Technical Report

Dimensioning and Worst-case Analysis of Cluster-Tree Sensor Networks

Petr Jurcik

Ricardo Severino

Anis Koubaa

Mário Alves

Eduardo Tovar

HURRAY-TR-100302

Version:

Date: 06-01-2010

Dimensioning and Worst-case Analysis of Cluster-Tree Sensor Networks

Petr Jurcik, Ricardo Severino, 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:

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

Abstract

Modeling the fundamental performance limits of Wireless Sensor Networks (WSNs) is of paramount importance to understand their behavior under the worst-case conditions and to make the appropriate design choices. This is particular relevant for time-sensitive WSN applications, where the timing behavior of the network protocols (message transmission must respect deadlines) impacts on the correct operation of these applications. In that direction this paper contributes with a methodology based on Network Calculus, which enables quick and efficient worst-case dimensioning of static or even dynamically changing cluster-tree WSNs where the data sink can either be static or mobile. We propose closed-form recurrent expressions for computing the worst-case end-to-end delays, bu ering and bandwidth requirements across any source-destination path in a cluster-tree WSN. We show how to apply our methodology to the case of IEEE 802.15.4/ZigBee cluster-tree WSNs. Finally, we demonstrate the validity and analyze the accuracy of our methodology through a comprehensive experimental study using commercially available technology, namely TelosB motes running TinyOS.

Dimensioning and Worst-case Analysis of Cluster-Tree Sensor Networks

PETR JURČÍK
CISTER-ISEP, Polytechnic Institute of Porto, Portugal
Department of Control Engineering, Faculty of Electrical Engineering, Czech Technical University in Prague, Czech Republic
and
ANIS KOUBÂA
CISTER-ISEP, Polytechnic Institute of Porto, Portugal
Al-Imam Muhammad Ibn Saud University
and
RICARDO SEVERINO, MÁRIO ALVES and EDUARDO TOVAR
CISTER-ISEP, Polytechnic Institute of Porto, Portugal

Modeling the fundamental performance limits of Wireless Sensor Networks (WSNs) is of paramount importance to understand their behavior under the worst-case conditions and to make the appropriate design choices. This is particular relevant for time-sensitive WSN applications, where the timing behavior of the network protocols (message transmission must respect deadlines) impacts on the correct operation of these applications. In that direction this paper contributes with a methodology based on Network Calculus, which enables quick and efficient worst-case dimensioning of static or even dynamically changing cluster-tree WSNs where the data sink can either be static or mobile. We propose closed-form recurrent expressions for computing the worst-case end-to-end delays, buffering and bandwidth requirements across any source-destination path in a cluster-tree WSN. We show how to apply our methodology to the case of IEEE 802.15.4/ZigBee cluster-tree WSNs. Finally, we demonstrate the validity and analyze the accuracy of our methodology through a comprehensive experimental study using commercially available technology, namely TelosB motes running TinyOS.

Categories and Subject Descriptors: B.4.4 [**Input/output and data communications**]: Performance Analysis and Design Aids—*Worst-case analysis*; C.2 [**Computer-communication networks**]: Network Architecture and Design—*Sensor networks*; C.4 [**Performance of systems**]:

This work was funded by FCT under the CISTER Research Unit (FCT UI 608), by the PLURALITY project (CONCREEQ/900/2001), by the ARTIST2 NoE and by the Ministry of Industry and Trade of the Czech Republic under the project 61 03001.
This paper results from prior conference papers, namely "Modelling and Worst-Case Dimensioning of Cluster-Tree Wireless Sensor Networks", which appeared in *Proceedings of the 27th IEEE Real-Time Systems Symposium (RTSS'06)*, and "Real-Time Communication over Cluster-Tree Sensor Networks with Mobile Sink Behaviour", which appeared in *Proceedings of the 14th IEEE Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA'08)*.
Authors' addresses: CISTER-ISEP Research Unit, Polytechnic Institute of Porto, Rua Dr. Antnio Bernardino de Almeida 431, 4200-72 Porto, Portugal; email: {petr, aska ,rars, mjf, emt}@isep.ipp.pt.

—*Design studies*

General Terms: Design, Experimentation, Performance, Theory

Additional Key Words and Phrases: Cluster-tree, IEEE 802.15.4, Network Calculus, Network dimensioning, Sensor networks, Worst-case analysis, ZigBee

1. INTRODUCTION

The tendency for the integration of computations with physical processes at large scale is pushing research on new paradigms for networked embedded systems design [Stankovic et al. 2005]. Wireless Sensor Networks (WSNs) have naturally emerged as enabling infrastructures for cyber-physical applications that closely interact with external stimulus. Homeland security, physical infrastructures monitoring, health care, building or factory automation are just a few elucidative examples of how these emerging technologies will impact our daily life and society at large.

WSN applications can be of many different types and can have different requirements [Raman and Chebrolu 2008]. For example, an environmental monitoring application that simply gathers temperature readings has less stringent requirements than a real-time tracking application using a set of wireless networked cameras. Therefore, it is crucial that sensor network resources are predicted in advance, to support the prospective applications with a pre-defined Quality-of-Service (QoS) such as transmission delay. Thus, it is important to have adequate methodologies to dimension network resources in a way that the requested QoS of the sensor network application is satisfied [Stankovic et al. 2003]. However, the provision of QoS has always been considered as very challenging due to the usually severe limitations of WSN nodes, such as the ones related to their energy, computational and communication capabilities, and due to communication errors resulting from the unreliable and time-varying characteristics of wireless channels [Bai and Atiquzzaman 2003]. Consequently, it is unrealistic to support hard real-time communications in a WSN. In general, no (wireless) communication channel is error-free thus being able to provide 100% guarantees.

