IPP HURRAY!

www.hurray.isep.ipp.pt

Technical Report

Modelling and Worst-Case Dimensioning of Cluster-Tree Wireless Sensor Networks

Anis Koubaa

Mário Alves

Eduardo Tovar

TR-061004

Version: 1.0

Date: October 2006

Modelling and Worst-Case Dimensioning of Cluster-Tree Wireless Sensor Networks

Anis KOUBAA, Mário ALVES, Eduardo TOVAR

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: {akoubaa@dei, mjf, emt@dei}.isep.ipp.pt

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

Abstract

Time-sensitive Wireless Sensor Network (WSN) applications require finite delay bounds in critical situations. This paper provides a methodology for the modeling and the worst-case dimensioning of cluster-tree WSNs. We provide a fine model of the worst-case cluster-tree topology characterized by its depth, the maximum number of child routers and the maximum number of child nodes for each parent router. Using Network Calculus, we derive "plug-and-play" expressions for the end-to-end delay bounds, buffering and bandwidth requirements as a function of the WSN cluster-tree characteristics and traffic specifications. The cluster-tree topology has been adopted by many cluster-based solutions for WSNs. We demonstrate how to apply our general results for dimensioning IEEE 802.15.4/Zigbee cluster-tree WSNs. We believe that this paper shows the fundamental performance limits of cluster-tree wireless sensor networks by the provision of a simple and effective methodology for the design of such WSNs.

Modeling and Worst-Case Dimensioning of Cluster-Tree Wireless Sensor Networks

Anis KOUBAA, Mário ALVES, Eduardo TOVAR

IPP-HURRAY! Research Group, Polytechnic Institute of Porto
Rua Dr. Antonio Bernardino de Almeida, 431, 4200-072 Porto, PORTUGAL
{akoubaa, emt}@dei.isep.ipp.pt , mjf@isep.ipp.pt

Abstract

Time-sensitive Wireless Sensor Network (WSN) applications require finite delay bounds in critical situations. This paper provides a methodology for the modeling and the worst-case dimensioning of cluster-tree WSNs. We provide a fine model of the worst-case cluster-tree topology characterized by its depth, the maximum number of child routers and the maximum number of child nodes for each parent router. Using Network Calculus, we derive "plug-and-play" expressions for the end-to-end delay bounds, buffering and bandwidth requirements as a function of the WSN cluster-tree characteristics and traffic specifications. The cluster-tree topology has been adopted by many cluster-based solutions for WSNs. We demonstrate how to apply our general results for dimensioning IEEE 802.15.4/Zigbee cluster-tree WSNs. We believe that this paper shows the fundamental performance limits of cluster-tree wireless sensor networks by the provision of a simple and effective methodology for the design of such WSNs.

1. Introduction

In time-sensitive Wireless Sensor Network (WSN) applications, it is important that time-critical messages arrive to their destination prior to the expiration of their deadlines [1]. This requires *a priori* dimensioning of the available resources of the WSN to provide an end-to-end guaranteed service from the source node to the sink (e.g. control station).

Typically, wireless sensor networks can be organized in unstructured peer-to-peer or structured cluster-based topologies. In spite of a greater flexibility, the peer-to-peer model is, in general, not suitable to provide predictable service guarantees, mainly due to its unstructured nature, and also to the typical use of contention-based Medium Access Control (MAC) mechanisms. On the other hand, structured cluster-based topologies are quite suitable for WSNs with demanding requirements in terms of Quality of Service (QoS) support and real-time communications. In the literature, cluster-based topologies have been deployed to improve service guarantees in WSNs, by either using deterministic MAC protocols based on Time Division Multiple Access (TDMA) [2, 3] or two-tiered architectures [4, 5]. The cluster-tree topology is a particular case of cluster-based topologies, which uses multi-hop tree routing to transport data from the source to the destination. The tree defines a backbone that consists of a set of routers (also called cluster-heads) that collect data from sensor nodes belonging to their cluster, and forward it to the next level routers in the tree until reaching the sink.

A common feature of cluster-tree WSNs is that each node (or a subset of nodes) can be granted a minimum service guarantee all along the path through which the data is relayed, by the allocation of some resources (e.g. time slots in TDMA or bandwidth sharing) in each intermediate router. The communication path between two nodes in the cluster-tree network will then have an end-to-end predictable service guarantee, thus enabling the evaluation of worst-case performance metrics, namely the delay bounds and resource requirements. In what follows, we refer to *resource requirements* to denote bandwidth and buffering requirements in each router.

In this paper, we show that a cluster-tree topology can be modeled by three parameters: its *depth*, the *maximum number of child nodes* and the *maximum number of child routers* per parent router. In a cluster-tree topology, a *node* is a simple device that collects sensory data and forwards it to the parent router to which it is associated. A *router* is a device that has more advanced networking capabilities, in addition to the node functionalities.

Given such a network model, it is then possible to predict the end-to-end performance of the WSN in terms of delay bounds and resource requirements, at design time. The purpose of this paper is to provide a methodology that permits this worst-case dimensioning of cluster-tree wireless sensor networks. The problem that we tackle in this paper can be roughly formulated as follows.

> *Having a WSN organized in a cluster-tree topology, with a given number of nodes, a given number of routers, and a given depth, and provided that a minimum service is guaranteed to every node and router, what are the delay bounds for flows originated from nodes at a given depth in the WSN, and what are the minimum resource requirements in each router?*

A practical motivation that drives this work is that the cluster-tree topology is supported by the IEEE 802.15.4/Zigbee protocol standards [6, 7], recently defined for Low-Rate Wireless Personal Area Networks (WPANs), with a great potential for deployment in WSN applications [8]. Hence, and just as an example of instantiation, we apply the general solution of the aforementioned problem to the specific case of cluster-tree WSNs based on the IEEE 802.15.4/Zigbee protocols. Notably, our approach can easily be applied to any other cluster-tree WSN offering service guarantees, such as LEACH [2].

2. Related Work and Contributions

The prediction of the worst-case performance of WSNs has recently attracted several recent research works. In [9], the authors have defined the concept of *real-time capacity* of wireless networks as the ability of the network to deliver data by their deadlines. They also derived a sufficient schedulablity condition for a class of non-preemptive fixed priority scheduling algorithms. The analysis presented in this paper is topology-independent. Even though this work is a relevant contribution to the understanding of the real-time capacity of multi-hop WSNs, the applicability of the results to a real WSN remains constrained by the restrictive assumption of an ideal MAC, implementing a medium arbitration with zero overhead.

Another line of research works dealing with the prediction of the worst-case performance of WSNs has considered the extension of the Network Calculus methodology [10] to WSNs [11-13]. Network Calculus is a theory for designing and analyzing deterministic queuing systems, which provides a mathematical framework based on *min-plus* and *max-plus* algebras for delay bound analysis in packet-switched networks. In [11], the authors have defined a general analytical framework, which extends Network Calculus to be used in dimensioning WSNs, taking into account the relation between node power consumption, node buffer requirements and the transfer delay. The main contribution in [11] is the provision of general expressions modeling the arrival curves of the input and output flows at a given *parent* sensor node in the network, as a function of the arrival curves of its *children*. These expressions are obtained by direct application of Network Calculus theorems. Then, the authors have defined an iterative procedure to compute the internal flow inputs and outputs in the WSN, node by node, starting from the lowest leaf nodes until arriving to the sink. Using Network Calculus theorems, the authors have extended the general expressions of delay bounds experienced by the aggregated flows at each hop and have deduced the end-to-end delay bound as the sum of all per-hop delays on the path.

In [12], the same authors use their methodology for the worst-case dimensioning of WSNs under uncertain topologies. The key difference, as compared to [11], is the computation of the worst-case topology, i.e. the topology that experiments the worst-case behavior in terms of delay bounds and buffering requirements. The same models (expressions between input and output flows, and the iterative procedure) in [11] have been used in the analysis presented in [12]. In [13], the analysis has been extended to support multiple sinks. The main results of the Sensor Network Calculus methodology that we use in this paper will be presented in Section 2.

In [14], the authors have analyzed the performance of general-purpose sink-tree networks using network calculus and derived tighter end-to-end delay bounds.

