Utilization-Aware Hybrid Beacon Scheduling in Cluster-Tree ZigBee Networks

Junghee HAN\(^{(a)}\), Member, Jiyong HAN\(^{(b)}\), Dongseup LEE\(^{(c)}\), and Changgun LEE\(^{(d)}\), Nonmembers

SUMMARY In this paper, we propose an utilization-aware hybrid beacon scheduling method for a large-scale IEEE 802.15.4 cluster-tree ZigBee network. The proposed method aims to enhance schedulability of a target network by better utilizing transmission medium, while avoiding inter-cluster collisions at the same time. To achieve this goal, the proposed scheduling method partially allows beacon overlaps, if appropriate. In particular, this paper answers for the following questions: 1) on which condition clusters can send overlapped beacons, 2) how to select clusters to overlap with minimizing utilization, and 3) how to adjust beacon parameters for grouped clusters. Also, we quantitatively evaluate the proposed method compared to previous works—i.e., non-beacon scheduling and a serialized beacon scheduling algorithm—from several aspects including total duty cycles, packet drop rate, and end-to-end delay.

key words: IEEE 802.15.4, ZigBee, collision-free, beacon scheduling, overlap, schedulability

1. Introduction

With the emergence of smart and converged services, there has been rapid increase in the need for multi-functional sensor nodes deployed in many fields including health system, battle field, digital home network, and factory monitoring. IEEE 802.15.4/ZigBee\(^1\), \(^2\) is a global standard for such emerging applications for low-power, low-cost and low-rate wireless transmission. IEEE 802.15.4/ZigBee supports star, peer-to-peer, and cluster-tree topologies. The cluster-tree topology is a special case of a peer-to-peer topology, and it provides better scalability compared to a star topology. This paper targets to the cluster-tree ZigBee networks.

A cluster-tree ZigBee network\(^3\) is composed of multiple clusters, each of which has a cluster head (also called router or coordinator), and end devices. Among these cluster heads, there is one main coordinator, called PAN. In many cases, the PAN acts as a common sink node and is connected to a server computer, and, in turn, routers and/or devices are connected with PAN. Each end device is allowed to transmit data when it receives a beacon from the router in the cluster. For example, a node N2 in a cluster 1 of Fig. 1 can transmit data only when it receives beacon from its router, N9. In such cluster-tree networks, we should be aware that clustering is just logical grouping of nodes and so two nodes in different clusters can physically interfere with each other. In other words, when two physically nearby clusters transmit beacons at the same time, beacon and data transmission in these clusters can be corrupt. Hence, when beacons in different clusters are not carefully scheduled, it can cause a huge amount of retransmission and eventually packet loss.

To avoid such inter-cluster collisions, many beacon scheduling algorithms have been proposed\(^4\)--\(^6\), \(^8\)--\(^10\). The previous works solved this inter-cluster collision problem by serializing all beacon frames, not allowing overlaps of beacon frames among clusters. However, such serialized beacon scheduling can accommodate only small size cluster networks because this scheme allows only one cluster over the whole network to transmit beacon/data at a time. This results in a waste of resources unnecessarily, and even further it can conservatively determine the target network to be “not-schedulable”. In summary, non-scheduled beacon scheme is one extreme and the serialized beacon scheduling is the other extreme in the sense that the former scheme might initiate beacons of all clusters at the same time but the latter scheme organizes beacons in a serial way without allowing any overlap among them.

Based on this observation, this paper proposes a hybrid beacon scheduling algorithm—it carefully schedules beacons to enhance schedulability of a target network by better utilizing network resources while it keeps the target cluster-tree network collision-free. To achieve this goal, the proposed scheme allows beacon overlaps among clusters, while minimizing interference among clusters. Specifically, the proposed scheme 1) first defines overlapping conditions, 2) selects a subset of clusters to be overlapped in an “optimal” way with minimizing a total duty cycle, and 3) adjusts beacon intervals and superframe durations for the overlapped

---

\(^{1}\)The author is with Korea Aerospace University, Korea.

\(^{2}\)The author is with KAIST, Korea.

\(^{3}\)The author is with Korean Air Co, Ltd., Korea.

\(^{4}\)The author is with Seoul National University, Korea.

\(^{(a)}\)E-mail: junghee@kau.ac.kr

DOI: 10.1587/transinf.2014EDP7379
clusters to make the target network be schedulable. We perform detailed simulations and present in-depth analysis to quantitatively compare non-scheduled beacon scheme, serialized beacon scheduling, and the proposed hybrid scheme from several aspects of interests.

The rest of the paper is organized as follows. Section 2 overviews the background and related works. Section 3 presents the problem statement and Sect. 4 introduces the proposed algorithm. Section 5 provides details of simulation results. Finally, the paper concludes with Sect. 6.

2. Related Works and Background

In this section, we present related works and also describe background information about a beacon frame structure and a serialized SDS algorithm.