For achieving predictable resource guarantees (e.g. bandwidth and buffer size) all over the sensor networks, it is mandatory to rely on structured logical topologies such as cluster-tree or hexagonal (e.g. [Abdelzaher et al. 2004; Gibson et al. 2007; Prabh and Abdelzaher 2007]) with deterministic routing and MAC (Medium Access Control) protocols. Basically, these network models rely on (1) the use of contention-free MAC protocols (e.g. Time Division Multiple Access (TDMA) or token passing) to ensure collision-free and predictable access to the medium, and (2) the ability to perform end-to-end resource reservation. These represent important advantages of structured and deterministic topologies when compared to what can be achieved in flat mesh-like topologies, where contention-based MAC protocols and probabilistic routing protocols are commonly used. This paper presents a comprehensive framework that provides a scientific approach for quick and efficient worst-case dimensioning of WSN resources to avoid their overflows and to minimize clusters' duty-cycle (maximizing nodes' lifetime), still satisfying that given messages' deadlines are met (i.e. message end-to-end delays are smaller than a certain bound).

WSNs are commonly used for monitoring applications that gather sensory data in a central point called sink. In this paper, we consider the sink as an autonomous entity that does not make part of the WSN, but can be associated to any of its routers through any (wired or wireless) communication means. Thus, the sink's mobility does not impact the WSN topology, but affects the destination of the data flow (any router in the WSN). However, while the statically associated sink is adequate for root centric WSN applications (e.g. an intruder alarm system delivering alerts to the control center), other applications may impose or benefit from collecting data at different network locations (e.g. a doctor with a hand held computer collecting patients' status, a fire-fighter in a rescue mission or a mobile robot in a factory-floor).

In this paper we aim at proposing a simple and efficient system model, an analytical methodology and a software tool that enable the worst-case dimensioning and analysis of static or even dynamically changing WSNs with a cluster-tree topology, assuming bounded communication errors. Consequently, the worst-case performance bounds (e.g. the worst-case end-to-end delay) can be derived for a cluster-tree WSN with bounded resources such as bandwidth and nodes' buffer size. The analytical methodology is based on Network Calculus as a trade-off between complexity and accuracy. The main benefit of using Network Calculus is its generality and simplicity.

We also describe how to instantiate our generic methodology in the design of IEEE 802.15.4/ZigBee [IEEE-TG15.4 2006; ZigBee 2005] networks. These protocols stand as the leading technologies for WSNs (In 2012, 802.15.4 enabled chips will reach 292 million, up from 7 million in 2007 [In-Stat 2009]). Finally, we assess the validity and pessimism of our theoretical model by comparing worst-case results (buffer requirements and message end-to-end delays) with the maximum and average values measured through an experimental test-bed based on Commercial-Off-The-Shelf (COTS) technologies.

Contribution of This Paper

This paper builds upon [Koubaa et al. 2006a] and [Jurcik et al. 2008]. Its main outcome is the provision of a comprehensive framework for the worst-case dimensioning of cluster-tree WSNs and subsequent evaluation of the end-to-end delay bounds for upstream and downstream flows. The design framework presents three main components: (1) a theoretical system model of the cluster-tree network, (2) an analytical methodology for the dimensioning of network resources and delay bound analysis, (3) the impact of the sink's mobility on the worst-case performance of the cluster-tree network. This enables WSN designers to efficiently predict network resources that ensure a minimum QoS during extreme conditions (performance limits). In particular, the paper presents the following contributions:

(1) We provide a generic system model of cluster-tree WSNs and data-flow and cluster scheduling models for the worst-case scenario (Section 3).

(2) We present a simple yet efficient methodology, based on Network Calculus, to characterize input and output flows in each router in the cluster-tree WSN (Section 4) and to derive upper bounds on buffer requirements and per-hop and

end-to-end delays for both upstream and downstream directions (Section 5).

(3) We show how to apply our methodology to dimension IEEE 802.15.4/ZigBee cluster-tree WSNs, as an illustrative example that confirms the applicability of our general approach for specific protocols (Section 6).

(4) We demonstrate the validity of our methodology through an experimental test-bed based on COTS technologies, where we compare the experimental results against the theoretical results and assess the pessimism of our theoretical model (Section 7).

(5) Finally, we analyze the impact of the sink mobility on the worst-case network performance and outline alternatives for sink mobility management (Section 8).

Table I presents the organization of the paper to improve its legibility. Notations and symbols used in the paper are summarized in Table IV (Appendix A).