In this paper, we apply and extend the Sensor Network Calculus methodology to the worst-case dimensioning of cluster-tree topologies, which are particularly appealing for WSNs with stringent timing requirements. Our work differs from the previous works and contributes to the state-of-the art in three aspects. **First**, we provide a fine general model for cluster-tree WSNs defined by a depth, the maximum number of child nodes and the maximum number child routers per each parent router, and consider input flows at each nodes bounded by a (*b*,*r*) arrival curve, where *b* is the maximum burst size of the flow, and *r* is its average rate. Our work differs from [14] in the system model used in the analysis. In [14], the authors have considered a general-purpose tandem of nodes, different from the cluster-tree model defined in this paper. Our model is more accurate in the context WSNs. **Second**, we address the particular problem of the worst-case dimensioning of *cluster-tree topologies*, which we believe are of a great interest for time-sensitive WSN applications. We apply the Sensor Network Calculus theory to our model and derive simple recurrent equations that express the resource requirements at each node in the network, and the per-hop as well as end-to-end delay bounds as function of the cluster-tree parameters. A first advantage of our work as compared to [11-13] is the provision of practical recurrent equations, thus avoiding iterative computations (node by node). The resulting time-complexity of such an approach is not suitable for large-scale WSNs. In addition, Our model is more accurate for this specific WSN topology than the general WSN structures considered in [11-13], and the results presented in this paper accurately show its worst-case performance. We also propose to evaluate the end-to-end delay bound of a given individual flow differently from the approaches in [11-13]. Instead of computing the sum of per-hop delays for aggregate flows, we propose to compute the end-to-end service curve of every individual flow along its path from its source to its destination, using the concatenation theorem of Network Calculus [10]. This methodology was used in [14] and shows that it provides tighter end-to-end delay bounds. The numerical results that we present in this paper confirm the above conclusion. **Third**, we show how to apply these results in the dimensioning of the worst-case performance of IEEE 802.15.4/Zigbee WSNs, which helps to have a better understanding of the limits of this standardized technology.

On the other hand, the deterministic performance of the IEEE 802.15.4 protocol has been addressed in some recent research works [15-17]. These works have basically addressed the evaluation and the improvement of the Guaranteed Time Slot (GTS) mechanism in IEEE 802.15.4 single-cluster star-based networks. In [15], the authors have presented an analytical tool using Network Calculus for modeling and evaluating the delay bound guaranteed by the GTS mechanism in a star-based WSN. In [16, 17], some schemes for improving the GTS mechanism have been proposed and analyzed. The applicability of these results only holds for single-cluster star-based WSNs. This paper contributes to the analysis of the GTS mechanism by extending it to a multi-hop cluster-tree topology.

To our best knowledge, the analysis of deterministic guarantees in cluster-tree WSNs and its application to IEEE 802.15.4/Zigbee networks has not been addressed yet.

3. Background

3.1 Network Calculus Fundamentals

Network Calculus is a mathematical tool based on *min-plus* and *max-plus* algebras for designing and analyzing deterministic queuing systems [10]. A basic system representation is illustrated in Fig. 1.

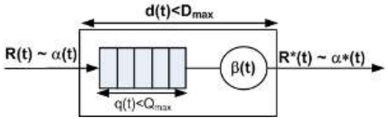


Fig. 1. System representation in Network Calculus theory

For a given data flow, the **input function** is the cumulative arrival function denoted by $R(t)$, which represents the number of bits that arrive during the interval $[0, t]$. We denote by $R^*(t)$ the **output function** of the flow, which represents the number of bits that leave the system during the interval $[0, t]$.

Furthermore, Network Calculus theory assumes that:

- It exists an **arrival curve** $\alpha(t)$ that upper bounds $R(t)$ such that $\forall s,\ 0 \le s \le t,\ R(t) - R(s) \le \alpha(t-s)$. This inequality means that the amount of traffic that arrives to receive service in any interval $[s, t]$ never exceeds $\alpha(t-s)$. It is also said that $R(t)$ is constrained by $\alpha(t)$, or $R(t) \sim \alpha(t)$.
- It exists a minimum **service curve** $\beta(t)$ guaranteed to $R(t)$. This means that the output flow during any given *busy period* $[t, t+\Delta]$ of the flow is at least equal to $\beta(\Delta)$, i.e. $R^*(t+\Delta) - R^*(t) \ge \beta(\Delta)$, where $\Delta > 0$ is the duration of any busy period.

The knowledge of the arrival and service curves enables the computation of the delay bound D_{max}, which represents the worst-case response time of a message, and the backlog bound Q_{max}, which is the maximum queue length of the flow.

The **delay bound**, D_{max}, for a data flow with an arrival curve $\alpha(t)$ that receives the service curve $\beta(t)$ is the maximum horizontal distance between $\alpha(t)$ and $\beta(t)$ (see Fig. 2), and is expressed as follows:

$$D_{\max} = \sup_{s \ge 0} \left\{ \inf\left(\tau \ge 0 \,\middle|\, \alpha(s) \le \beta(s+\tau)\right) \right\} \ge d(t) \quad ,\forall t \qquad \textbf{(1)}$$

The **backlog bound**, Q_{max}, for a data flow with an arrival curve $\alpha(t)$ that receives the service $\beta(t)$ is the maximum vertical distance between $\alpha(t)$ and $\beta(t)$, and is expressed as:

$$Q_{\max} = \sup_{s \ge 0} \left(\alpha(s) - \beta(s) \right) \ge q(t) \quad ,\forall t \qquad \textbf{(2)}$$

Fig. 2 presents an example of the delay and backlog bound computation for a linear arrival curve $\alpha(t) = b + r \cdot t$ that receives a *rate-latency* service curve $\beta_{R,T}(t) = R \cdot (t - T)^+$, where $R \ge r$ is the guaranteed bandwidth, T is the maximum latency of the service and $(x)^+ = \max(0, x)$.

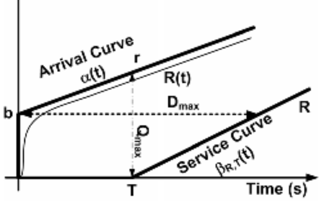


Fig. 2. Delay and backlog bounds

This service curve is typically used for servers that provide a bandwidth guarantee with a certain latency. The latency T refers to the deviation of the service (e.g. blocking factor of non-preemptive transmissions).

The delay bound D_{max} (presented in Fig. 2) guaranteed for the data flow with the arrival curve $\alpha(t) = b + r \cdot t$ (also called **(*b*, *r*)-curve**) by the service curve $\beta_{R,T}(t) = R \cdot (t - T)^+$ is computed as follows [10]:

$$D_{\max} = \frac{b}{R} + T \qquad \textbf{(3)}$$

and the backlog bound is expressed as [10]:

$$Q_{\max} = b + r \cdot T \qquad \textbf{(4)}$$

In our analysis, we will use the previous linear arrival curve and the rate-latency service curve since they accurately represent the system as it will be explained in Section 4.

In Network Calculus, it is also possible to express an upper bound for the output flow and the equivalent service curve for the concatenation of two service curves.

The output function $R^*(t)$, of a flow $R(t)$ constrained by an arrival curve $\alpha(t)$ that traverses a system offering a service curve $\beta(t)$, is constrained by **output bound** $\alpha^*(t)$:

$$\alpha^*(t) = (\alpha \oslash \beta)(t) \qquad \textbf{(5)}$$

where $\oslash$ is the **min-plus deconvolution** defined for $f, g \in \mathbb{F}$, where $\mathbb{F}$ is the set of wide-sense increasing functions, as:

$$(f \oslash g)(t) = \sup_{s \ge 0} \left(f(t+s) - g(s) \right)$$

We consider the following corollary as an application of Eq. (5) to the case of a linear arrival curve and a rate-latency service curve. The proof can be found in [18].