2.1 Related Works

Despite importance of low-cost and low-power wireless sensor networks, many wireless sensor network systems, however, turn out to be ineffective. Some are too complicated and too expensive for pervasive deployment, and some stop working too quickly due to the rapid depletion of their batteries. To address this issue, in the ubiquitous and embedded system research area, many researchers including our group have devoted their efforts to develop power-efficient sensor networks[11]–[13] and to support reliable and real-time communications between embedded sensor devices [14]–[16]. [11]–[13] attempted to maximize the life-time of wireless sensor network. The lifetime of the network is determined by the maximum of per-node power consumption. If any single node runs out its power, the entire system might not be working fine any more. Hence, to lengthen the life-time of the entire system, we should minimize the maximum of per-node consumption. On the other hand, [14]–[16] have proposed several methods to guarantee the end-to-end delivery of data packets with time constraints. A large number of sensor network applications including battlefield and disaster rescue systems deal with time-critical information so that they can react to the physical world on-time.

All of these previous works rely on the assumption that underlying MAC layer data transmission would be well “scheduled” and so MAC layer transmission is highly reliable without collision. Several works [4]–[7], [10], [19] have attempted to provide collision-free data transmissions. [7], [10] addressed this issue by scheduling beacons in a local and probabilistic way at each device. But it cannot totally avoid collision though. We explain [4]–[6] in detail in Sect. 2.3.

2.2 Beacon Frame Structure of IEEE 802.15.4

IEEE 802.15.4 MAC [1] protocol supports two modes: Non-beacon-enabled and beacon-enabled modes. At beacon-enabled, coordinators periodically transmit beacon frames and end devices transmit data based on the received beacon frame. As shown in Fig. 2, a beacon frame is divided into two parts: Beacon Interval (BI) and Superframe Duration (SD). BI is an interval from one beacon frame to the next beacon frame. SD starts following a beacon frame. SD is an active part during which each node belonging to the corresponding cluster becomes active and transmits data using CSMA/CA and GTS (Guaranteed Time Slots). In [1], BI and SD are formally defined as follows:

$$BI = aBaseSuperframeDuration \ast 2^{BO}$$
$$SD = aBaseSuperframeDuration \ast 2^{SO}$$

The ranges of BO (BI Order) and SO (SD Order) are $0 \leq SO \leq BO \leq 14$ and the $aBaseSuperframeDuration$ is 15.36 ms when frequency is 2.4GHz and data rate is 250 Kbps.

2.3 Superframe Duration Schedule (SDS)

To avoid inter-cluster collisions, [4]–[6] proposed the serialized superframe duration beacon scheduling scheme (called SDS). In this serialized SDS approach, only one router node transmits a beacon at a time so that a beacon frame does not interfere others. It also organizes beacon frames not to overlap with the data frames of other clusters as well. Hence, the SDS approach serializes both BIs and SDs in a non-overlapping way. For example in Fig. 3, when beacon frames B1 and B2 of cluster 1 and cluster 2 are sent at the same time (i.e., $t_1$), then two beacons collide with each other. But, if B2 is delayed by an offset, then BIs and SDs are not overlapped and so they can avoid inter-cluster collisions.

However, such a serialized beacon scheduling method is too conservative in the sense that only one cluster over the whole network can transmit a beacon at one time. Due to this feature, the serialized scheme can pessimistically determine a certain cluster-tree network to be “infeasible” (“not
schedulable”). For instance, [4] presents its schedulability condition that if the total sum of duty cycles (i.e., $U_{total}$) is bigger than 1, then the target cluster-tree network is determined to be not-schedulable as shown in Eq. (2).

$$\text{Total Utilization } U_{total} = \sum_{k \in \text{clusters}} \frac{SD_k BI_k}{BI_k} > 1$$

However, if some overlaps are permitted, the total utilization becomes lowered and so a target network has more chance to be schedulable. To address this issue, this paper proposes a hybrid beacon scheduling algorithm. This argument will be examined in more detail in the next section.

3. Problem Statement

In this section, we illustrate the target problem to solve using an example. Given a cluster-tree ZigBee network as in Fig. 4, suppose a scenario that BI and SD values are assigned as in Table 1. With the given scenario, all clusters might launch beacons at the same time in non-scheduling algorithm as shown in the upper part of Fig. 5. In contrast, the original SDS method presents the non-overlapping serialized schedule as in the middle part of Fig. 5.

Both non-scheduling and SDS algorithms have their own drawbacks. First, all clusters in non-scheduling method might send beacons at the same and so inter-cluster collision among neighborhood clusters such as cluster 0 and 2 in Fig. 4 can happen. Such collisions can generate a huge number of packet drops especially in a crowded network and also lengthen end-to-end delay. In the other extreme, the serialized SDS shows collision-free beacon scheduling while occupying more time slots. As shown in the middle part of Fig. 5, SDS results in a less amount of idle slots than non-scheduling scheme. This drawback is magnified when a total utilization by SDS is greater than 1. In contrast to SDS, non-scheduling scheme or cluster-overlapping method (i.e., the proposed hybrid scheme) can reduce the total utilization and so the target network can become schedulable. Furthermore, SDS might not be able to additionally admit a new cluster. For example in Fig. 5, SDS cannot admit a new cluster with $BI=16$ and $SD=4$ because admitting this additional cluster makes the network become non-schedulable. On the contrary, both non-scheduling and overlapping scheme can admit this cluster without any schedulability problem. From schedulability’s point of view, SDS is less applicable than non-scheduling and cluster-overlapping scheme especially when a total utilization is high.