Topic	Section
Background on Network Calculus	2
System Model	3
Cluster-tree Topology Model	3.1
Data Flow Model	3.2
Time Division Cluster Scheduling	3.3
Data Flow Analysis	4
Upstream Data Flows	4.1
Downstream Data Flows	4.2
Worst-case Network Dimensioning	5
Per-router Resources Analysis	5.1
End-to-End Delay Analysis	5.2
Application to IEEE 802.15.4/ZigBee	6
Overview of the IEEE 802.15.4/ZigBee Protocols	6.1
Guaranteed Bandwidth of a GTS Time Slot	6.2
Characterization of the Service Curve	6.3
IEEE 802.15.4/ZigBee Cluster-Tree WSN Setup	6.4
Performance Evaluation	7
Network Setup	7.1
Analytical Evaluation	7.2
Experimental Evaluation	7.3
Discussion on Mobility Support	8
Conclusions	9

Table I. Content of the paper.

Related Work

The evaluation of the fundamental performance limits of WSNs has been addressed in several research works. In [Hu and Li 2004], the energy-constrained limits of WSNs with respect to the network throughput and operational lifetime has been evaluated. The authors have showed that with fixed node density, lifetime of WSN decreases in the order of $1/n$ as the number of deployed nodes n grows. Even with renewable energy sources, the maximum sustainable throughput in energy-constrained sensor networks scales worse than the capacity based on interference among concurrent transmissions, as long as the physical network size grows with n in an order greater than $\log n$. In [Abdelzaher et al. 2004], the authors have evaluated the real-time capacity of multi-hop WSNs, identifying how much real-time data the network can transfer by their deadlines. A capacity bound has been derived for load-balanced as well as load-unbalanced sensor networks using (ideal) MAC protocols with fixed priority packet scheduling mechanisms. The effects of various link layer multiplexing schemes such as time-division multiplexing and frequency-division multiplexing have been discussed. It has been shown that deadlines are never missed when the network capacity bound is not exceeded. Both above mentioned papers consider unstructured WSNs with ad-hoc deployment. In [Gibson et al. 2007], the authors have explored the fundamental limits for acceptable loads, utilization and delays in multi-hop sensor networks with fixed linear and grid topologies, in case all sensor nodes are equally capable to reach the sink (fair-access criterion). The upper bounds on network utilization and lower bounds on sensing time interval have been derived for any MAC protocol conforming to the fair-access criterion.

Another line of research works deals with soft real-time routing in WSNs. SPEED, MMSPEED and RPAR are some of the routing protocols providing soft end-to-end deadline guarantees in unstructured WSNs with ad-hoc topology. These protocols utilize location information to carry out routing decisions such that each node must be location-aware. SPEED [He et al. 2005] guarantees a uniform delivery speed all over the network so that the end-to-end delay of a message is proportional to the distance between source and sink. Thus, it is possible to predict if the end-to-end deadlines can be met or not. However, the SPEED protocol provides only one network-wide speed, which is not suitable for differentiating various traffic with different deadlines. MMSPEED [Lee et al. 2006] extends SPEED to support different delivery speeds and levels of reliability across the network, such that differentiated QoS can be achieved. Both SPEED and MMSPEED use fixed transmission power. Real-time Power-Aware Routing (RPAR) protocol [Chipara et al. 2006] integrates transmission power control and real-time routing for supporting energy-efficient soft real-time communication. It is based on the assumption that a higher transmission power results in higher speed. The transmission power is increased if the required speed is not satisfied, otherwise if the required speed is satisfied the transmission power is decreased (to improve energy efficiency). Other real-time routing algorithm minimizing the energy consumption in multi-hop WSNs was proposed in [Trdlicka et al. 2007]. The authors have assumed a collision-free MAC protocol, and they have used multicommodity network flow model to schedule the optimal flows' paths in terms of energy consumption while not exceeding links' bandwidths and flows' deadlines. The routing algorithm ensures polynomial-time complexity. On the contrary, in our work we assume cluster-tree WSNs where the routing decisions are simple and time-efficient because each node only interacts with its pre-defined parent/child nodes, and worst-case dimensioning is our main objective.

The worst-case analysis and resource dimensioning of WSNs using Network Calculus has been pursued by Schmitt et al., who proposed the Sensor Network Calculus methodology. In [Schmitt and Roedig 2005a], Sensor Network Calculus was introduced and basic components such as arrival and service curves were defined. The system model assumes generic tree-based topologies with nodes transmitting sensor data towards the sink, that is associated to the root. The authors also proposed a general iterative procedure to compute the network internal flows and, subsequently, the resource requirements and the delay bounds. On the contrary, our work provides recurrent equations so that to avoid iterative computations that are more complex and time consuming and not suitable for large-scale WSNs. In [Schmitt et al. 2007], the previous Sensor Network Calculus framework was extended to incorporate in-network processing features (e.g. data aggregation). In our work, we abstract from the computational resources in the network nodes and from data aggregation. Lenzini et al. [2006] have derived a tighter end-to-end delay bound for each single data flow in tree-based WSNs with FIFO multiplexing nodes. In [Schmitt and Roedig 2005b], the authors searched for the worst-case topology (i.e. the topology that exhibits the worst-case behavior in terms of buffer requirements, delay bounds and network lifetime) in networks with random nodes deployment. Finding the general worst-case topology is a complex task, thus their methodology explores the worst-case tree, constrained on maximum depth and number of child routers, that maximizes the arrival curve of the root node. As compared to the aforementioned papers, our system model is more accurate for the specific case of cluster-tree topologies and the sink can be associated to any router in the WSN.