Corollary 1. Assume that a flow is constrained by an arrival curve $\alpha(t) = b + r \cdot t$ and a FIFO node provides a guaranteed service curve $\beta_{R,T}(t) = R \cdot (t - T)^+$ to the flow. Then, the output bound of the flow is expressed as:

$$\alpha^*(t) = \alpha(t) + r \cdot T \qquad \textbf{(6)}$$

And for any constant $K \in \mathbb{R}$, we easily show that:

$$\left(K \cdot \alpha(t)\right) \oslash \beta_{R,T}(t) = K.\left(\alpha \oslash \beta_{R,T}\right)(t) \qquad \textbf{(7)}$$

Concatenation of Nodes. Assume that a flow *R(t)* traverses systems *S1* and *S2* in sequence, where *S1* offers service curve $\beta_1(t)$ and *S2* offers $\beta_2(t)$. Then, the resulting system *S*, defined by the concatenation of the two systems *S1* and *S2*, offers the following service curve to the flow:

$$\beta(t) = (\beta_1 \otimes \beta_2)(t) \qquad \textbf{(8)}$$

where $\otimes$ is the **min-plus convolution** defined for $f, g \in \mathbb{F}$ as:

$$(f \otimes g)(t) = \inf_{0 \le s \le t} \left(f(t-s) + g(s) \right)$$

3.2 Network Flow Analysis

Some results of the Sensor Network Calculus methodology that are relevant for our analysis are presented next.

The sensor network model (refer to Fig. 3) considers that, for a given path, each node has one parent and one or more children (with the exception of end nodes). It is assumed that each node i has an input flow with an arrival curve $\alpha_i(t)$. Hence, the total input of a given parent node i is the sum of its input and the outputs of its children as obtained by Eq. (5).

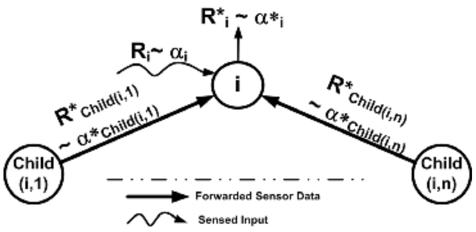


Fig. 3. The sensor network model

As a result, the total input flow of a given parent node *i* is:

$$\overline{\alpha}_i(t) = \alpha_i(t) + \sum_{j=1}^{n} \alpha^*_{Child(i,j)}(t) \tag{9}$$

Applying Eq. (5) again to the parent node *i,* assuming that it has been guaranteed a service curve $\beta_i(t)$, its output flow is expressed as follows:

$$\alpha^*_i(t) = \left(\overline{\alpha}_i \odot \beta_i\right)(t) = \left(\left[\alpha_i + \sum_{i=1}^{n} \alpha^*_{Child(i,j)}\right] \odot \beta_i\right)(t) \tag{10}$$

Hence, the network flow analysis in the Sensor Network Calculus methodology consists in computing iteratively the output flow bound $\alpha^*_i(t)$ using the above equations, from the bottom of the network until arriving to the destination (sink). Then, the per-hop delay bound is computed node by node using Eq. (1), and the end-to-end delay bound in a given path is then equal to the sum of all per-hop delay bounds.

3.3 Aggregate Scheduling

Consider a FIFO queue that multiplexes many flows and offers them a given guaranteed service curve $\beta(t)$. Hence, applying Eqs. (1) and (2), it is possible to compute the delay and backlog bounds for the *entire aggregate* flow (the sum of all flows) that enters the FIFO queue, provided that this aggregate is bounded by an arrival curve. Note that while these delay and backlog bounds are global for all flows, it is also possible to compute the delay bounds for individual flows. We provide the following corollary for aggregate scheduling in Network Calculus, which will be used in our approach. This corollary is a direct result from Proposition 6.2.1 in [10], and the proof can be found in [19].

Corollary 2. Aggregate Scheduling. Consider a FIFO node that multiplexes two flows 1 and 2. Assume that flow 2 is constrained by an arrival curve $\alpha_2(t) = b_2 + r_2 \cdot t$ and the FIFO node provides a guaranteed service curve $\beta_{R,T}(t) = R \cdot (t - T)^+$ to the aggregate of flows. Then, for any $\theta \geq 0$, flow 1 is guaranteed the service curve:

$$\beta^1_\theta(t) = (R - r_2) \cdot \left[t - \left(\frac{b_2 + r_2 \cdot (T - \theta)}{R - r_2} + T\right)\right]^+ 1_{\{t > \theta\}} \tag{11}$$

4. System Model

In this section, we present the cluster-tree network model and the corresponding traffic model that we consider in the rest of this paper. We also discuss its validity for real-world WSNs.

4.1 The Cluster-Tree Network Model

Like in any tree network, the cluster-tree topology contains a special node called *root*, which identifies the entire network. In addition, in a tree network, some special devices may have the ability to allow the association from other nodes. These nodes are called *routers*. Other end devices with no ability to associate other devices are called *child nodes*. Both child nodes and routers are assumed to have sensing capabilities and are referred to as *sensor nodes*.

Fig. 4 presents an example of the cluster-tree network with the three types of nodes. A cluster-tree network is then a tree network where each router forms it own logical cluster.

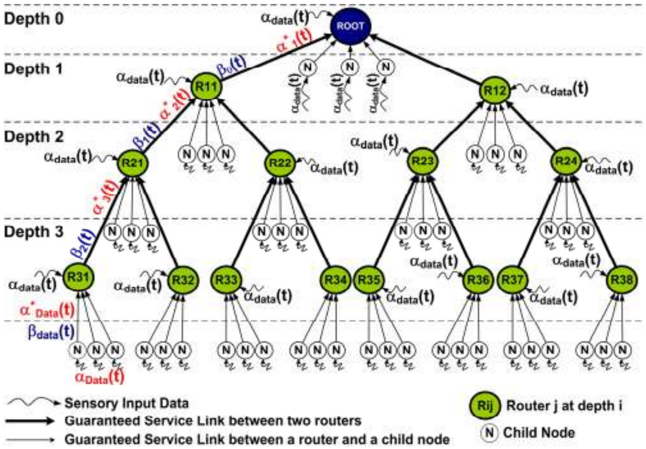


Fig. 4. The cluster-tree network model

Basically, we aim to specify the worst-case cluster-tree topology, i.e. the network configuration that leads to the worst-case delay bounds and resource requirements. This means that a dynamically changing cluster-tree WSN can assume different cluster-tree configurations, but, it can never exceed the worst-case topology, in terms of maximum depth and number of child routers/nodes.

For that purpose, we specify the worst-case cluster-tree topology model by the following three parameters:

- *maxDepth*: represents the maximum depth of the network, which specifies the maximum number of logical hops for a message from a **router** to reach the **root** (including the root as final hop). This means that the network cannot expand more if the maximum logical distance from a router to the root is equal to *maxDepth*. The root is considered to be in a depth equal to zero. Hence, the maximum depth of a child node is then *maxDepth+1* (see Fig. 4).
- N_{child}: the maximum number of child nodes that can be associated to a parent router and have been allocated resource guarantees (e.g. time slots or bandwidth).
- N_{router}: the maximum number of child routers that can be associated to a parent router and have been allocated resource guarantees.

The example illustrated in Fig. 4 corresponds to a setting where *maxDepth* = 3, N_{router} = 2 and N_{child} = 3.

Note that a cluster-tree WSN may contain additional routers/nodes per parent router than those defined by N_{router} and N_{child}. However, these additional devices are not granted guaranteed resources. An illustrative example showing the constraints on these parameters will be presented in the application to IEEE 802.15.4/Zigbee protocols, in Section 6.

By convention, we say that a router at depth i is upstream to a router at a depth j, if and only if $i < j$.

4.2 The Traffic Model

Data flows can be upstream (from a sensor node to the sink) or downstream (from the sink to a given node). Typically, in WSNs, critical messages are forwarded from individual sensor nodes to the sink (e.g. control station), in the upstream direction. The downstream direction is more dedicated to queries sent by the control station. Without loss of generality, we assume that the control station is attached to the root, and thus we focus on critical flows in the upstream direction, from sensor nodes to the root. In this paper, the case of downstream flows is not considered due to space limitations.

