW pipelining in the parallel compiler community [29]).

The cyclic schedule has to follow several constraints:

- *Precedence constraints and relative deadlines* are given by inequality $s_j - s_i \geq w_{i,j}$. As a result, by applying Eq. (4) we obtain:
$$(\hat{s}_j + \hat{q}_j \cdot \text{BI}) - (\hat{s}_i + \hat{q}_i \cdot \text{BI}) \geq w_{i,j}. \tag{5}$$
- *Offset precedence constraints and offset relative deadlines* are used to bind the flow-related dummy tasks with the cluster-task. They represent the relation between two tasks that can be from different waves. Therefore, they do not contain the segment values $\hat{q}$ and can be expressed as:
$$\hat{s}_j - \hat{s}_i \geq v_{i,j}. \tag{6}$$
The offset weights $v_{i,j}$ are used to distinguish the *offset precedence constraints* from *"normal" precedence constraints*.
- *Resource constraints* given by $\mathcal{M}$, the set of potential conflicts of the cluster-tasks. The conflicts have to be avoided in order to obtain a feasible schedule (detailed explanation is given in Section IV-D).

C. Graph of the communication tasks

An important step of the scheduling algorithm is the construction of the graph of the communication tasks G (Fig. 8) using the data flows in Tab. I and topology in Fig. 1 (i.e. adjacency matrix A and collision matrix C).

Each dummy-task is synchronized with the corresponding cluster-task. The synchronization is made by means of *offset precedence constraints* represented by dashed edges in Fig. 8. All of them have the weight $v_{i,j} = 0$, therefore, for example, $\hat{s}_6 = \hat{s}_{11}$ is given by two inequalities (6), i.e. $\hat{s}_6 \geq \hat{s}_{11}$ and $\hat{s}_6 \leq \hat{s}_{11}$.

Positive edges are used to represent precedence constraints of the flows. For example of flow 1, dummy-task T_9 starts after dummy-task T_{11} is completed, which is represented by the positive edge with weight $w_{11,9}$ equal to the processing time of cluster-task T_6, i.e. $w_{11,9} = p_6 = 16$.

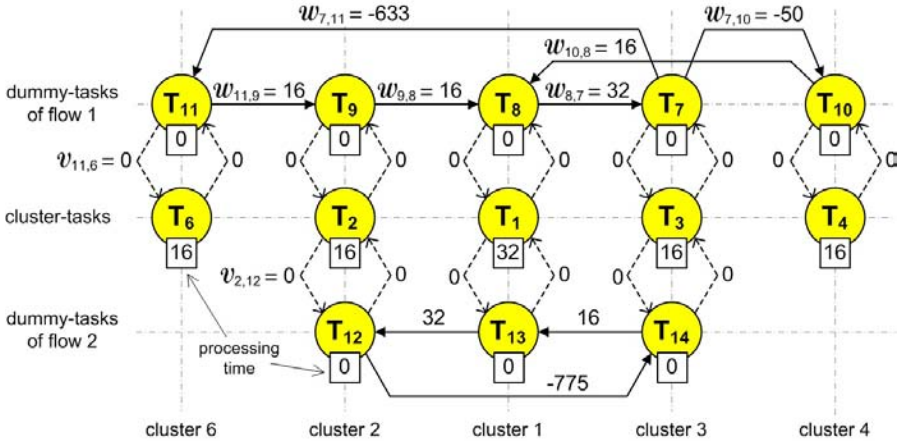


Fig. 8. Graph G of the tasks corresponding to example in Figure 1.

$$\min \sum_{i=1}^{n} \hat{s}_i + \hat{q}_i \cdot \text{BI} \tag{8}$$

subject to:

$$\hat{s}_j + \text{BI} \cdot \hat{q}_j - \hat{s}_i - \text{BI} \cdot \hat{q}_i \geq w_{ij} \qquad \forall (i,j); i \neq j, w_{ij} \neq -\infty \tag{9}$$

$$\hat{s}_j - \hat{s}_i \geq v_{ij} \qquad \forall (i,j); i \neq j, v_{ij} \neq -\infty \tag{10}$$

$$\hat{s}_i - \hat{s}_j + \text{BI} \cdot x_{ij} \geq p_j \qquad \forall \{i,j\} \in \mathcal{M}; i < j \tag{11}$$

$$\hat{s}_i - \hat{s}_j + \text{BI} \cdot x_{ij} \leq \text{BI} - p_i \qquad \forall \{i,j\} \in \mathcal{M}; i < j \tag{12}$$

where: $\hat{s}_i \in \langle 0, \text{BI} - p_i \rangle$; $\hat{q}_i \geq 0$; $\hat{s}_i, \hat{q}_i \in \mathbb{Z}$; $x_i \in \{0,1\}$