There have also been several research works on contention-free protocols and mechanisms based on resource reservations to achieve the desired QoS [Caccamo et al. 2002; Facchinetti et al. 2004; Crenshaw et al. 2007]. Caccamo et al. [2002] has proposed a collision-free MAC protocol based on the decentralized earliest-deadline first (Implicit-EDF) packet scheduling algorithm. The key idea is to replicate the EDF schedule at each node to ensure contention-free packet transmission. If the schedules are kept identical, each node will know which one has the message with the shortest deadline and has the right to transmit next. However, it only works when the nodes are organized in hexagonal cells using frequencies different from any of their nearby cells, which requires transceivers supporting multiple frequencies. In addition, the nodes need tight clock synchronization and to know the characteristics of all periodic traffic a priori. These assumptions are uncommon in most WSN applications. Facchineti et al. [Facchinetti et al. 2004] presented a MAC protocol based on implicit-EDF to schedule real-time wireless communication in a network of mobile robots. They assumed the network is not fully linked and developed a consensus protocol to tolerate hidden nodes, allow dynamic schedule updates and dynamic node membership. Like implicit-EDF, their protocol relies on clock synchronization and cooperation among nodes. Each node needs to know the positions

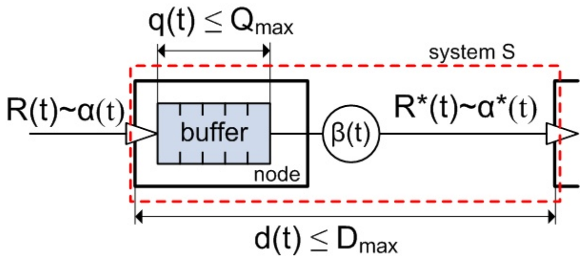


Fig. 1. The basic system model in Network Calculus.

of all other nodes, and to take into account the number of hops a message needs to reach destination. The Robust Implicit-EDF (RI-EDF) protocol [Crenshaw et al. 2007] also builds upon the EDF scheduling algorithm to derive a schedule for the network, implicitly avoiding a contention on the medium. Contrary to I-EDF, RI-EDF protocol assumes no clock synchronization among nodes and a fully linked network, and provides robustness in the presence of node failures or packet losses. In our work, we assume a TDMA-like policy for medium access, such as the Guaranteed Time Slot (GTS) in IEEE 802.15.4 protocol. Differentiation is made based on the flow specification by guaranteeing more time slots.

On the other hand, several research works addressed the use of sink mobility to minimize energy consumption in WSNs [Gandham et al. 2003; Poe and Schmitt 2007]. The proposed approaches use random, predictable or controlled mobility of one or more sinks [Gandham et al. 2003]. Four strategies (random, geographically, intelligent and genetic algorithm based) focusing on optimal sink placement for minimizing the worst-case delay as well as maximizing the lifetime of a WSN have been introduced in [Poe and Schmitt 2007]. Conversely, in our work we compute the worst-case delays and resource requirements for any sink position.

2. BACKGROUND ON NETWORK CALCULUS

Network Calculus [Boudec and Thiran 2004] is a mathematical methodology based on min-plus algebra that applies to the deterministic analysis of queuing/flows in communication networks. This section briefly introduces the aspects of the Network Calculus formalism that are most significant to this paper. For additional details please refer to [Boudec and Thiran 2004].

A basic system model S in Network Calculus consists of a buffered FIFO (First-In, First-Out order) node with the corresponding transmission link (Figure 1). For a given data flow, the input function $R(t)$ represents a cumulative number of bits that have arrived to system S in the time interval $(0, t)$. The output function $R^*(t)$ represents the number of bits that have left S in the same interval $(0, t)$. Both functions are wide sense increasing, i.e. $R(s) \leq R(t)$ if and only if $s \leq t$.

Guaranteeing performance bounds to traffic flow requires that input function $R(t)$ and output function $R^*(t)$ be constrained. In Network Calculus these features are modeled by the concept of arrival and service curves.

Definition 1. *Arrival Curve* $\alpha(t)$ (Figure 2). Let $\alpha(t)$ be a wide-sense increasing function for $t \geq 0$. Then an incoming flow with input function $R(t)$ is upper

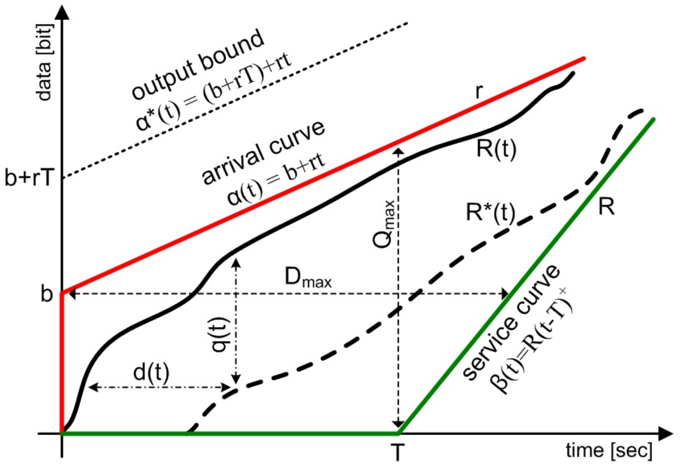


Fig. 2. Example of a cumulative input function $R(t)$ constrained by affine arrival curve $\alpha_{b,r}(t)$ and a cumulative output function $R^*(t)$ constrained by rate-latency service curve $\beta_{R,T}(t)$.

bounded by $\alpha(t)$ iff for $\forall s, 0 \leq s \leq t, R(t) - R(s) \leq (t - s)$. It is also said that $R(t)$ is α smooth or $R(t)$ is constrained by $\alpha(t)$, i.e. $R(t) \sim \alpha(t)$.