In critical situations, every child node/router in a WSN can be required to send a data flow to report its sensory data. In the worst-case scenario, all child nodes/routers that have allocated resources will have data to send to the sink. We assume that the maximum individual data flow that can be sent by each child node/router is bounded by the arrival curve $\alpha_{data}(t) = b_{data} + r_{data} \cdot t$, where b_{data} is the maximum burst size of the data flow, and r_{data} is its average rate. Observe in Fig. 4 that each child node and router has its sensory data input bounded by $\alpha_{data}(t)$. This is an advantage of using Network Calculus representation, since instead of considering the real flow, which may be variable (e.g. periodic traffic, aperiodic traffic, stochastic traffic), we merely consider an upper bound of the cumulative arrivals of the flow, independently from its nature. This traffic model also incorporates the classical representation of the periodic arrival model with or without jitter [20]. In case of heterogeneous traffic sources (different types of sensors), $\alpha_{data}(t)$ will represent the curve of the highest cumulative arrival function. This may introduce more pessimism to the analysis if the variance between different node's traffic is very significant. However, in most WSN applications, the variance between different traffic flows is likely to be small, since special events are commonly reported by similar sensory data (e.g. temperature measurements, electromagnetic signals).

As for the service model granted for each flow, recall that we consider child nodes and routers that have been allocated guaranteed resources. Thus, since the arrival curve in every child node is constrained by $\alpha_{data}(t)$, it is assumed that each child node has a service guarantee from its parent router corresponding to the service curve $\beta_{data}(t) = R_{data} \cdot (t - T_{data})^{+}$, where $R_{data} \geq r_{data}$ is the guaranteed bandwidth and T_{data} is the maximum latency of the service, which refers to the deviation of the service (e.g. blocking factor or non preemptive transmissions). The latency depends on the resource allocation mechanism. This service curve model fits any kind of bandwidth guarantees, such as fair queuing, TDMA slot allocation or IEEE 802.15.4 GTS mechanism [15].

On the other hand, child routers are also allocated guaranteed resources by their parent routers. Contrarily to the previous case, the amount of bandwidth required for each child router depends on the amount of traffic at its input. For instance, a router that is located at a higher depth in the tree (closer to the root) must provide more bandwidth and buffering resources than a router located at a lower depth (farther from the root), due to the accumulation of upstream data flows in the direction of the root. In addition, due to the symmetry of our model, the bandwidth and buffering requirements only depend on the depth of the router, i.e. all routers at the same depth must provide the same resource guarantees. As a result, we assume that any router j at a depth i provides a service guarantee to each of its child routers corresponding to the service curve $\beta_i(t) = R_i \cdot (t - T_i)^{+}$, where R_i is the guaranteed bandwidth, which must be higher than the overall rate of all the input flows, and T_i is the maximum latency of the service.

Given such a cluster-tree topology model, we address the worst-case dimensioning and performance analysis of the WSN. In particular, we aim to characterize:

- The **minimum resource requirements** in each router, in terms of (1) bandwidth requirement R_i and (2) buffering requirement, i.e. the maximum buffer size needed to store the bulk of data at the router's input.
- The **maximum delay bound** of the WSN, which represents the delay experienced by a data flow of a node in the lowest depth (*maxDepth*+1) to reach the root.

5. Cluster-Tree Network Analysis

In this section, we analyze the cluster-tree topology model for WSN presented in Section 4. To address the worst-case dimensioning problem, the first step is to derive recurrent equations of the input and output flows inside the WSN. Then, we characterize the resource requirements and the corresponding service curves at each router. Finally, with the knowledge of the input arrival curves and the service curves, we derive the delay bounds for individual data flows.

To give a practical intuition on the general solution, let us consider the example in Fig. 4 corresponding to a cluster-tree WSN with *maxDepth* = 3, N_{router} = 2 and N_{child} = 3. We propose to evaluate the input/output arrival curves and service curves, depth by depth, using the Sensor Network methodology starting from the lowest leafs. Then, we deduce the general recurrent expressions.

5.1 Computation of Input and Output Flows

Consider the following queuing system in Fig. 5, which is equivalent to the one in Fig. 4.

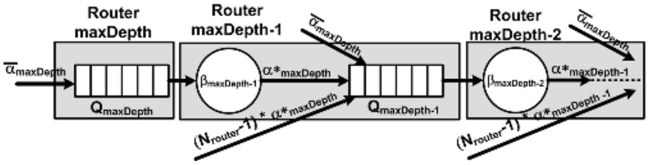

Router maxDepth
Router maxDepth-1
Router maxDepth-2
$\overline{\alpha}_{maxDepth}$
$Q_{maxDepth}$
$\beta_{maxDepth-1}$
$\alpha^*_{maxDepth}$
$\overline{\alpha}_{maxDepth}$
$Q_{maxDepth-1}$
$(N_{router}-1) * \alpha^*_{maxDepth}$
$\beta_{maxDepth-2}$
$\alpha^*_{maxDepth-1}$
$\overline{\alpha}_{maxDepth}$
$(N_{router}-1) * \alpha^*_{maxDepth-1}$

Fig. 5. Queuing system model

Analysis of depth *maxDepth+1* (*depth* = 4)

At depth *maxDepth*+1 (see Fig. 4), there is no router, and there are nodes with input data flows, each flow constrained by the arrival curve $\alpha_{data}(t)$. Since each node is granted a service curve $\beta_{data}(t)$, then using Eqs. (5) and (6), the output flow of each child node can be expressed as follows:

$$\alpha^*_{data}(t) = (\alpha_{data} \oslash \beta_{data})(t) = \alpha_{data}(t) + r_{data} \cdot T_{data} \qquad \textbf{(12)}$$

Analysis of depth m*axDepth* (*depth* = 3)

At depth *maxDepth* (see Fig. 4 and Fig. 5), the total input of each router, denoted by $\overline{\alpha}_{maxDepth}(t)$, comprises its sensory data flow constrained by $\alpha_{data}(t)$, and the sum of the output flows of its child nodes.

$$\overline{\alpha}_{maxDepth}(t) = \alpha_{data}(t) + N_{Child}\cdot \alpha^{*}_{data}(t)$$

Thus, according to Eq. (12), we have:

$$\overline{\alpha}_{maxDepth}(t) = (N_{Child}+1)\cdot \alpha_{data}(t) + N_{Child}\cdot r_{data}\cdot T_{data} \tag{13}$$

Note that $\overline{r}_{maxDepth} = (N_{child}+1)\cdot r_{data}$ is the resulting rate of the aggregate of $(N_{child}+1)$ input data flows, and $\overline{b}_{maxDepth} = (N_{child}+1)\cdot b_{data} + N_{Child}\cdot r_{data}\cdot T_{data}$ is its resulting burst.

The input flow $\overline{\alpha}_{maxDepth}(t)$ is forwarded by the router at depth *maxDepth* to its parent router at depth *maxDepth*-1. This child router is allocated a service curve $\beta_{maxDepth-1}(t) = R_{maxDepth-1}\cdot(t - T_{maxDepth-1})^{+}$ by its parent. Hence, according to Eq. (5), the output flow from a child router at depth *maxDepth* is then expressed as:

$$\alpha^{*}_{maxDepth}(t) = \left(\overline{\alpha}_{maxDepth}(t) \odot \beta_{maxDepth-1}(t)\right)$$

As a result, applying Eq. (6) we get:

$$\alpha^{*}_{maxDepth}(t) = \left(\overline{\alpha}_{maxDepth}(t) + \sigma_{maxDepth-1}\right) \quad \text{where } \sigma_{maxDepth-1} = \overline{r}_{maxDepth}\cdot T_{maxDepth-1} \tag{14}$$

Analysis of depth *maxDepth*-1 (*depth* = 2)

At depth *maxDepth*-1, the total input of each router, denoted by $\overline{\alpha}_{maxDepth-1}(t)$, comprises its sensory data flow constrained by $\alpha_{data}(t)$, and the sum of the output flows of its child routers $\alpha^{*}_{maxDepth}(t)$ and the output of its child nodes $\alpha^{*}_{data}(t)$. It results that:

$$\overline{\alpha}_{maxDepth-1}(t) = \left(\left(\alpha_{data}(t) + N_{child}\cdot \alpha^{*}_{data}(t)\right) + N_{router}\cdot \alpha^{*}_{maxDepth}(t)\right)$$