To solve the aforementioned problems of non-scheduling and the serialized SDS schemes, this paper proposes a hybrid beacon scheduling algorithm, which allows partial overlaps. By partially allowing overlap between clusters, the proposed hybrid scheduling scheme complements both non-scheduling and SDS algorithms. For example, a lower part in Fig. 5 presents the beacon schedule produced by the proposed hybrid scheme. First of all, the total duty cycle has been lowered from 18/32 to 14/32. Thanks to such slot overlapping, the hybrid scheduling algorithm can admit the above additional cluster without hurting schedulability. Furthermore, by adjusting “overlapping conditions”, the proposed hybrid algorithm is able to allow more aggressive overlapping while minimizing conflicts. This feature makes the proposed algorithm is more applicable to a large-scale cluster-tree networks. At the same time, it also keeps the target cluster-tree network to be collision-free as much as SDS by carefully overlapping clusters.

4. Proposed Method

The main idea of the proposed approach is based on the observation that when two clusters are “physically separated”, they can be overlapped without collision. This concept seems to be simple and straightforward. However, instead of just mentioning the anecdotal idea, this paper presents the complete method with 1) identifying overlapping conditions on which subsets of clusters can send beacons at the same
time, 2) proposing an optimal way of grouping to minimize total duty cycles, and 3) developing a way for assigning new BI and SD for the grouped clusters to maximize benefits of overlapping.

4.1 On which Conditions Clusters Can Be Grouped?

To answer this question, we define the two overlapping conditions for cluster grouping: one for collision-free feature and the other for duty cycle reduction. The first overlapping condition is as follows.

**Overlapping Condition 1** (conflict-free): Let \( a \) and \( b \) be router nodes of cluster \( A \) and \( B \). Then, cluster \( A \) and \( B \) can be overlapped if \( D_{(a,b)} \geq \max(r_a, r_b) + \alpha \) where \( 0 \leq \alpha \leq \min(r_a, r_b) \).

This condition intuitively implies that two clusters \( A \) and \( B \) should be “far enough” to each other. In this condition, \( D(a, b) \) is correspondent to the physical distance between two ZigBee router nodes \( a \) and \( b \) of cluster \( A \) and \( B \), respectively. \( r_a \) and \( r_b \) are transmission ranges of node \( a \) and \( b \). The value of \( \alpha \) is controlling the amount of overlaps. When it is large enough, overlapping of the two clusters is safe in the sense that they are sufficiently far from each other and so their RF signal would not interfere with each other. However, if \( \alpha \) is too large, it would be too conservative to find a set of clusters satisfying this condition. On the other hand, if it is too small, then overlapping of two clusters has high chance of collision. In the extreme, if \( \alpha = 0 \), then we can only avoid direct collisions between two ZigBee routers but they can suffer indirect collisions when one of the routers has its child in the common area such as \( D_1 \) in Fig. 6 (a). Note that the direct collision happens when a ZigBee router is in the range of its neighbor routers while the indirect collision occurs even when the routers cannot directly hear each other if there exists a child node in the two routers’ common radio range [4]. To avoid such indirect collisions, we assign \( \min(r_a, r_b) \) to \( \alpha \) so that we can ensure two routers are far from each other by at least \( r_a + r_b = \max(r_a, r_b) + \min(r_a, r_b) \).

The bottom line is that we control the \( \alpha \) value depending on how much we need to overlap clusters at the cost of collision risk. Specifically, if a total utilization is \( < 1 \) and the network is schedulable, we assign \( \alpha \) the maximum value (i.e., \( \min(r_a, r_b) \)) in order to avoid both indirect and direct collisions. In contrast, if the target network is overly crowded and “not schedulable”, then we need to reduce \( \alpha \) value in order to loosen the overlap condition. By doing it, we can allow more amount of overlaps but we might experience more collisions. The optimal value of \( \alpha \) can vary depending on BI and SD parameters of clusters for individual target network. The algorithm in Sect. 4.4 provides the procedural way so that we can determine the \( \alpha \) value for each target network.