Definition 2. *Service Curve* $\beta(t)$ (Figure 2). Consider system S and a flow through S with input and output functions $R(t)$ and $R^*(t)$, respectively. Then S offers to the traversing flow a service curve $\beta(t)$ iff $\beta(t)$ is a wide-sense increasing function with $\beta(0) = 0$, and for $\forall t$ there exists $t_0 \leq t$ such that $R^*(t) - R^*(t_0) \geq \beta(t - t_0)$. This means that an outgoing flow with output function $R^*(t)$ during any period $(t - t_0)$ is at least equal to $\beta(t - t_0)$.

The knowledge of the arrival and service curves enables to determine the performance bounds for a lossless system, namely the delay bound D_{max}, which represents the worst-case delay of a message traversing the system S, and the backlog bound Q_{max}, which represents the worst-case queue length of a flow, i.e. indicates the minimum buffer size requirement inside the system S. Let a flow with input function $R(t)$, constrained by arrival curve $\alpha(t)$, traverses a system S that offers a service curve $\beta(t)$. It results that:

Definition 3. The *Delay Bound* D_{max} is the maximum horizontal distance between $\alpha(t)$ and $\beta(t)$, and for $\forall t \geq 0$ the delay $d(t)$ satisfies:

$$d(t) \leq \sup_{s \geq 0} \left\{ \inf \left\{ \tau \geq 0 : \alpha(s) \leq \beta(s + \tau) \right\} \right\} = D_{max} \tag{1}$$

Definition 4. The *Backlog Bound* Q_{max} is the maximum vertical distance between $\alpha(t)$ and $\beta(t)$, and for $\forall t \geq 0$ the backlog $q(t)$ satisfies:

$$q(t) \leq \sup_{s \geq 0} \left\{ \alpha(s) - \beta(s) \right\} = Q_{max} \tag{2}$$

In Network Calculus, it is also possible to express an upper bound for an outgoing flow with output function $R^*(t)$, called *output bound*.

Definition 5. *Output Bound* $\alpha^*(t)$. Assume that a flow with input function $R(t)$, constrained by arrival curve $\alpha(t)$, traverses a system S that offers a service curve $\beta(t)$. Then, the output function $R^*(t)$ is upper bounded by the following output bound $\alpha^*(t)$:

$$\alpha^*(t) \leq (\alpha \odot \beta) \geq \alpha(t) \tag{3}$$

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

$$(f \odot g)(t) = \sup_{s \geq 0} \Big\{ f(t+s) - g(s) \Big\} \qquad \text{for } \forall t \in \mathbb{R} \tag{4}$$

So far we have handled a system S as a single buffered node (Figure 1). However, system S might also be a sequence of nodes or even a complete network. In this case, the concatenation theorem enables to investigate a set of nodes in sequence as a single node.

Definition 6. *Concatenation Theorem.* Assume a flow with input function $R(t)$ traversing system S_1 and S_2 in sequence, where S_1 offers service curve $\beta_1(t)$ and S_2 offers $\beta_2(t)$. Then the concatenation of these two systems offers the following single service curve $\beta(t)$ to the traversing flow:

$$\beta(t) = (\beta_1 \otimes \beta_2)(t) = (\beta_2 \otimes \beta_1)(t) \tag{5}$$

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

$$(f \otimes g)(t) = \inf_{0 \leq s \leq t} \Big\{ f(t-s) + g(s) \Big\} \qquad \text{for } \forall t \geq 0 \tag{6}$$

Min-plus convolution has several important properties, including being commutative and associative. Furthermore, convolution of concave curves is equal to their minimum [Lenzini et al. 2006].

Due to the aggregation of the data flows in the direction of the sink, each router must provide a service curve $\beta(t)$ to the aggregated data flow. Thus, the delay and backlog bounds are computed for the aggregated data flow traversing the router. On the other hand, using the aggregate scheduling theorem, tighter bounds can be computed for each individual data flow traversing the network. In this work, we use both approaches (i.e. aggregated flow per router and individual flow over network) to compare the results.

Definition 7. *Aggregate Scheduling Theorem.* Consider a lossless node multiplexing two data flows, 1 and 2, in FIFO order. Assume that flow 2 is constrained by the arrival curve $\alpha_2(t)$ and the node guarantees a service curve $\beta(t)$ to the aggregate of these two flows. Define the family of functions as:

$$\beta_1^{eq}(t, \Theta) = \Big[\beta(t) - \alpha_2(t - \Theta)\Big]^+ \cdot 1_{\{t > \Theta\}} \tag{7}$$

where notation $1_{\{expr\}}$ is equal to 1 if *expr* is true, and 0 otherwise, and $(x)^+$ denotes $\max(0, x)$. Then for $\forall \Theta \geq 0, \beta_1^{eq}(t, \Theta)$ is an equivalent service curve guaranteed for flow 1.