Thus, according to Eqs. (13) and (14) we have:

$$\overline{\alpha}_{maxDepth-1}(t) = \left((N_{router}+1)\cdot \overline{\alpha}_{maxDepth}(t) + N_{router}\cdot \sigma_{maxDepth-1}\right) \tag{15}$$

The input flow $\overline{\alpha}_{maxDepth-1}(t)$ is forwarded by the router at depth *maxDepth*-1 to its parent router at depth *maxDepth*-2. This child router is allocated a service curve $\beta_{maxDepth-2}(t) = R_{maxDepth-2}\cdot(t - T_{maxDepth-2})^{+}$ by its parent. Hence, according to Eq. (5), the output flow from a child router at depth *maxDepth*-1 is then expressed as:

$$\alpha^{*}_{maxDepth-1}(t) = \overline{\alpha}_{maxDepth-1}(t) \odot \beta_{maxDepth-2}(t)$$

As a result, applying Eqs. (6) and (15) we get:

$$\alpha^{*}_{maxDepth-1}(t) = \left(\begin{array}{l}(N_{router}+1)\cdot \overline{\alpha}_{maxDepth}(t) + \\ N_{router}\cdot \sigma_{maxDepth-1} + \sigma_{maxDepth-2}\end{array}\right) \quad \text{where } \sigma_{maxDepth-2} = (N_{router}+1)\cdot \overline{r}_{maxDepth}\cdot T_{maxDepth-2} \tag{16}$$

Analysis of depth *maxDepth*-1 (*depth* = 1)

Similarly to the previous case, the input flow of each router at depth *maxDepth*-2 is expressed as follows:

$$\overline{\alpha}_{maxDepth-2}(t) = \left(\begin{array}{l}\left(N^{2}_{router} + N_{router} + 1\right)\cdot \overline{\alpha}_{maxDepth}(t) \\ N^{2}_{router}\cdot \sigma_{maxDepth-1} + N_{router}\cdot \sigma_{maxDepth-2}\end{array}\right) \tag{17}$$

and the output flow from a child router at depth *maxDepth*-2 for a service curve $\beta_{maxDepth-3}(t)$ is then expressed as:

$$\alpha^{*}_{maxDepth-2}(t) = \left(\overline{\alpha}_{maxDepth-2}(t) + \sigma_{maxDepth-3}\right) \text{ where } \sigma_{maxDepth-3} = \left(N^{2}_{router} + N_{router} + 1\right)\cdot \overline{r}_{maxDepth}\cdot T_{maxDepth-3} \tag{18}$$

General expressions of input/output flows for depth *maxDepth*-i

By recurrence, we can easily prove that the input flow of each router at depth (*maxDepth*-*i*) is expressed as follows:

$$\overline{\alpha}_{maxDepth-i}(t) = \left(\sum_{j=0}^{i} N^{j}_{router}\right)\cdot \overline{\alpha}_{maxDepth}(t) + \sum_{j=0}^{i-1}\left(N^{i-j}_{router}\cdot \sigma_{maxDepth-(j+1)}\right) \quad \text{where } \sigma_{maxDepth-n} = \left(\sum_{k=0}^{n-1} N^{k}_{router}\right)\cdot \overline{r}_{maxDepth}\cdot T_{maxDepth-n} \tag{19}$$

and the output flow from a child router at depth (*maxDepth*-*i*) for a service curve $\beta_{maxDepth-(i+1)}(t)$ is then expressed as:

$$\alpha^{*}_{maxDepth-i}(t) = \overline{\alpha}_{maxDepth-i}(t) + \sigma_{maxDepth\text{-}(i+1)} = \left(\left(\sum_{j=0}^{i} N^{j}_{router}\right)\cdot \overline{\alpha}_{maxDepth}(t) + \sum_{j=0}^{i}\left(N^{i-j}_{router}\cdot \sigma_{maxDepth\text{-}(j+1)}\right)\right) \tag{20}$$

5.2 Per-Router Resource Requirements

Bandwidth requirements

So far, we have computed the internal input and output flows at each router as a function of its depth. Now, we propose to compute the resource requirements at each router that must be provided to its children to ensure bounded end-to-end delays and to avoid buffering overflow.

First, in order to ensure bounded delays, it is mandatory that the amount of bandwidth guaranteed to the input flow at each router is greater than or equal to the input arrival rate.

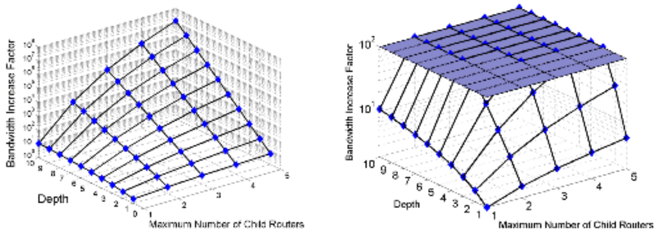
Consider a parent router at depth *maxDepth*-(*i*+1) that offers the service curve $\beta_{maxDepth-(i+1)}(t)$ to one of its child routers with the input flow arrival curve $\alpha_{maxDepth-i}(t)$. It is then necessary to have:

$$R_{maxDepth-(i+1)} \geq \overline{r}_{maxDepth-i} \tag{21}$$

According to Eqs. (19) and (20), we obtain:

$$\begin{aligned}\overline{r}_{maxDepth-i} = r^{*}_{maxDepth-i} &= \left(\sum_{j=0}^{i} N^{j}_{router}\right)\cdot \overline{r}_{maxDepth} \\ &= \gamma_{maxDepth-i}\left(N_{router}\right)\cdot \overline{r}_{maxDepth}\end{aligned} \tag{22}$$

$\gamma_{maxDepth-i}\left(N_{router}\right)$ is called the *bandwidth increase factor* at a given depth (*maxDepth-i*). Note that $\gamma_{maxDepth}\left(N_{router}\right)$ refers to the total number of routers in the network. The parameter $\gamma_{maxDepth-i}\left(N_{router}\right)$ increases with the depth and N_{router}, and this factor represents the ratio of the additional bandwidth that a router, at a depth (*maxDepth-i*), must provide to each of its child routers as compared to the bandwidth guaranteed at the lowest depth *maxDepth*.



a. Bandwidth increase factor as a function of the depth and N_{router}

b. Feasible region for $\gamma_i(N_{router})=10^2$

Fig. 6. Bandwidth increase factor (log-scale)

Fig. 6 presents the variation of the bandwidth increase factor (logarithmic-scale) as a function of the depth of the router and N_{router}.

It can be observed that if N_{router} is high (e.g. equal to 5) the impact of the depth on the bandwidth requirement is very significant. Note that the variation is very limited for the case of $N_{router} = 1$, even for a depth equal to 10. Depending on the maximum bandwidth increase factor allowed when dimensioning the WSN, high values of the N_{router} parameter can be tolerated if the maximum depth of the network is limited. For instance, if the cluster-tree WSN cannot tolerate a bandwidth increase factor more than 10^2 (see Fig. 6.b) all points in the (X,Y,Z) axis located below the plan defined by $Z=10^2, \forall X,Y$ are potential solutions to determine the pair (N_{router}, *maxDepth*). For example, with this bandwidth increase constraint, the *maxDepth* parameter cannot exceed 2 if $N_{router} = 5$, while it can be set to 5 if $N_{router} = 2$.

Buffering requirements

The buffering requirement of a given router at a depth (*maxDepth-i*) stands for the minimum buffer size required to store the incoming bulk of data to avoid buffer overflow. Since $\overline{\alpha}_{maxDepth-i}\left(t\right)$ is the input of a router at a depth (*maxDepth-i*), the minimum buffer size must be greater than the burst size $\overline{b}_{maxDepth-i}$ of the input arrival curve $\overline{\alpha}_{maxDepth-i}\left(t\right)$. If we denote by $Q_{maxDepth-i}$ the minimum buffering requirement of a router at a depth (*maxDepth-i*), then according to Eq. (21), we obtain:

$$Q_{maxDepth-i} = Q^{burst}_{maxDepth-i} + Q^{latency}_{maxDepth-i} = \left(\sum_{j=0}^{i} N^{j}_{router}\right)\cdot \overline{b}_{maxDepth} + \sum_{j=0}^{i-1}\left(N^{i-j}_{router}\cdot \sigma_{maxDepth-(j+1)}\right) \qquad \textbf{(23)}$$

Observe that the buffering requirement is the sum of two terms. The first term is related to the input burst and is a function of the $\gamma_{maxDepth-i}\left(N_{router}\right)$ factor, thus the same behavior as with the bandwidth requirement applies for this term. The second term represents the cumulative effect of the service latency at each depth. This term closely depends on the service curve guaranteed to the child routers.