Now, we introduce the second overlapping condition by illustrating an example in Fig. 7. Suppose we group cluster 1 and 2 in Table 1. In this case, BI and SD of one cluster cannot be subsumed into another cluster—BI of cluster 1 is larger than cluster 2 and SD of cluster 1 is larger than cluster 2. In other words, cluster 1 transmits a bulk of data but less frequently while cluster 2 sends a small amount of data but more frequently. In this case, we should assign BI and SD for the group in order not to violate the original BIs and SDs of cluster 1 and cluster 2. Specifically, we should assign a long SD (for bulk data) and small BI (for more frequent transmission) for the group. However, we get all the worse for such overlapping—total duty cycle gets increased from 29/32 to 33/32 after grouping and so this target network becomes non-schedulable. In this case, grouping of these two clusters does not give us any gain and makes things even worse. To reflect this case, we come up with another overlapping condition as below. With this condition, we allow overlapping only when overlapping gives us duty cycle gain.

**Overlapping Condition 2** (total duty cycle gain): Let \( BI_a \) and \( BI_b \) be beacon intervals of clusters \( A \) and \( B \), respectively. Similarly, let \( SD_a \) and \( SD_b \) be superframe durations of clusters \( A \) and \( B \), respectively. Then, cluster \( A \) and \( B \) cannot be overlapped if \( BI_a > BI_b \) and \( SD_a > SD_b \).

4.2 How to Assign BI and SD for the Grouped Clusters?

For further beacon scheduling, we need to assign new values of BI and SD to the grouped clusters. Specifically, we treat the grouped clusters as a single virtual cluster for scheduling purpose. Thus, in this proposed approach, we “carefully” assign new BI and SD for each group (i.e., a virtual cluster) to satisfy the following constraints.

- The total duty cycle (utilization value in Eq. (2)) should be decreased after assigning new BI and SD values.
- The original BIs and SDs of clusters should be kept—i.e., the newly assigned BI and SD of the group should...
subsume the original BIs and SDs of its clusters inside.

To satisfy these constraints, we come up with the following rules. Suppose that cluster \(i\) and cluster \(j\) are grouped together.

1. **When** \(SD_i = SD_j\): \(BI_{\text{group}}\) is assigned to be the smaller of \(BI_i\) and \(BI_j\). Otherwise, a cluster with a smaller \(BI\) cannot be subsumed into a new beacon interval of the group and so it might collide with others. Of course, \(SD_{\text{group}} = SD_i = SD_j\). Grouping of cluster 1 and cluster 6 in Fig. 5 is an example of this case.

2. **When** \(BI_i = BI_j\): Analogically, \(SD_{\text{group}}\) is assigned to be the bigger of \(SD_i\) and \(SD_j\). Otherwise, a cluster with a bigger \(SD\) value cannot be properly scheduled and so collide with others. Of course, \(BI_{\text{group}} = BI_i = BI_j\).

3. **When** \(BI_i > BI_j\) and \(SD_i \leq SD_j\): In this case, both beacon and SD of cluster \(i\) can be subsumed into cluster \(j\)'s schedule, just like the cluster 2 and cluster 5 in Fig. 5. Hence, \(BI_{\text{group}} = BI_i\) and \(SD_{\text{group}} = SD_j\).

4. **When** \(BI_i > BI_j\) and \(SD_i \geq SD_j\): This case violates the 2nd overlapping condition in Sect. 4.1 because \(BI\) and \(SD\) of one cluster are not subsumed into the other and so grouping of these clusters lengthens duty cycles. Hence, a pair of clusters falling into this category cannot be grouped together.

After ruling out the fourth case from our grouping candidates, we obtain the generalized \(BI\) and \(SD\) assignment rules as follows for assigning \(BI_{\text{group}}\) and \(SD_{\text{group}}\) when cluster \(i\) with \(BI_i\) and \(SD_i\) and cluster \(j\) with \(BI_j\) and \(SD_j\) are grouped.

\[
\begin{align*}
\text{If } & SD_i \geq SD_j \quad \text{and} \quad BI_i > BI_j, \quad \text{then} \quad BI_{\text{group}} = BI_i, \quad (3) \\
& SD_{\text{group}} = SD_i, \quad (4) \\
& BI_{\text{group}} = BI_j, \quad (5)
\end{align*}
\]

Once \(BI_{\text{group}}\) and \(SD_{\text{group}}\) of a grouped cluster has been determined as above, each cluster belonging to the group has an option to adjust its own \(BI\) and \(SD\) values. For example, cluster 3, 5, 7 are grouped as shown in the low part of Fig. 10 and their group's BO and SO are determined as 16 and 2, respectively. In this example, cluster 7 can adjust their BO from 32 to 16 without affecting any other nodes. By doing this, cluster 7 can shorten its BI and hence get better end-to-end delays. We will examine this effect in Sect. 5.

### 4.3 How to Select Groups?

Once we define two overlapping conditions, and \(BI\) and \(SD\) assignment rules, we now need to select subset(s) of non-interfering clusters for grouping. At a glance, we might think of simply applying a graph coloring algorithm for this purpose [17], [18]. However, reducing the number of colors to cover the whole graph—i.e., simply maximizing the overlapping among clusters—is not the objective of the beacon overlapping algorithm in this paper. Instead, minimizing the total duty cycle is more important.