So far we have considered an abstract Network Calculus model. The accuracy of the worst-case bounds depends on how tightly the selected arrival and service curves follow the real network behavior. Different types of arrival and service curves have been proposed in Network Calculus (e.g. [Boudec and Thiran 2004; Schmitt and Roedig 2005a]). However, the affine arrival curve and rate-latency service curve are the most used (as illustrated in Figure 2), since they lead to a fair trade-off between computing complexity and accuracy (approximation to the real system behavior), as it will be shown in this paper.

The *affine arrival curve* (Figure 2) is defined as $\alpha_{b,r}(t) = b + r \cdot t$ for $\forall t > 0$ and 0 otherwise, where b is called burst tolerance, which is the maximum number of bits that can arrive simultaneously at a given time to the system S, and r is the average data rate. This type of arrival curve represents a data flow based on the average sensing rate with short-term fluctuations given by the burst tolerance, i.e. it allows a node to send b bits at once, but no more than an average of r bits per second over a long run.

The *rate-latency service curve* is defined as $\beta_{R,T}(t) = R \cdot (t - T)^+$, where $R \geq r$ is the guaranteed forwarding rate, T is the maximum latency of forwarding data (both depend on the nodes features, such as processing speed and resource allocation mechanism). If $r > R$, the bounds are infinite.

Hereafter, we consider a system S that guarantees a rate-latency service curve $\beta_{R,T}(t)$ and that stores input data in a FIFO buffer. Then, the performance bounds D_{max} and Q_{max} (see Figure 2 for additional intuition) guaranteed to the data flow, constrained by the affine arrival curve $\alpha_{b,r}(t)$ and traversing system S, are easily computed as:

$$D_{max} = \frac{b}{R} + T \qquad\qquad Q_{max} = b + r \cdot T \tag{8}$$

Note that the first term b/R is interpreted as the part of the delay due to the burstiness of the input flow, whereas T is due to the latency of the node.

An application of Eq. (3) to a data flow constrained by affine arrival curve $\alpha_{b,r}(t)$ and traversing system S guaranteeing a rate-latency service curve $\beta_{R,T}(t)$, the output bound of this data flow is expressed as (the proof can be found in [Koubaa et al. 2006b]):

$$\alpha^*(t) = \alpha_{b,r}(t) \oslash \beta_{R,T}(t) = \alpha_{b,r}(t) + r \cdot T = (b + r \cdot T) + r \cdot t = b^* + r^* \cdot t \tag{9}$$

According to the aggregate scheduling theorem, we assume that $\alpha_2(t)$ is a affine arrival curve and $\beta(t)$ is a rate-latency service curve, then an equivalent service curve for data flow 1 is expressed as:

$$\beta_1^{eq}(t, \Theta) = (R - r_2) \cdot \left[t - \left(\frac{b_2 + r_2(T - \Theta)}{R - r_2} + T \right) \right]^+ \cdot 1_{\{t > \Theta\}} \tag{10}$$

Hereafter, we omit repeating that arrive curves are affine and service curves are rate-latency, and that all curves are functions of time whenever doing so does not generate ambiguity.

3. SYSTEM MODEL

This section defines the cluster-tree topology and data flow models that will be considered in the analysis. It also elaborates on the worst-case cluster scheduling; that is, the time sequence of clusters' active portions leading to the worst-case end-to-end delay for a message to be routed to the sink. To ensure predictable performance of a WSN, the network topology and data flows must be bounded. To provide closed-form recurrent expressions for computing the worst-case performance bounds in a WSN, the network topology and data load must be balanced. The unbalanced cluster-tree WSN with unbalanced data flows requires specific and complex analysis, i.e. the worst-case performance bounds must be computed separately for each data flow and each subtree. The unbalanced network case has been analyzed in [Schmitt and Roedig 2005b]. However, the approach is too complex to be effectively used in practice, and derived results do not provide a direct solution.

Network communication protocols, e.g. at the data link layer, are able to detect most communication errors and, in some cases, correct some of them. The ultimate objective of communication protocols is to guarantee that messages arrive to the destination logically correct and on time. A corrupted or lost message can be detected by simple checksum or acknowledgment techniques, respectively, and it can be restored by a retransmission mechanism, for example. Of course, it is possible to lose all data frames due to communication errors, but this is an extreme case where the application cannot have any guarantee. Our analysis simply does not consider such cases, and assumes that a maximum bound exists on the number of retransmissions with a certain confidence. Even if we have to deal with some unknown parameters, such as channel error, we must assume that there is an upper bound on the maximum number of retransmissions, otherwise, the analysis will not be possible. In fact, to engineer applications with certain guarantees, we must have a certain confidence on the channel, and this can be done by empirically analyzing the maximum number of losses of the channel prior to a given deployment.

3.1 Cluster-Tree Topology Model

Cluster-tree WSNs feature a tree-based logical topology where nodes are organized in different groups, called *clusters*. Each node is connected to a maximum of one node at the lower depth, called *parent node*, and can be connected to multiple nodes at the upper depth, called *child nodes* (by convention, trees grow down). The multi-hop communication is simple and time-efficient because each node only interacts with its pre-defined parent and child nodes.