5.3 Delay Bound Analysis

We propose to compute the **maximum delay bound** of the cluster-tree WSN, which is the delay bound of a data flow sent by a node in the lowest depth (*maxDepth*+1) to reach the root. There are two approaches to compute this delay bound.

The First Approach (per-hop delay bounds for aggregates)

The first approach consists in computing the per-hop delay bounds of the aggregate input flows, and then deducing the end-to-end delay bound as the sum of per-hop delays. This approach was used in [11, 12].

The maximum per-hop delay bound in a router at a depth $\left(maxDepth-i\right)$ can be obtained using Eq. (3) applied to the input arrival curve $\overline{\alpha}_{maxDepth-i}\left(t\right)$ and to the service curve $\beta_{maxDepth-(i+1)}\left(t\right)$. Assuming inequality (21) is satisfied, the delay bound is expressed as:

$$D_{maxDepth-i} = \frac{\overline{b}_{maxDepth-i}}{R_{maxDepth-(i+1)}} + T_{maxDepth-(i+1)} \qquad \textbf{(24)}$$

where $\overline{b}_{maxDepth-i}$ is the burst size of $\overline{\alpha}_{maxDepth-i}\left(t\right)$ defined in Eq. (19).

Hence, using this approach, the maximum end-to-end delay bound in the cluster-tree topology is the sum of all maximum per-hop delay bounds and is equal to:

$$D^{e2e}_{\max} = D_{data} + \sum_{i=0}^{maxDepth} D_{maxDepth-i} \quad \text{where} \qquad D_{data} = \frac{b_{data}}{R_{data}} + T_{data} \qquad \textbf{(25)}$$

Note that D_{data} is the delay bound guaranteed to a child node associated to a router at depth *maxDepth*.

This approach is a bit pessimistic, since the delay bound at each hop concerns the **aggregate** input flow at each router. A tighter delay bound is derived next.

***The Second Approach (tighter delay bounds*)**

The idea of the second approach is to use the aggregate scheduling corollary based on Eq. (11) and the service curve concatenation theorem based on Eq. (8). First, we aim to derive the service curve offered to a particular individual flow F among the aggregate by a router at a given depth, using Eq. (11). Then, we deduce the *equivalent service curve* for this particular flow along the path, using Eq. (8). The delay bound will be computed based on the equivalent service curve. This technique has been used in [14].

We consider the tandem of service curve elements as presented in Fig. 5. The approach is based on the following algorithm:

- *Step 1.* β_{last} is equal to the last service curve element (i.e. router) in the tandem.
- *Step 2.* Compute the β_{eq} *equivalent service curve* to an output flow of the previous service curve element $\beta_{\text{last-1}}$ using Eq. (11).
- *Step 3.*Replace $\beta_{\text{last}} = \beta_{\text{last-1}} \otimes \beta_{\text{eq}}$ since the concatenation is also a service curve to the input of $\beta_{\text{last-1}}$. The length of the tandem is then reduced by one.
- *Step 4.* if the tandem length is greater than one, then Go to *Step 1*; else, β_{last} is the equivalent end-to-end service curve.
- *Step 5.* Compute the delay bound using the equivalent service curve applied to the input arrival curve.

It is easy to derive recurrent expressions for the delay bound using the above algorithm, as it is shown in [18]. In Section 6, we show that this approach provides tighter delay bounds than the first one.

6. Application to IEEE 802.15.4/Zigbee

The aforementioned analysis is independent from any specific protocol. In addition, the proposed model is quite interesting for existing cluster-tree WSN protocols that provide guaranteed services, such as LEACH [2] or IEEE 802.15.4/Zigbee [6, 7], and it can be easily used for their worst-case dimensioning. In this section, we show the practical applicability of our approach by instantiating the general model proposed in Section 5 for IEEE 802.15.4/Zigbee cluster-tree WSNs, and provide a methodology for its worst-case dimensioning. The computations are made using MATLAB.

6.1 The IEEE 802.15.4/Zigbee Protocol Features

In IEEE 802.15.4 beacon-enabled mode, beacon frames are periodically sent by a central device, called *PAN Coordinator*, to identify its WPAN and synchronize nodes that are associated with it. Doing so, a superframe structure is defined by (see Fig. 7) (1) the *Beacon Interval* (BI), which defines the time between two consecutive beacon frames, (2) the *Superframe Duration* (SD), which defines the active portion in BI, and is divided into 16 equally-sized time slots, during which frame transmissions are allowed. Optionally, an inactive period is defined if BI > SD. During the inactive period (if it exists), all nodes may enter in a sleep mode to save energy.

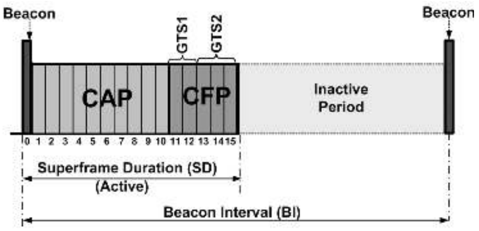


Fig. 7. Beacon Interval and Superframe concepts

BI and SD are determined by two parameters, the *Beacon Order* (*BO*) and the *Superframe Order* (*SO*), respectively, as follows:

$$\left.\begin{aligned} BI &= aBaseSuperframeDuration \cdot 2^{BO} \\ SD &= aBaseSuperframeDuration \cdot 2^{SO} \end{aligned}\right\} for\ \ 0 \le SO \le BO \le 14 \qquad \mathbf{(26)}$$

aBaseSuperframeDuration = 15.36 ms (assuming 250 kbps in the 2.4 GHz frequency band) denotes the minimum duration of the superframe, corresponding to $SO = 0$.

During the SD, nodes compete for medium access using slotted CSMA/CA in the *Contention Access Period* (CAP). For time-sensitive applications, IEEE 802.15.4 enables the definition of a *Contention-Free Period* (CFP) within the SD, by the allocation of *Guaranteed Time Slots* (GTS). It has been shown in [15] that the GTS mechanism provides a rate-latency service curve to nodes that allocate time-slots, where the rate and the latency depend on BI, SD and the number of allocated time slots in the GTS.

While IEEE 802.15.4 only supports the beacon-enabled mode only for star-based topologies, Zigbee has proposed its extension to cluster-tree topologies, where the PAN Coordinator (or Zigbee Coordinator) is identified as the root of the network, and the other *coordinators* as intermediate routers that also generate beacon frames to their child nodes (nodes that are associated to the network through the router). In order to avoid beacon collisions between multiple routers, the Zigbee standard has proposed a beacon scheduling approach such that the superframe durations are non-overlapping during a beacon interval. Fig. 8 illustrates a simple example of this approach for four nodes with the same SD and BI. This approach is suitable for WSNs operating in low duty cycles.

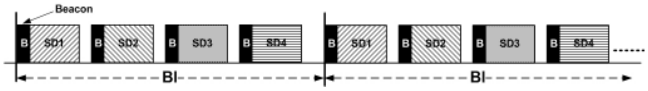


Fig. 8. The beacon scheduling approach in Zigbee

6.2 Dimensioning of an IEEE 802.15.4/Zigbee Cluster-Tree WSN

Let us consider a WSN organized in a cluster-tree topology, with the same parameters as for the example in Fig. 4 (*maxDepth* = 3, N_{router} = 2, N_{child} = 3). N_{router} and N_{child} are the number of routers and nodes that allocate GTSs from their parents. Since the standard does not allow more than seven GTS allocations, N_{router} and N_{child} are constrained as follows:

$$N_{router} + N_{child} \le 7 \qquad \mathbf{(27)}$$

In our application scenario, we assume that all routers have the same SD and BI, and the superframe durations are not overlapping with each other, as presented in Fig. 8. According to our traffic model, we assume that each sensor node (router, or child node) generates a data flow constrained by the arrival curve $\alpha_{data}(t) = b_{data} + r_{data} \cdot t$.