Hence, in this section we present an ILP (Integer Linear Programming) based cluster selection algorithm with minimizing total utilization. Specifically, we here develop utilization-aware ILP formulation as below (from Eq. (6) to Eq. (11)). The proposed formulation aims to achieve our goal of minimizing the total duty cycles. Specifically, in the proposed ILP algorithm, the objective function is to minimize total duty cycle while the objective function in the original graph coloring is to minimize the number of color assignments.

In particular, we first convert the target cluster-tree network to a graph, \(G\), as shown in Fig. 8. Let \(G = (V, E)\) be an undirected graph where each node \(v\) in the graph corresponds to each cluster. Each edge \(e\) between two nodes \(u\) and \(v\) represents that two clusters corresponding to the two nodes are violating at least one of two overlapping conditions explained in Sect. 4.1. For example, the cluster 1 and cluster 2 have an edge between them because these two clusters do not satisfy the first condition. Also cluster 1 and cluster 7 have an edge as well because this pair of clusters violates the second overlapping condition. On the other hand, there is no direct edge between cluster 2 and cluster 6, because these two clusters satisfy both condition 1 and 2. Specifically, all nodes belonging to cluster 2 are sufficiently far from any node belonging to cluster 6, which satisfies the condition 1. Also grouping of these two clusters makes a
total duty cycle lower, which satisfies the condition 2. Over this graph, the proposed ILP formulation attempts to group nodes in order to minimize total utilization level as follows.

\[
\text{minimize } \sum_{g \in K} U_g
\]

which subjects to

\[
x_{v,g} = \begin{cases} 
1, & \text{if cluster } v \text{ belongs to group } g \\
0, & \text{otherwise}
\end{cases} \tag{7}
\]

\[
\sum_{g=1}^{k} x_{v,g} = 1, \forall \text{ cluster } v \in V \tag{8}
\]

\[
x_{u,g} + x_{v,g} \leq 1, \forall \text{ edge } (u, g) \in E \tag{9}
\]

\[
U_g \geq 0, \forall \text{ group } g \in k \tag{10}
\]

\[
U_g \geq x_{v,g} \cdot \frac{SD_v}{BI_g} \cdot \text{MaxBI} \tag{11}
\]

\[
V \text{ group } g \in K, \forall \text{ cluster } v \in V
\]

Recall that any single node is assigned to only one group, not more than one. So the maximum number of groups is the same as the total number of nodes—this maximum case happens when no grouping is performed and so each cluster belongs to a different group (i.e., different color in a graph coloring algorithm). Reflecting this fact, the proposed ILP-based algorithm defines \( K \) as a set of groups with a size \( k = |V| \). Some groups would consist of more than one clusters and on the other hand some groups would not have any cluster at all. Eq. (6) presents the objective function of the proposed optimization formulation. \( U_g \) is the duty cycle of a group \( g \) and we attempt to minimize the total sum of the duty cycles of groups. If a group consists of more than one nodes, a duty cycle of a group is defined based on Eq. (4) and (5). When a group does not have any cluster in it, its duty cycle is 0, of course. There are five constrains in the formulation. The first and second ones (Eq. (7) and (8)) correspond to the fact that each cluster should belong to only one group. The third constraint enforces that two clusters \( u \) and \( v \) can belong to the same group \( g \) only if they are satisfying both of two overlapping conditions in Sect. 4.1. The fourth and fifth constraints reflect that the duty cycle of each group is defined as a maximum of \( SD/BI \) values of clusters belonging to the group, as defined in Eq. (4) and (5). To implement the concept of “maximum” in ILP, we use the constraints in Eq. (10) and (11). Also, to make the duty cycle, \( U_g \), be an integer value, we multiply it by MaxBI, which is a LCM (Least Common Multiple) of \( BI \)s of all clusters. This MaxBI value is pre-known and so considered as a constant value.

In summary, the proposed grouping method has two unique features:

- We define two types of edges in the graph representing two overlapping conditions: 1) beacon conflict and 2) duty cycle overhead.
- We develop utilization-aware ILP formulation to minimize total duty cycles

We explain the advantages of the proposed grouping algorithm by illustrating an example of Fig. 8. \( BI \)s and \( SD \)s of clusters in Fig. 8 are given as in Table 1 of the previous section. Two figures in Fig. 9 and Fig. 10 show grouping results and corresponding total duty cycles after applying an original graph-coloring and the proposed grouping method, respectively. Note that the objective function of the original graph coloring algorithm is to minimize the number of colors (i.e., groups). From this point of view, the graph in Fig. 9(a) is good as much as the graph of Fig. 9(b). because both graphs have the same number of colors. Hence, the original graph coloring method might produce the left graph. In contrast, the utilization-aware ILP method in this paper produces the grouping result of Fig. 9(b) because the right graph provides lower utilization than the left graph. Figure 10 explores this fact in more detail by examining cluster schedules generated as a consequence of each grouping method. The top, middle, and bottom parts in Fig. 10 represent beacon schedules of 1) the serialized SDS, 2) hybrid grouping with the original graph coloring method, and 3) hybrid grouping with utilization-aware ILP-based method, respectively.