A cluster-tree topology contains two main types of nodes: routers and end-nodes (refer to Figure 3). The nodes that can associate to previously associated nodes and can participate in multi-hop routing are referred to as *routers* (R_{ij}, i.e the j^{th} router at depth i). The leaf nodes that do not allow association of other nodes and do not participate in routing are referred to as *end-nodes* (N). The router that has no parent is called *root* and it might hold special functions such as identification, formation and control of the entire network. Note that the root is at depth zero. Both routers and end-nodes can have sensing capabilities, therefore they are generally referred to as *sensor nodes*. Each router forms its cluster and is referred to as *cluster-head* of this cluster (e.g. router R_{11} is the cluster-head of cluster$_{11}$).

All child nodes (i.e. end-nodes and routers) of a cluster-head are associated to its cluster, and the cluster-head handles all their data transmissions. It results that each router (except the root) belongs to two clusters, once as a child and once as a parent (i.e. a cluster-head).

In this work, we aim at specifying the *worst-case cluster-tree topology* which contains the maximum number of nodes in the network, i.e. the network topology configuration that leads to the worst-case performance. In the worst-case, when we reach the maximum depth, and all routers have the maximum number of associated child end-nodes and routers, the topology will be balanced (regular). However, a particular WSN can have unbalanced or even dynamically changing cluster-tree topology, but it can never exceed the worst-case topology. The irregularities in a particular topology introduce some pessimism to our analysis. On the other hand, given any network deployment several cluster-tree logical topologies can be found. Depending on the application, the network designer should select the most regular topology in design time to reduce the pessimism of the worst-case results.

The worst-case cluster-tree topology is graphically represented by a rooted balanced directed tree [Diestel 2000] defined by the following three parameters (derived from the ZigBee [ZigBee 2005] specification):

H	Height of the tree, i.e. the maximum number of logical hops for a message from the deepest router to reach the root (including the root as a final hop). A network with only a root has a height of zero.
$N_{end-node}^{MAX}$	Maximum number of end-nodes that can be associated to a router and have been allocated resource guarantees (e.g. time slots or bandwidth).
N_{router}^{MAX}	Maximum number of child routers that can be associated to a parent router and have been allocated resource guarantees.

The depth of a node is defined as the number of logical hops from that node to the root. The root is at depth zero, and the maximum depth of an end-node is $H+1$. Note that a cluster-tree WSN may contain additional nodes per router than those defined by N_{router}^{MAX} and $N_{end-node}^{MAX}$ parameters. However, these additional nodes cannot be granted guaranteed resources.

Data gathering (all-to-one) and data dissemination (one-to-all) are two fundamental traffic patterns in WSNs [Prabh 2007]. In this paper, we only assume the former, so called *convergecast*, where the sink gathers sensory data from all sensor nodes. We consider the sink to be an autonomous and topology-independent static or mobile entity. The mobile behavior means that a sink moves arbitrarily within a cluster-tree WSN and can be associated to any router within communication range. The router to which the sink is associated in a given moment is referred to as sink router. There can be more than one mobile sink in a WSN, but we assume that only one is active (i.e. gathers the sensory data) at a given time. Hence, we specify another parameter, $H_{sink} \in (0, H)$, to represent the maximum depth of the sink router in a cluster-tree topology, at a given moment. Note that the sink can be also statically attached to the root, i.e. $H_{sink} = 0$. In this case, the network contains only upstream flows (i.e. from child nodes to the root) [Koubaa et al. 2006a].

Our terminology and conventions are as illustrated in Figure 3, corresponding to a configuration where $H = 2$, $N_{end-node}^{MAX} = 3$, $N_{router}^{MAX} = 2$, and $H_{sink} = 2$.

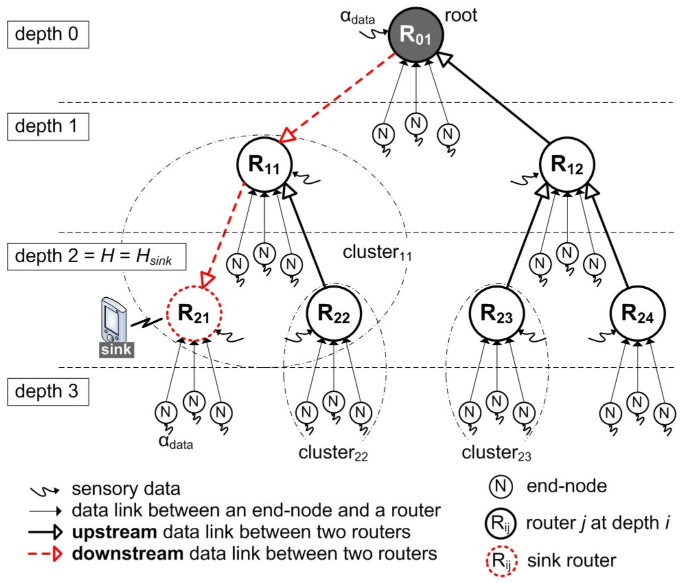


Fig. 3. The worst-case cluster-tree topology model corresponding to a configuration where $N_{end-node}^{MAX} = 3$, $N_{router}^{MAX} = 2$, $H_{sink} = 2$, and $H = 2$.