BO and SO settings

It has been shown in [15] that the service curve provided by a GTS allocation intrinsically depends on the setting of BI and SD. Hence, the first problem that we address is to determine the *BO* and *SO* parameters. First, let us assume that *SO* = 0, which corresponds to SD = 15.36 ms for all routers. On the other hand, the number of routers in cluster-tree topology is equal to $\gamma_3(2) = 15$, according to Eq. (22).

The first constraint is that *BO* must be set such that at least 15 superframe durations with *SO* = 0 fit inside the beacon interval to have non-overlapping active periods (as in Fig. 8). It results that:

$$BI \ge \gamma_3(2) \cdot SD \Leftrightarrow 2^{BO} \ge \gamma_3(2) \cdot 2^0 \qquad \mathbf{(28)}$$

As a result, the minimum *BO* is defined as:

$$BO_{\min} = \left\lceil \log_2\left(\gamma_3(2)\right)\right\rceil = 4 \qquad \mathbf{(29)}$$

It is then possible to have 2^4 = 16 SDs inside one BI. The resulting duty cycle for each router is equal to (1/16) = 6.25%.

Bandwidth per time slot

Each allocated time slot of a GTS has a portion used for effective data transmission and a portion used by overheads (inter-frame spacing, acknowledgement frames if required). According to [15], the maximum bandwidth guaranteed by a time slot for *SO* = 0 is equal to 9.38 kbps with 100% duty cycle. Hence, with the above network setting, the bandwidth guaranteed by one allocated time slot in a given superframe is equal to $R_{TS} = 9.38 \text{ kbps} \cdot 0.625$, which gives $R_{TS} = 0.586 \text{ kbps}$.

Sensing input rate limits

Each SD is divided into 16 equal time slots. The standard suggests to have a minimum CAP length of 7.04 ms, which corresponds to approximately 8 time slots with $SO = 0$. Hence, the maximum CFP length in this case is restricted to 8 time slots. Just for illustration purposes, we assume that the maximum CFP length is equal to $L_{CFP} = 14$ (only two time slots are left for the CAP).

With this constraint, a router cannot reserve more than L_{CFP} time slots for its child nodes and routers. Assuming that each child node allocates at most one time slot (arrival rate of sensory data is smaller than R_{TS}), thus the remaining time slots for the N_{router} child routers is equal to ($L_{CFP} - N_{child}$). Since the bandwidth requirement mainly depends on the arrival rate of the sensory data flow (see Eq. (22)), this parameter must be limited in order to not exceed the maximum bandwidth that a router can provide.

Obviously, due to the cumulative upstream flow effect, the maximum bandwidth requirement will be claimed by the child routers of the root. Hence, at the root level, the maximum number of time slots that can be allocated to each child router is equal to $\left\lfloor \left(L_{CFP} - N_{child}\right)/N_{router} \right\rfloor$. The corresponding guaranteed bandwidth is equal to:

$$R_0 = \left\lfloor \frac{L_{CFP} - N_{child}}{N_{router}} \right\rfloor \cdot R_{TS} .$$

According to Eq. (21), the maximum input rate from a child router at a depth equal to 1, i.e. $\bar{r}_1$, satisfies:

$$\bar{r}_1 = \left\lfloor \frac{L_{CFP} - N_{child}}{N_{router}} \right\rfloor \cdot R_{TS} = \gamma_1\left(N_{router}\right) \cdot \bar{r}_{maxDepth}$$

since $\bar{r}_{maxDepth} = \left(N_{child} + 1\right) \cdot r_{data}$, we deduce that:

$$r_{data}^{\max} = \left\lfloor \frac{L_{CFP} - N_{child}}{N_{router}} \right\rfloor \cdot \frac{R_{TS}}{\gamma_1\left(N_{router}\right) \cdot \left(N_{child} + 1\right)} \qquad \textbf{(30)}$$

As a result for $L_{CFP} = 14$, we get $r_{data}^{\max} = 0.104$ kbps. In what follows, we assume that $b_{data} = 200$ bits and $r_{data} = 0.1$ kbps.

Bandwidth requirement and time slot number per router

Depending on its bandwidth requirement, each child router/node must allocate a given number of time slots such that the resulting bandwidth is greater than the input rate, as mentioned in Eq. (21). Hence, we propose to compute the minimum number of time slots required for each child router in each depth, which will enable us to determine the rate-latency service curve in each router. In fact, according to [15], a GTS with n allocated time slots provides a service curve $\beta_{Rn,Tn}(t)$, where the bandwidth $R_n = n \cdot R_{TS}$ and the latency $T_n = BI - n \cdot TS$. TS=SD/16 is the duration of the time slot.

The bandwidth requirement $\bar{r}_{maxDepth-i}$ corresponding to each depth is computed using Eq. (22). Thus, based on Eq. (21), the corresponding number of time slots is expressed as:

$$N_{maxDepth-(i+1)}^{TS} = \left\lceil \frac{\bar{r}_{maxDepth-i}}{R_{TS}} \right\rceil$$

Fig. 9 presents the results for the bandwidth requirement versus the reserved bandwidth per router, for each depth.

Observe that the maximum number of time slots is allocated by a router at depth 1, and is equal to 5. Since there are two child routers and three child nodes, the total number of allocated time slots is 13, which is smaller than $L_{CFP} = 14$.

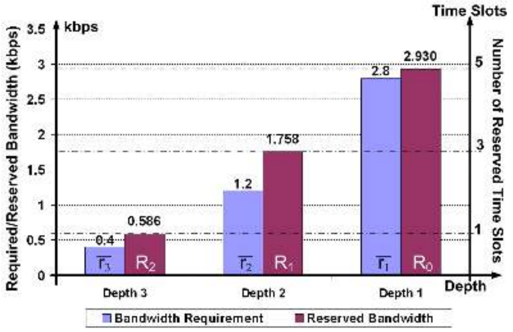


Fig. 9. Bandwidth requirements versus reserved bandwidth per router as a function of the depth

Buffering requirement per router

To estimate the buffering requirement at each router, we apply Eq. (23). The results are presented in Fig. 10, which shows the impact of depth on the buffering requirement.

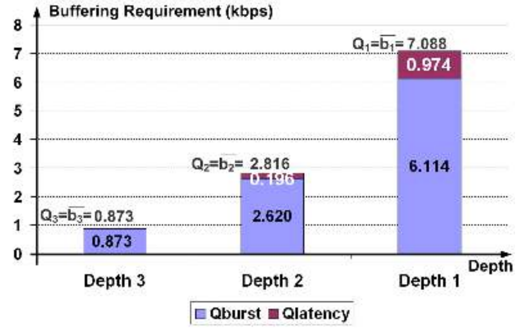


Fig. 10. Buffering requirements per router as a function of the depth

Observe that the cumulative effect of the input burst is more important than the cumulative effect of the service latency on the buffering requirement. This is mainly due to the fact that the input arrival rate is relatively low. The effect of the service latency may be more important for other settings of b_{data}, and r_{data}. Due to space limitations, we do not address the effect of different settings of the arrival curve on the buffering requirement. It can be observed that Q_1 is roughly seven times greater than Q_3, which is basically due to the impact of the $\gamma_{maxDepth-i}\left(N_{router}\right)$ parameter.

Delay bound evaluation

Fig. 11 presents the per-hop delay bounds in each router computed using Eq. (24), and the end-to-end delay bounds obtained by the first approach (using Eq. (25)), and by the second approach (using the recursive algorithm).