We observe that without grouping, the total duty cycle is 18/32 while two hybrid grouping algorithms with an original graph-coloring and ILP-based methods reduce total cycles to 14/32 and 11/32, respectively. Furthermore, from utilization’s perspective, the proposed ILP-based grouping is the best among the three schedules. So, from now on, “hybrid beacon scheduling” in this paper refers to the hybrid beacon algorithm with the proposed utilization-aware ILP-based grouping method, unless otherwise specified.

4.4 Hybrid Beacon Scheduling Algorithm

Overall, the proposed utilization-aware hybrid scheduling algorithm is summarized as follows.

5. Evaluations

In summary, the proposed method aims to satisfy two objectives of 1) lowering a total duty cycle (i.e., maximizing resource reuse) and 2) minimizing collision risks. On the
Set the value of $\alpha$ to the maximum, (i.e., $\alpha = \min(r_a, r_b)$)

1. Determine schedulability of the target clustered network
2. If non-schedulable
3. Determine if two clusters interfere with each other for all possible pairs of clusters (overlapping condition 1)
4. Determine if or not overlapping of two clusters give us duty cycle gain (overlapping condition 2)
5. Perform ILP solver to obtain a set of clusters to be grouped
6. Assign BI and SD for each group (refer to Sect. 4.2)
7. Check if the overlapped clusters is schedulable. If not, decrease the value of $\alpha$ and goto step 3.
8. Perform beacon scheduling over the grouped network

other hand, non-scheduling and the serialized SDS pursue only one of the objectives at the cost of the other one.

To quantitatively evaluate the proposed method, we conduct two sets of simulations. In the first set, we examine the total duty cycles of the proposed ILP-based hybrid scheduling and the original SDS scheduling. In the second set, we measure and compare the amount of collisions for non-scheduling, the original SDS, and the proposed ILP-based hybrid scheduling methods.

5.1 Total Duty Cycle

First, we measured total duty cycles for 10000 random samples. All of 10000 samples are based on the same topology as in Fig. 4—the same number of nodes and the same connection among them. For each sample, we randomly selected scheduling parameters (i.e., BI and SD values) and measured a total duty cycle of each sample based on Eq. (2). From the simulation results in Fig. 11, we observed that the proposed hybrid scheduling method produces a lot lower total duty cycles than the serialized SDS. In Fig. 12, we detailed this observation by quantitatively analyzing the degree of “improvement” based on the following metric.

$$\text{improvement}(\%) = (1 - \frac{\text{dutycycle}_{\text{HYBRID}}}{\text{dutycycle}_{\text{SDS}}}) \times 100 \quad (12)$$

From this analysis, we found that most samples got a significant amount of total duty cycle improvement by applying the proposed hybrid scheduling method. This implies that the hybrid scheduling algorithm is better at resource reuse and so it provides better schedulability.

5.2 Collision

We also compared the proposed approach with non-scheduling and SDS algorithms from collision’s perspective. To measure the amount of collisions, we use three metrics, 1) packet drop, 2) retransmission, and 3) end-to-end delay.

Specifically, we conducted Omnet++ simulations over two test sets as shown in Fig. 13: one sparse network with a small number of end devices and the other one with many end devices. Both test sets are basically based on the same cluster topology described on Fig. 4. Only the number of end devices is different. For both test sets, we used the same BI and SD values as presented in Table 1. Detailed simulation parameters are described in Table 2.

In the case of non-scheduling algorithm, all coordinators transmit beacons at the same time. In contrast, the SDS scheduling forces each coordinator to send beacons once at a time as shown in the top part of Fig. 10. As described in the previous section, the proposed hybrid scheme forms two groups: (cluster 7, cluster 5, cluster3) and (cluster 2, cluster 6). The proposed ILP-based grouping algorithm chose
these two sets of clusters because overlapping of these clusters maximizes total duty cycle saving compared to the serialized SDS while it still prevents beacon conflicts. In this hybrid scheme, each cluster is scheduled to transmit beacons as shown in the lowest part of Fig. 10.

(1) Packet Drop Rate

Table 3 shows packet drop rates of each algorithm over two test sets. Most importantly, we observed that the packet drop rate of the proposed hybrid approach is very close to the SDS. This observation confirms our argument that the proposed algorithm keeps collision-free feature while it lowers a total duty cycle. Also, as expected, non-scheduling shows the highest packet drop rate because beacon transmissions among nearby clusters would interfere with each other. However, we noticed that the packet drop rate of non-beacon scheduling is not that significant in Test 1, but huge in Test 2. This can be explained by the fact that Test 1 is a less crowded network, which consists of a small number of end devices and so each device can have more chances to successfully send packets via several re-trials (i.e., retransmission). To further explore this issue, we present another analysis about retransmission.