Fig. 9. ILP formulation for cyclic extension of the scheduling problem.

Negative edges are used to represent the e2e deadlines of the flows. The e2e deadline of the flow spans from the beginning of transmit or receive GTS's groups to the end of transmit or receive GTS's groups (see Fig. 11). On the other hand, the relative deadline between corresponding tasks, given by the weight $w_{i,j}$, starts and ends at the beginning of the tasks. Hence, the e2e deadline must be aligned with the beginning of corresponding tasks. For example of a sub-flow of flow 1 from source end-node N_{14} to sink end-node N_{10}, the relative deadline between the corresponding dummy-tasks T_7 and T_{11} is given by the weight $w_{7,11}$ as follows:

$$\begin{aligned} w_{7,11} &= -\big(e2e\_deadline_{N_{14}N_{10}} + (p_6^{CAP} + p_6^T \cdot \theta_{1,6}) - (p_3^{CAP} + p_3^T + p_3^R \cdot (1 - \theta_{1,3}))\big) \\ &= -\big(635 + (14 + 2 \cdot 0) - (10 + 2 + 4 \cdot 1)\big) = -633 \text{ ptu} \end{aligned} \tag{7}$$

where $\theta_{f,r}$ is a binary constant, which is equal to 1 when router r is source/sink of flow f and is equal to 0 when a child node of router r is source/sink of flow f. The resulting end-to-end delay ($d_{N_{14}N_{10}}$) is constrained by $e2e\_deadline_{N_{14}N_{10}}$ (see Fig. 11), and it spans from the beginning of transmit GTS's group of cluster 6 (since the measured data has to be received from the end-node N_{14} through a transmit GTS) to the end of receive GTS's group of cluster 3 (since the received data has to be dispatched to the end-node N_{10} through a receive GTS).

Graph G in Fig. 8 is given by W, the adjacency matrix of the weights $w_{i,j}$, and V, the adjacency matrix of the offset weights $v_{i,j}$. If there is no edge from T_i to T_j, then $w_{i,j} = v_{i,j} = -\infty$.

D. Solution of the scheduling problem

1) Integer Linear Programming (ILP) formulation for cyclic extension of RCPS/TC: The ILP model is given in Fig. 9 where x_{ij} is a binary decision variable such that $x_{ij} = 1$ if $\hat{s}_i \leq \hat{s}_j$ (i.e. T_i is followed by T_j or both T_i and T_j start at the same time) and $x_{ij} = 0$ if $\hat{s}_i > \hat{s}_j$ (i.e. T_j is followed by T_i).

Note that the period BI, vector p, matrices W, V and the set of potential conflicts $\mathcal{M}$ are input parameters of the declarative program in Fig. 9.

Constraint (9) is a direct application of the precedence constraints and relative deadlines given by W. Constraint (10) relates to the offset precedence constraints and offset relative deadlines given by V. Constraints (11) and (12) limit the number of