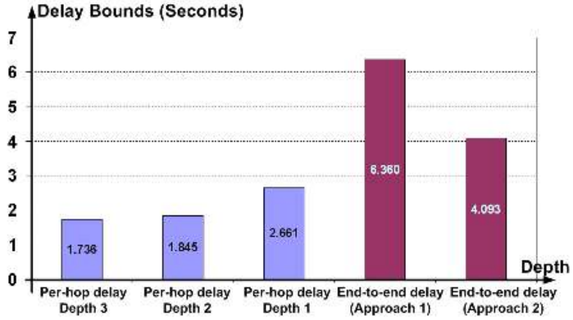


Fig. 11. Per-hop delay bounds and end-to-end delay bounds as a function of the depth

A first observation confirms that the first approach using the sum of per-hop delays is more pessimistic than the second one based on the computation of the end-to-end service curve. The end-to-end delay bounds are quite high, even though the b_{data} and r_{data} are low. This is typically due to the low duty cycle (6.25%). It is possible to reduce the delay bounds by allocating more time slots in a superframe (if possible), and also by finding another beacon scheduling approach such that the beacon interval would be smaller, leading to smaller service latencies and higher bandwidth guarantees, since the duty cycle will increase. Observe also that the per-hop delay bounds are relatively steady, since the buffering and bandwidth requirements are both proportional to the $\gamma_{maxDepth-i}\left(N_{router}\right)$.

7. Concluding Remarks

This paper improves on the state-of-the-art with the proposal of a general model for wireless sensor networks (WSNs) organized in a cluster-tree topology, and a methodology for dimensioning the required network resources and analyzing its timing performance. We assumed a worst-case topology defined by a maximum depth, the maximum number of child routers and child nodes per parent router. We have provided "plug-and-play" recurrent expressions to compute the resource requirements (bandwidth and buffering) and message delay bounds for our WSN model. In addition to this theoretical contribution that can be applied to any cluster-tree network with resource guarantees, we have demonstrated how to apply the general results to the case of cluster-tree IEEE 802.15.4/Zigbee WSNs.

Our methodology provides a practical means to choose the adequate settings of cluster-tree WSNs, for applications with real-time requirements, depending on the available resources, and the delay bound requirement. In fact, one of the important general results is the relation between the resource increase ratio as a function of the depth and the number of routers. For low arrival rates, the bandwidth and buffering requirement increase ratios are both proportional to the factor $\gamma_{maxDepth-i}\left(N_{router}\right)$. In this case, the per-hop delay bounds are roughly steady, as it has been shown in Fig. 11. For higher arrival rates, the impact of the service latency on the buffering requirements would be more important, and consequently, leading to an increased variance on the per-hop delays.

The work carried out in this paper can be extended to evaluate the cluster-tree topology in the downstream direction, and also to study the impact of data aggregation on reducing the resource requirements and the delay bounds.

On the other hand, the model and the methodology that we have proposed trigger new research lines. For instance, they can be used to optimize the dimensioning of IEEE 802.15.4/Zigbee networks. The basic beacon scheduling approach proposed by Zigbee is not adequate for an optimized behavior of cluster-tree WSNs. Thus, one open issue is to optimize the beacon scheduling mechanism in a way that routers at higher depths (closer to the root) will be operating at higher duty cycle than routers at lower depths. The problem is how to choose different *SO* and *BO* for each router depending on its depth (i.e. bandwidth requirement). Another research line concerns a more efficient use of the GTSs by allowing the allocation of the same GTS by more than one node at the same time, as proposed in [16] for the single cluster case.

References

[1] J. A. Stankovic, T. Abdelzaher, C. Lu, L. Sha, and J. Hou, "Real-Time Communication and Coordination in Embedded Sensor Networks," *Proceedings of the IEEE*, vol. 91, pp. 1002-1022, 2003.

[2] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-efficient communication protocols for wireless microsensor networks," in Proceedings of the Hawaii International Conference on Systems Sciences, Hawai, 2000.

[3] G. Pei and C. Chien, "Low power TDMA in Large Wireless Sensor Networks," in Proceedings of the Military Communications Conference (MILCOM'01). Communications for Network-Centric Operations: Creating the Information Force, 2001.

[4] V. A. Kottapalli, A. S. Kiremidjiana, J. P. Lyncha, E. Carryerb, T. W. Kennyb, K. H. Law, and Y. Lei, "Two-Tiered Wireless Sensor Network Architecture for Structural Monitoring," in Proceedings of the 10th Annual International Symposium on Smart Structures and Materials, San Diego (USA), 2003.

[5] G. Gupta and M. Younis, "Fault-Tolerant Clustering of Wireless Sensor Networks," in Proceedings of the IEEE Wireless Communication and Networks Conference (WCNC 2003), New Orleans (Louisiana), 2003.

[6] IEEE-TG15.4, "Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (LR-WPANs)," *IEEE standard for Information Technology*, 2003.

[7] Zigbee-Alliance, "ZigBee specification," *http://www.zigbee.org/*, 2005.

[8] A. Koubâa, M. Alves, and E. Tovar, "IEEE 802.15.4: a Federating Communication Protocol for Time-Sensitive Wireless Sensor Networks," in *Technical Report TR-060201, to appear in Sensor Networks and Configurations: Fundamentals, Techniques, Platforms, and Experiments*, N. P. Mahalik, Ed. Germany: Springer-Verlag, 2006.

[9] T. F. Abdelzaher, S. Prabh, and R. Kiran, "On Real-Time Capacity Limits of Multihop Wireless Sensor Networks," in Proceedings of the IEEE International Real-Time Systems Symposium, Lisbon, Portugal, 2004.

[10] J.-Y. Leboudec and P. Thiran, *A Theory of Deterministic Queuing Systems for the Internet*: Lecture Notes in Computer Science (LNCS), Vol. 2050, 2001.

[11] J. Schmitt and U. Roedig, "Sensor Network Calculus - A Framework for Worst Case Analysis," in Proceedings of the IEEE/ACM International Conference on Distributed Computing in Sensor Systems (DCOSS'05), LNCS 3560, Marina del Rey, USA, 2005.

[12] J. Schmitt and U. Roedig, "Worst Case Dimensioning of Wireless Sensor Networks under Uncertain Topologies," in Proceedings of the 3rd IEEE International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks, (WiOpt'05) Workshop on Resource Allocation in Wireless Networks, Riva del Garda, (Italy), 2005.

[13] J. Schmitt, F. Zdarsky, and U. Roedig, "Sensor Network Calculus with Multiple Sinks," in Proceedings of the IFIP Networking 2006, Workshop on Performance Control in Wireless Sensor Networks, Coimbra, (Portugal), 2006.

[14] L. Lenzini, L. Martorini, E. Mingozzi, and G. Stea, "Tight end-to-end per-flow delay bounds in FIFO multiplexing sink-tree networks," *Performance Evaluation Journal (Elsevier)*, vol. In Press, Corrected Proof, Available online 20 December 2005, 2006.

[15] A. Koubâa, M. Alves, and E. Tovar, "GTS Allocation Analysis in IEEE 802.15.4 for Real-Time Wireless Sensor Networks," in *14th International Workshop on Parallel and Distributed Real-Time Systems (WPDRTS 2006)*. Rhodes Island (Greece): IEEE, 2006.

[16] A. Koubâa, M. Alves, and E. Tovar, "i-GAME: An Implicit GTS Allocation Mechanism in IEEE 802.15.4," in Proceedings of the Euromicro Conference on Real-Time Systems (ECRTS 2006), 2006.

[17] S.-e. Yoo, D. Kim, M.-L. Pham, Y. Doh, E. Choi, and J.-d. Huh, "Scheduling Support for Guaranteed Time Services in IEEE 802.15.4 Low Rate WPAN," in Proceedings of the 11th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA'05), Hong Kong (CHINA), 2005.

[18] A. Koubâa, M. Alves, and E. Tovar, "Modeling and Worst-Case Dimensioning of Cluster-Tree Wireless Sensor Networks: proofs and computation details," Technical Report IPP-HURRAY!, TR-060601, available online http://www.dei.isep.ipp.pt/~akoubaa/submissions.htm, 2006.

[19] L. Lenzini, E. Mingozzi, and G. Stea, "Delay Bounds for FIFO Aggregates: A Case Study," *Computer Communications Journal, Vol. 28, No. 3, pp. 287-299*, 2005.

[20] A. Koubaa and Y. Q. Song, "Evaluation and improvement of response time bounds for real-time applications under non-pre-emptive fixed priority scheduling," *International Journal of Production Research*, vol. 42, pp. 2899-2913, 2004.