Finally, we found that even SDS and Hybrid algorithms suffer from a non-negligible amount of packet drops in Test 2. This is because we use CSMA/CA transmission mode in this simulation. In CSMA/CA mode, each device competes for network resources with its neighbor end devices in the same cluster. Hence, in a crowded network like Test 2, there is more chances for a node to fail in data transmission due to the nature of CSMA/CA within a cluster, not due to inter-cluster beacon collision. This issue is further analyzed in the next analysis as well.

(2) Retransmission

For further analysis, we examined an amount of retransmission packets of the three algorithms. Figure 14 and 15 show the number of retransmission packets observed at each end device of Test 1 and Test 2, respectively. As we expected, each end device in non-scheduling suffered a significant amount of retransmission while nodes in SDS and Hybrid methods experience little retransmission. Especially, nodes in Test 1 with Hybrid and SDS do not suffer any retransmission. In Test 2, all of none-scheduling, SDS, and Hybrid methods show some retransmission behavior. This is due to the nature of CSMA/CA. To totally exclude CSMA/CA effects, we conducted another simulation using GTS mode instead of CSMA/CA. Table 4 shows retransmission rates of end device with GTS transmission. The results confirmed that SDS and Hybrid with GTS transmission mode do not cause any single retransmission whereas non-scheduling still produces a high volume of retransmissions.

Overall, we observed that even if the packet drop rate of non-scheduling case is not significant, the retransmission amount is a lot larger than other two scheduling algorithms. This is because nodes in non-scheduling algorithm are very likely to send packets at the same time, which obviously causes transmission conflicts. Such a high volume of retransmission can cause extra power consumption of ZigBee sensors. In a mobile and wireless environment, this feature might be very critical and problematic.
Table 4 Details of retransmission rate at each end device in Test2 with GTS transmission mode

| End node | NONE | SDS | Hybrid |
|----------|------|-----|--------|
| 8        | 33%  | 0%  | 0%     |
| 9        | 34%  | 0%  | 0%     |
| 10       | 0%   | 0%  | 0%     |
| 11       | 0%   | 0%  | 0%     |
| 12       | 0%   | 0%  | 0%     |
| 13       | 0%   | 0%  | 0%     |
| 14       | 0%   | 0%  | 0%     |
| 15       | 34%  | 0%  | 0%     |
| 16       | 0%   | 0%  | 0%     |
| 17       | 0%   | 0%  | 0%     |
| 18       | 0%   | 0%  | 0%     |
| 19       | 99%  | 0%  | 0%     |
| 20       | 0%   | 0%  | 0%     |

Fig. 16 End-to-end delay in uncongested traffic scenario

(3) End-to-End Delay

Finally, we examined collision effects on end-to-end delays in uncongested and congested network scenarios. As you see in Fig. 16, in the uncongested scenario, the non-scheduling method shows best end-to-end delay while the original SDS method results in the worst end-to-end delay. It is explained by the fact that non-scheduling allows all clusters to send beacons at the same time and so there is no delay before launching data transmission. On the other hand, the serialized SDS allows only one cluster to be active at a time and so other clusters should wait their turns with delaying their data transmission. This lengthens the end-to-end delay.

In contrast, the proposed scheme improves the end-to-end delay mainly because of the following two facts. First, the grouped clusters transmit beacons at the same time, which advances a schedule of certain clusters. For example, the hybrid scheme allows cluster 2 in Fig. 5 to send a beacon at the 1st time slot whereas it was originally scheduled to send its first beacon at the 12th frame in SDS. More importantly, a cluster belonging to the group can have an opportunity to increase its SD and beacon frequency and hence improve its transmission end-to-end delay. For example in Fig. 5, by putting together cluster 2 and 5, cluster 2 can change its BI and SD from 32 to 16 and from 1 to 2, respectively. This gives a cluster 2 more slots to transmit data and hence results in end-to-end delay gain.

In addition, we repeated simulations in a congested scenario and observed that both Hybrid and SDS schemes outperformed non-scheduling method. In the congested network, due to beacon conflicts of non-scheduling scheme, even successfully delivered packets are very likely to experience a large amount of retransmission, which lengthened end-to-end delays. Note that early packet drops might help end-to-end delay of other successful packets in the congested scenario. However, we observed that the proposed scheme provided shorter end-to-end delays as well as lower packet drop rate than non-scheduling algorithm. Compared to SDS, the end-to-end delay gain of the proposed scheme is not clear because the end-to-end delay in this paper was measured only for the successfully delivered packets and so such gain might be thanks to the proposed algorithm’s benefits but also might come from early packet drops. But note that in the previous uncongested scenario, the end-to-end delay gain of the proposed scheme compared to SDS was more clearly shown. Overall, we observed that the proposed hybrid scheme shows 1) better end-to-end delay than non-scheduling in the congested scenario and 2) shorter delay than SDS and very similar delay to non-scheduling case even in uncongested scenario while providing lower packet drop rate than non-scheduling.

In summary, we conclude that the proposed hybrid scheme beats non-scheduling and SDS from all aspects of interests—total duty cycles, a packet drop rate, and an end-to-end delay.