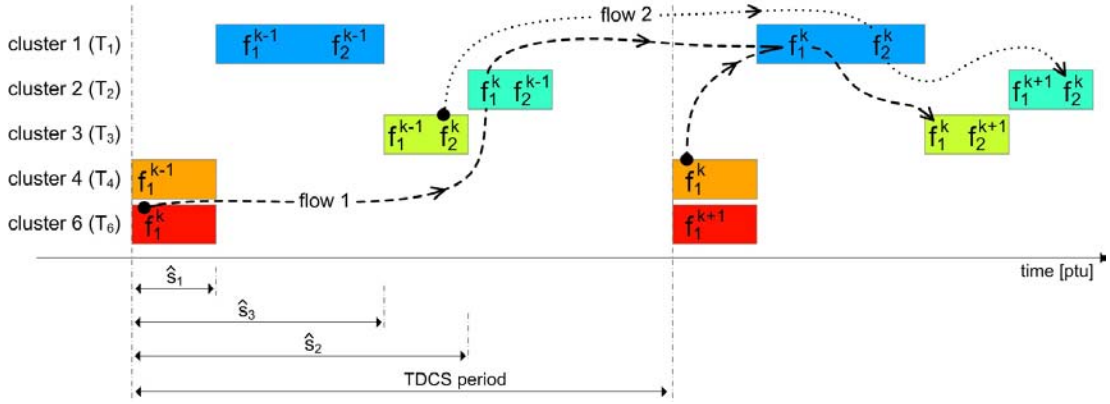


Fig. 10. Gantt chart of TDCS including flows 1 and 2.

tasks executed at a given time. The binary decision variable x_{ij} defines the mutual relation of tasks T_i and T_j ($i \neq j$) within the period as follows:

- When $x_{ij} = 0$, constraint (12) is eliminated in effect (since $\hat{s}_i - \hat{s}_j + \text{BI} \geq p_j$ is always true with respect to the definition domain of variable s) and constraint (11) reduces to $\hat{s}_j + p_j \leq \hat{s}_i$, i.e. T_j is followed by T_i within the period.
- When $x_{ij} = 1$, constraint (11) is eliminated in effect and constraint (12) reduces to $\hat{s}_i + p_i \leq \hat{s}_j$, i.e. T_i is followed by T_j within the period.

We have implemented the above mentioned scheduling algorithm in the Matlab [17] using GLPK solver. Figure 10 shows the offsets of start times ($\hat{s}$) of cluster-tasks (namely $\hat{s}_1 = 16$, $\hat{s}_2 = 64$, $\hat{s}_3 = 48$, $\hat{s}_4 = 0$, $\hat{s}_6 = 0$) in the form of a Gantt chart for one whole period of a feasible TDCS of the cluster-tree WSN in Fig. 1 including flows 1 and 2 along the wave k. The value of start times $\hat{s}$ is in processing time units (ptu). Note that the cluster-tasks T_4 and T_6 can overlap because the collision domain of cluster 4 does not include cluster 6 and vice versa. In addition, the output of algorithm contains the index ($\hat{q}$) of the TDCS period for each flow related to dummy-task as follows: $\hat{q}_{11} = 0$, $\hat{q}_{10} = 1$, $\hat{q}_9 = 0$, $\hat{q}_8 = 1$, $\hat{q}_7 = 1$ and $\hat{q}_{14} = 0$, $\hat{q}_{13} = 1$, $\hat{q}_{12} = 1$.

2) Calculation of configuration parameters: The $StartTime$ parameter of each cluster's active portion (except the root) is computed from the offset of start times as follows:

$$StartTime_i = \hat{s}_i + \gamma \cdot \text{BI} - \hat{s}_{parent} \tag{13}$$

where $\hat{s}_{parent}$ is the offset of start time of the parent cluster-task of cluster-task i, and $\gamma = 1$ if $\hat{s}_i < \hat{s}_{parent}$; otherwise $\gamma = 0$. The $StartTime$ parameter of the active portion of the root is equal to 0. In case of the example in Fig. 1 in which $BO = 5$ (BI = 512 ptu), the $StartTime$ parameter of the active portion of cluster 4 is then computed as: $StartTime_4 = \hat{s}_4 + 1 \cdot \text{BI} - \hat{s}_1 = 0 + 512 - 16 = 496$ ptu.

Using Eq. (4), the e2e delay between each source and sink of a given flow is computed. For example of a sub-flow of flow 1 from source end-node N_{14} to sink end-node N_{10}, the e2e delay is computed as follows:

$$\begin{aligned} d_{N_{14}N_{10}} &= \left(s_7 + p_3^{CAP} + p_3^R + p_3^T \cdot (1 - \theta_{1,3})\right) - \left(s_{11} + p_6^{CAP} + p_6^T \cdot \theta_{1,6}\right) \\ &= \left((560 + 10 + 2 + 4 \cdot (1 - 0)\right) - \left(0 + 14 + 2 \cdot 0\right) = 562 \text{ ptu} \end{aligned} \tag{14}$$

where binary constant $\theta_{f,r}$ has been defined in Eq. (7); s_7 and s_{11} are start times of corresponding dummy-tasks T_7 and T_{11}; p_3 and p_6 are processing times of cluster 3 and 6, respectively, where end-nodes N_{10} and N_{14} are associated.

The time line of clusters' active portions including allocated GTSs is presented in Fig. 11. The $StartTime$ parameters and e2e delays are based on the values of $\hat{s}$, $\hat{q}$ and using of Egs. (14) and (13).

Using our methodology, system designers are able to configure the parameters of each cluster, such as BO, SO and $StartTime$, in IEEE 802.15.4/ZigBee cluster-tree WSNs. Furthermore, for every cluster's superframe, the configuration parameters (GTS_params) [8] of each allocated GTS such as *GTS device*, *GTS direction*, *GTS length* and *GTS starting slot* can be obtained as well.

V. Performance Evaluation

This section focuses on the time complexity of TDCS algorithm and the simulation study of energy consumption. The proposed TDCS algorithm was implemented in Matlab while using the simplex-based GLPK solver (GNU Linear Programming

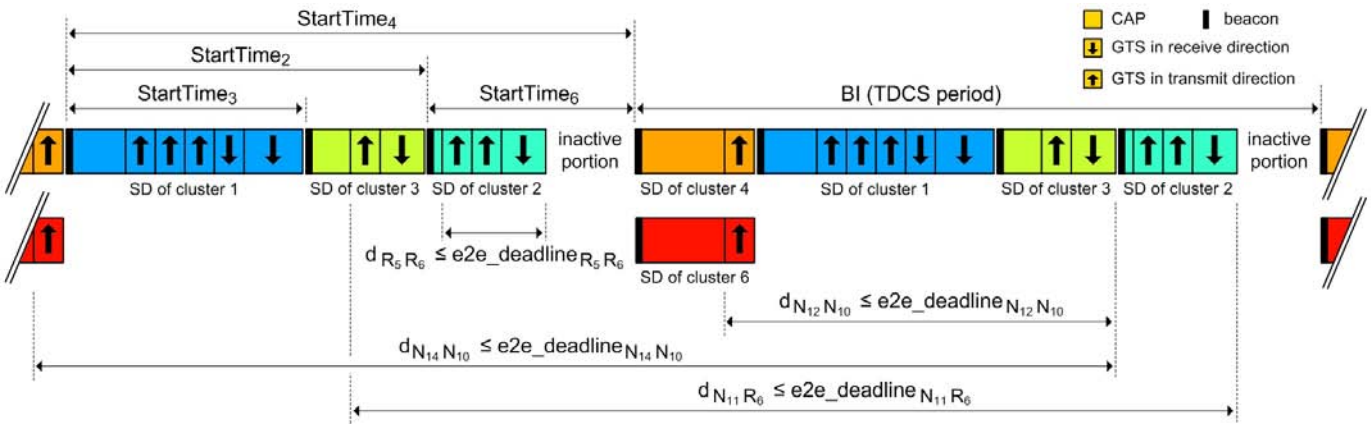


Fig. 11. Time line of TDCS corresponding to the example in Figure 1

Kit by A. Makhorin). To the best of our knowledge, so far no previous research has directly addressed the problem of energy efficient TDMA scheduling of time-bounded data flows in a cluster-tree WSN.

A. Time complexity

The time complexity usually depends on the number of decision variables, which is in this case less than $(n_c^2 - n_c)/2 + n_d$, where n_c stands for the number of active cluster-tasks (x_{ij} is generated for each couple of potentially conflicting tasks) and n_d stands for the number of dummy-tasks ($\hat{q}_i$ is generated for each of them).

In our experiment, the system model is configured as follows. The number of child routers is randomly generated for each parent router and varies from 0 to 3. The routers are successively generated until the total number of routers in the network reaches $total\_routers$. Each router has 3 child end-nodes. Note that the locations of child routers and end-nodes are randomly generated within the transmission range of their parent router ensuring random collisions. The total number of nodes ($total\_nodes$) in WSN is show in parentheses in the first column of Tab. III.