6. Conclusion

In this paper, we presented a hybrid beacon scheduling algorithm which complements drawbacks of non-scheduling and the previous SDS scheduling method. Our simulation results showed that allowing partial overlaps between two clusters increases “schedulability” by lowering total duty cycles of resources while the proposed scheme can keep collision-free feature as well. Also, the proposed utilization-aware grouping method makes it possible to select grouping clusters with minimizing total utilization by considering resource usage pattern and its beacon parameters.
References

[1] “IEEE 802.15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (WPANs),” IEEE, Sept. 2006.

[2] J. Zheng and M.J. Lee, “Will IEEE 802.15.4 Make Ubiquitous Networking Reality?: A Discussion on a Potential Low Power, Low Bit Rate Standard,” IEEE Communications Magazine, vol.42, no.6, pp.140–146, 2004.

[3] S.A. Khan and F.A. Khan, “Performance analysis of a Zigbee beacon enabled cluster tree network,” ICEE ’09 April 2009.

[4] A. Koubaa, M. Alves, L. Attia, and A. Vannieuwenhuyse, “Collision-Free Beacon Scheduling Mechanisms for IEEE 802.15.4/Zigbee Cluster-Tree Wireless Sensor Networks,” aswn2007, Nov 2007.

[5] A. Koubaa, A. Cunha, and M. Alves, “A Time Division Beacon Scheduling Mechanism for IEEE 802.15.4/Zigbee Cluster-Tree Wireless Sensor Networks,” Real-Time Systems, 2007.ECRTS ’07/19th Euromicro Conference, pp.125–135, 2007.

[6] E. Toscano and L.L. Bello, “Multichannel Superframe Scheduling for IEEE 802.15.4 Industrial Wireless Sensor Networks,” IEEE Trans. Ind. Informat., vol.8, no.2, pp.337–350, May 2012.

[7] W.-Y. Lee, K.-I. Hwang, Y.-A. Jeon, and S. Choi, “Distributed Fast Beacon Scheduling for Mesh Networks,” Mobile Adhoc and Sensor Systems (MASS), 2011 IEEE 8th International conference, pp.727–732, 2011.

[8] S. Santhi and K. Venkatachalapathy, “Multiple Cluster Tree Routing and Scheduling for Collision Avoidance in 802.15.4. Sensor Networks,” Research Journal of Applied Sciences, Engineering and Technology 7(15): 3075-3082, 2014.

[9] M.-S. Pan and P.-L. Liu, “Low latency scheduling for convergecast in Zigbee tree-based wireless sensor networks,” Journal of Network and Computer Applications, vol.46, pp.252–263, July 2014.

[10] L.-H. Yen, Y.W. Law, and M. Palaniswami, “Risk-aware Distributed Beacon Scheduling for Tree-Based Zigbee Wireless Networks,” IEEE Trans. Mobile Comput., vol.11, no.4, pp.692–703, April 2012.

[11] J. Han, S. Choi, and T. Park, “Maximizing lifetime of cluster-tree ZigBee networks under end-to-end deadline constraints,” IEEE Commun. Lett., vol.14, no.3, pp.214–216, March 2010.

[12] H. Liu, A. Chandra, and J. Srivastava, “eSENSE: energy efficient stochastic sensing framework scheme for wireless sensor platforms,” Proceedings of the fifth international conference on Information processing in sensor networks, April 19-21, 2006, Nashville, Tennessee, USA.

[13] A. Rowe, R. Mangharam, and R. Rajkumar, “RT-Link: A Global Time-Synchronized Link Protocol for Sensor Networks,” Elsevier Ad hoc Networks, Special Issue on Energy efficient design in wireless ad hoc and sensor networks, vol.6, no.8, pp.1201–1220, 2007.

[14] M.-K. Yoon, C.-G. Lee, and J. Han, “Migrating from Per-Job Analysis to Per-Resource Analysis for Tighter Bounds of End-to-End Response Times,” IEEE Trans. Comput., vol.59, no.7, pp.933–942, July 2010.

[15] J. Han, “Global Optimization of ZigBee Parameters for End-to-End Deadline Guarantee of Real-Time Data,” IEEE Sensors J., vol.9, no.5, pp.512–514, 2009.

[16] A. Koubaa, M. Alves, and E. Tovar, “Modeling and Worst-Case Dimensioning of Cluster-Tree Wireless Sensor Networks,” Real-Time Systems Symposium, pp.412–421, 2006.

[17] M.R. Garey and D.S. Johnson, “The Complexity of Near-Optimal Graph Coloring,” JACM, vol.23, no.1, pp.43–49, Jan. 1976.

[18] A. Dharwadker, The Vertex Coloring Algorithm, 2006.

[19] Y. Ding and H. Ho, “CFP scheduling for real-time service and energy efficiency in the industrial applications of IEEE 802.15.4,” Journal of Communications and Networks, vol.15, no.1, pp.87–101, Feb. 2013.