For each cluster-tree topology, we study effect of various number of data flows N_{flow} equal to 2 or 4 (or 8 for large networks), and the number of sources N_{source} of each data flow equal to 3 or 6. We fix the other parameters of data flows such as $req\_period = 1$ sec, $sample\_size = 120$ bits and $sample\_ack = 0$. For each combination of N_{flow} and N_{source}, we randomly generate a set of 20 instances and run the scheduling algorithm for each of them. Median of the number of tasks N_{task}, which represents the complexity of the problem, and median of the solution times (using the GLPK solver) are shown in Tab. III. The solution times, which exceed the time limit of 600 sec, are not encompassed in the median, and their number is shown in parentheses. The column $time_{compact}$ stands for the ILP formulation with objective function (8), which gives feasible and compact schedule while minimizing the sum of the start times. The column $time_{feasible}$ stands for the ILP formulation with objective function equal to 0 giving the feasible schedule in a shorter time. The results show that the complexity of the problem, given by N_{task}, grows with the number of flows and with the size of the network. On the other side, the number of sources of a given flow does not affect the complexity so much.

B. Simulation Study

This section demonstrates how the length of the TDCS period, given by the Beacon Order (BO), impacts the energy consumption of the nodes, using the simulation study based on the IEEE 802.15.4/ZigBee Opnet simulation model [30] that has been configured using the TDCS scheduling tool [17] presented in this paper. The simulation scenario consists of 14 TelosB motes forming a cluster-tree WSN as depicted in Fig. 1, and two time-bounded data flows with parameters defined in Tab. I. TelosB [31] is a battery-powered wireless device widely used in WSNs. The battery module of the simulation model estimates the energy consumption using the formula $U \cdot I \cdot t$ based on the execution time (t), the voltage (U), and current draw (I). The particular current draws were measured as follows: current draw in receive mode = 18.2 mA, transmit mode = 19.2 mA at 0 dBm, idle mode = 54.5 μA and sleep mode = 15 μA. New TDCS, which ensures that each data flow meets its e2e deadlines, is generated for each BO in range of BO_{min} to BO_{max}. Since the simulation model does not support the definition of the multiple collision domains, the non-overlapping TDCSs are assumed (i.e. a single collision domain). The simulation time of one run is equal to 40 minutes involving generation of 9582 frames in case of flow 1 and 4791 frames in case of flow 2.

In case of unacknowledged transmission ($macMaxFrameRetries = 0$), there exists three feasible TDCSs with the periods given by $BO = 3$, $BO = 4$ and $BO = 5$ (i.e. $BO_{min} = 3$ and $BO_{max} = 5$). Figure 12a confirms that all TDCSs are feasible, because the maximum end-to-end delays are shorter than the end-to-end deadlines ($e2e\_deadline$). Note that the dashed line at each column depicts the end-to-end deadline for a given sub-flow. Figure 12b shows the sum of energy consumption of all

$total\_routers$ ($total\_nodes$)	N_{flow}	N_{source}	N_{task}	$time_{compact}$ [sec]	$time_{feasible}$ [sec]
11	2	3	19	0.126	0.105
		6	19	0.137	0.1
(44)	4	3	31.5	0.408	0.19
		6	33.5	0.605	0.184
16	2	3	26.5	0.428	0.2 (1)
		6	27	2.75 (2)	0.22
(64)	4	3	40	8.94 (2)	0.63 (1)
		6	40.5	19.38 (2)	0.407 (2)
20	2	3	25	3.74 (2)	0.21
		6	24	9.54 (2)	0.21
(80)	4	3	44.43	–	0.63
		6	43.75	–	0.54
	8	3	97	–	6.14 (6)
		6	88.5	–	18.11 (8)
40	2	3	36.65	–	0.55 (1)
		6	34.90	–	0.51 (2)
(160)	4	3	66.30	–	29.75 (5)
		6	68.95	–	19.72 (3)
	8	3	111.5	–	56.38 (8)
		6	114	–	61.26 (10)
60	2	3	40.9	–	1.61
		6	40.15	–	1.33 (2)
(240)	4	3	73.75	–	7.68 (2)
		6	75.4	–	25.62 (4)
	8	3	154.5	–	61.1 (8)
		6	153	–	67.7 (11)

TABLE III
TIME COMPLEXITY OF TDCS ALGORITHM.

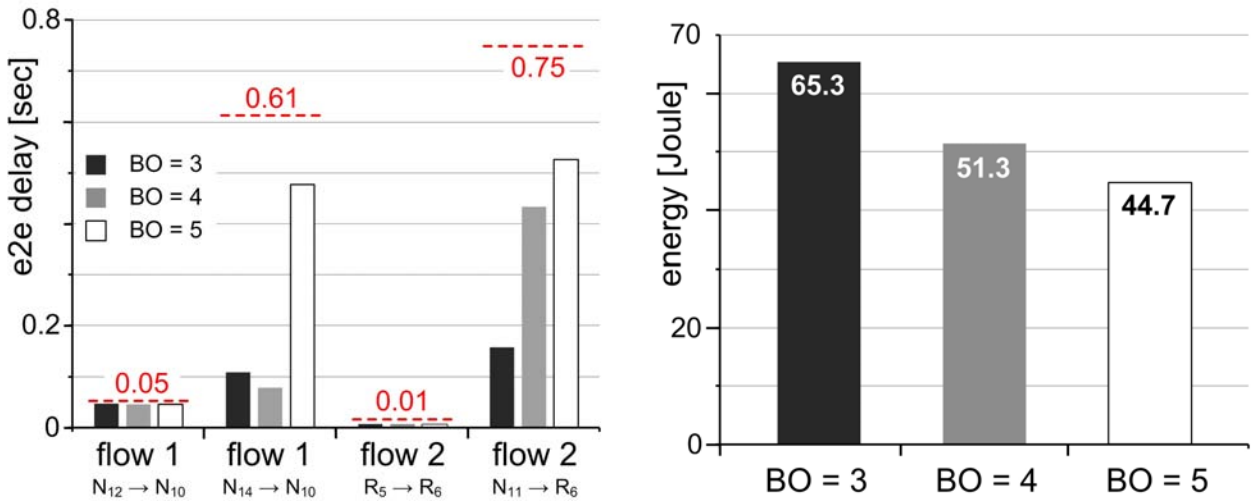
nodes within the simulation run as a function of BO. It can be observed that the nodes consume the minimum energy when the longest TDCS period ($BO = 5$) is applied. Hence, according to the required objectives, the TDCS scheduling tool returns the longest TDCS that meets all e2e deadlines while minimizing the energy consumption (i.e. maximizing the lifetime of the nodes).

VI. CONCLUSIONS

The paper shows how to minimize the energy consumption of the nodes by setting the beacon interval (TDCS period) as long as possible while respecting e2e deadlines of the flows and avoiding possible inter-cluster collisions. Binding of flows into one cluster-task is efficient with respect to the structure of superframe (dividing period BI into active portion and inactive portion) but also with respect to the complexity of the scheduling problem (volume of decisions is related to the square of potentially conflicting tasks). Note that grouping of GTSs in the transmit or receive direction leads to slightly pessimistic results (length of p_i^T or p_i^R is relatively short with respect to e2e deadline), but scheduling of separate GTSs would lead to dramatic increase of potentially conflicting tasks.

The solution is shown on iterative calls of the ILP algorithm, which gives precise mathematical formulation of the problem and shows acceptable performance for static configuration of middle-sized WSNs. Opposite direction of flows leads to cyclic formulation of the scheduling problem where one wave of a given data flow has to span over several periods. Thanks to the problem structure based on cyclic extension of RCPS/TC, it is quite straightforward to make cyclic extension of heuristic algorithms [26] that are able to handle RCPS/TC problems with up to one thousand of tasks in a few seconds.

An interesting issue to be investigated is the adaptive behavior of the scheduling problem when new tasks are added to the original schedule. Such a problem should be solvable by fixing the start times of original tasks and using the same optimization algorithm. The time complexity in such a case should be related to the number of new tasks, thus allowing one to use optimal solvers.



(a) Maximum e2e delay

(b) Energy consumption

Fig. 12. Maximum e2e delay and sum of energy consumption of all nodes as a function of BO assuming unacknowledged transmission.

APPENDIX

The following table reports the symbols that are used through the paper, along with their definition.

Symbol	Definition
$A = (a_{ij})$	adjacency matrix describing parent-child tree
BI	Beacon Interval
BO	Beacon Order
$C = (c_{ij})$	collision matrix
d_{ij}	delay
G	graph of communication tasks
n	number of tasks
p_i	processing time of task T_i
ptu	processing time unit
$\hat{q}_i$	segment of start time s_i
T_i	task i
s_i	start time of task T_i
$\hat{s}_i$	offset of start time s_i
$sample\_ack$	acknowledged or unacknowledged messages
$e2e\_deadline_{ij}$	bounded time between the instant when a source sends the message and the instant when the sink receives this message
req_period	maximal inter-arrival time between two consecutive measurements
$sample\_size$	size of sensing value
SD	Superframe Duration
SO	Superframe Order
$v_{i,j}$	offset weight
$w_{i,j}$	weight of the edge between tasks T_i and T_j
x_{ij}	binary decision variable

TABLE IV
TABLE OF SYMBOLS.

REFERENCES

[1] M. Chitnis, Y. Liang, J. Y. Zheng, P. Pagano, and G. Lipari, "Wireless Line Sensor Network for Distributed Visual Surveillance," in *Proc. of the 6th ACM International Symposium on Performance Evaluation of Wireless Ad Hoc, Sensor, and Ubiquitous Networks (PE-WASUN)*, Oct. 2009, pp. 71–78.

[2] A. Willig, "Recent and Emerging Topics in Wireless Industrial Communications: A Selection," *IEEE Trans. on Industrial Informatics*, vol. 4, no. 2, pp. 102–124, May 2008.

[3] M. Kohvakka, M. Hannikainen, and T. Hamalainen, "Wireless sensor network implementation for industrial linear position metering," in *Proc. of the 8th Euromicro Conference on Digital System Design (DSD)*, Sep. 2005, pp. 267–275.

[4] *IEEE P802.15 Wireless Personal Area Networks: Proposal for Factory Automation*, Working Draft Proposed Standard, Rev. 802.15.4-15/08/0571r0, 2009.

[5] *ZigBee Specification*, ZigBee Standards Org. Std. 053 474r13, 2006.

[6] T. He, J. A. Stankovic, C. Lu, and T. F. Abdelzaher, "A Spatiotemporal Communication Protocol for Wireless Sensor Networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 16, no. 10, pp. 995–1006, Oct. 2005.

[7] P. Jurčík, R. Severino, A. Koubâa, M. Alves, and E. Tovar, "Real-Time Communications over Cluster-Tree Sensor Networks with Mobile Sink Behaviour," in *Proc. of the 14th IEEE International Conf. on Embedded and Real-Time Computing Systems and Applications (RTCSA)*, Aug. 2008.

[8] *Part 15.4: wireless medium access control (MAC) and physical layer (PHY) specifications for low-rate wireless personal area networks (LR-WPANs)*, IEEE SA Standards Board Std. 802.15.4, 2006.

[9] W. Ye, J. Heidemann, and D. Estrin, "An Energy-Efficient MAC protocol for Wireless Sensor Networks," in *Proc. of the 21st Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, Jun. 2002, pp. 1567–1576.

[10] O. Chipara, Z. He, G. Xing, Q. Chen, X. Wang, C. Lu, J. Stankovic, and T. Abdelzaher, "Real-time Power-Aware Routing in Sensor Networks," in *Proc. of the 14th IEEE International Workshop on Quality of Service (IWQoS)*, Jun. 2006, pp. 83–92.

[11] J. Trdlička, M. Johansson, and Z. Hanzálek, "Optimal Flow Routing in Multi-hop Sensor Networks with Real-Time Constraints through Linear Programming," in *Proc. of the 12th IEEE International Conf. on Emerging Technologies and Factory Automation (ETFA)*, Sep. 2007, pp. 924–931.

[12] E. Toscano and L. LoBello, "A topology management protocol with bounded delay for wireless sensor networks," in *Proc. of 13th International Conf. on Emerging Technologies and Factory Automation (ETFA)*, Sep. 2008, pp. 942–951.

[13] L. LoBello and E. Toscano, "An Adaptive Approach to Topology Management in Large and Dense Real-Time Wireless Sensor Networks," *IEEE Trans. on Industrial Informatics*, vol. 5, no. 3, pp. 314–324, Aug. 2009.

[14] R. S. Oliver and G. Fohler, "Probabilistic Estimation of End-to-End Path Latency in Wireless Sensor Networks," in *Proc. of the 6th IEEE International Conf. on Mobile Ad-hoc and Sensor Systems (MASS)*, Oct. 2009, pp. 423–431.

[15] A. Koubâa, A. Cunha, M. Alves, and E. Tovar, "TDBS: a time division beacon scheduling mechanism for ZigBee cluster-tree wireless sensor networks," *Real-Time Systems Journal*, vol. 40, no. 3, pp. 321–354, Oct. 2008.

[16] R. Holte, A. Mok, L. Rosier, I. Tulchinsky, and D. Varvel, "The pinwheel: A real-time scheduling problem," in *Proc. of the 22nd Hawaii International Conf. on System Sciences (HICSS)*, Jan. 1989, pp. 693–702.

[17] P. Jurčík and Z. Hanzálek. (2010) TDCS Matlab tool. [Online]. Available: http://rtime.felk.cvut.cz/tdcs

[18] R. Diestel, *Graph Theory*. Springer-Verlag, 2005.

[19] H. Bai and M. Atiquzzaman, "Error modeling schemes for fading channels in wireless communications: A survey," *IEEE Commun. Surveys Tuts.*, vol. 5, pp. 2–9, 2003.

[20] C. de M. Cordeiro and D. Agrawal, *Ad Hoc Sensor Networks: Theory and Applications*. World Scientific, 2006.

[21] K. Neumann, C. Schwindt, and J. Zimmermann, *Project Scheduling with Time Windows and Scarce Resources*. Springer, 2003.

[22] P. Brucker, A. Drex, R. Möhring, K. Neumann, and E. Pesch, "Resource-constrained project scheduling: Notation, classification, models, and methods," *European Journal of Operational Research*, vol. 112, no. 1, pp. 3–41, 1999.

[23] B. de Dinechin, "Simplex scheduling: More than lifetime-sensitive instruction scheduling," in *Proc. of the International Conf. on Parallel Architecture and Compiler Techniques*, 1994, pp. 327–330.

[24] R. Alur and D. Dill, "A theory of timed automata," *Journal of Theoretical Computer Science*, vol. 126, pp. 183–235, 1994.

[25] W. Herroelen, B. Reyck, and E. Demeulemeester, "Resource-constrained project scheduling : A survey of recent developments," *Computers and operations research*, vol. 25, no. 4, pp. 279–302, 1998.

[26] Z. Hanzálek and P. Šůcha, "Time Symmetry of Project Scheduling with Time Windows and Take-give Resources," in *Proc. of the 4th Multidisciplinary International Scheduling Conf.: Theory and Applications (MISTA)*, Aug. 2009.

[27] C. Hanen and A. Munier, "A study of the cyclic scheduling problem on parallel processors," *Discrete Applied Mathematics*, vol. 57, pp. 167–192, February 1995.

[28] A. M. Kordon, "A graph-based analysis of the cyclic scheduling problem with time constraints: schedulability and periodicity of the earliest schedule," *Journal of Scheduling*, Feb. 2010, DOI: 10.1007/s10951-009-0159-z (online).

[29] S.-A.-A. Touati and Z. Mathe, "Periodic register saturation in innermost loops," *Journal of Parallel Computing*, vol. 35, no. 4, pp. 239–254, Apr. 2009.

[30] P. Jurčík and A. Koubâa. (2009) IEEE 802.15.4/ZigBee Opnet Simulation Model v3.0. [Online]. Available: http://www.open-zb.net

[31] Crossbow Technology, Inc. (2009) TelosB Mote Datasheet. [Online]. Available: http://www.xbow.com