3.2 Data Flow Model

We assume that all sensory data is sent to the sink router without any in-network processing on the way. In the worst-case, all sensor nodes are assumed to contribute equally to the network load, sensing and transmitting data upper bounded by the affine arrival curve $\alpha_{data} = b_{data} + r_{data} \cdot t$ (Figure 4), where b_{data} is the burst tolerance and r_{data} is the average data rate. In other words, each sensor node can be equipped with one or more sensors sensing and transmitting sensory data which compose a single sensory flow upper bounded by the arrival curve α_{data}. The affine arrival curve can represent any type of traffic, assuming that it can be bounded. It can represent a periodic or aperiodic traffic [Koubaa and Song 2004], or any other random traffic (VBR traffic). This is the main reason for using this simple but effective and general arrival model: to be independent of any specific pattern/distribution of traffic.

In case of different data flows, α_{data} is considered to represent the upper bound of the highest sensory flow in the network. The analysis will lead to some pessimism if the variance between the highest flow and the others is high, i.e. the pessimism increases with the variance. However, in many WSN applications the variance between data flows is likely to be small, since the sensing events are commonly reported by similar data types (e.g. single-precision floating-point number which occupies 32 bits).

Each end-node is granted a service guarantee from its parent router corresponding to the rate-latency service curve $\beta_{end-node} = R_{end-node} \cdot (t - T_{end-node})^{+}$ (Figure 4), where $R_{end-node} \geq r_{data}$ is the guaranteed link bandwidth and $T_{end-node}$ is the maximum latency of the service. The same service curve is provided to all end-

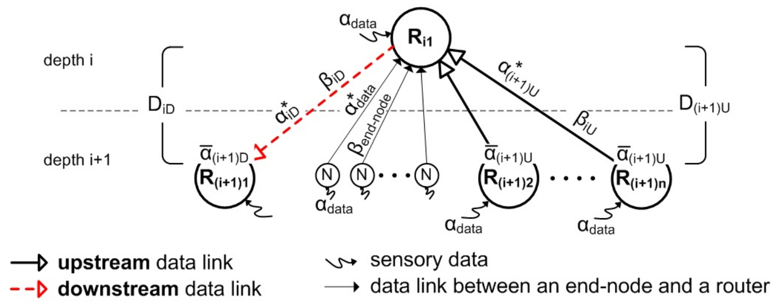


Fig. 4. General data flow model with corresponding arrival curves, service curves and delays.

nodes by their parent routers. By applying Eq. (9) to a flow constrained by the arrival curve $\alpha_{data}(t)$ and that is granted a service curve $\beta_{end-node}$, we obtain the output bound α^*_{data}, which upper bounds the outgoing data flow from any end-node. It results that:

$$\alpha^*_{data} = \alpha_{data} + r_{data} \cdot T_{end-node} \tag{11}$$

On the other hand, the amount of bandwidth allocated by each router depends on the cumulative amount of data at its inputs, which increases towards the sink. Thus, the total input function $R(t)$ of each router depends on the depth, and consists of the sum of the output functions $R^*(t)$ of its end-nodes and child routers. Additionally, the router itself can be equipped with sensing capability producing a traffic bounded by α_{data}. Thus, in general case, the arrival curve constraining the total input function $R(t)$ of a router at a depth i is expressed as (Figure 4):

$$\bar{\alpha}_i = \alpha_{data} + N^{MAX}_{end-node} \cdot \alpha^*_{data} + N^{MAX}_{router} \cdot \alpha^*_{i+1} \tag{12}$$

This result can then be used in Eq. (9). The outgoing flow of a router at depth i, that receives guaranteed service curve β_{i-1}, is constrained by the output bound as follows:

$$\alpha^*_i = \bar{\alpha}_i \odot \beta_{i-1} \tag{13}$$

Hence, the data flow analysis consists in the computation of the arrival curves $\bar{\alpha}_i$ and output bounds α^*_i, using iteratively Eqs. (12) and (13), from the deepest routers until reaching the sink router. After that, the resource requirements of each router, in terms of buffer requirement Q_i and bandwidth requirement R_i, and the worst-case end-to-end delay bounds are computed.

In cluster-tree WSNs where the sink can be associated to any router, data may flow in the upstream and downstream directions. In the upstream case, data is sent from the child nodes to its parent router (so called *upstream flow*), and the parent router must reserve enough bandwidth for the outgoing data of its child nodes. On the contrary, in the downstream case, data is sent from a parent router to its child router (so called *downstream flow*), and the parent router must reserve enough bandwidth for its own outgoing data. Note that if the sink is associated to the root, i.e. $H_{sink} = 0$, the network contains only upstream flows. In what follows, the upstream and downstream flows are marked by the subscripts U and D, respectively (e.g. α^*_{iU}, $\bar{\alpha}_{iD}$). We also assume two types of service curves (i.e. β_{iU} for the upstream flows and β_{iD} for the downstream flows) provided by each parent router at depth i to its child routers at depth $i+1$, and expressed as:

$$\beta_{iU} = R_{iU} \cdot (t - T_{iU})^+ \qquad\qquad \beta_{iD} = R_{iD} \cdot (t - T_{iD})^+ \tag{14}$$

where R_i is the guaranteed link bandwidth, and T_i is the maximum latency that a data flow must wait for a service. To ensure the balanced properties of the worst-case cluster-tree topology assumed in our methodology, the same upstream/ downstream service curves must be guaranteed to all upstream/downstream flows at a given depth. We mean by *balanced properties* that the worst-case cluster-topology and worst-case data flows are balanced such that the upstream/downstream routers in a given depth allocate and use the same resources. Note that the routers forwarding data flows in the upstream direction are referred to as *upstream routers* (e.g. R_{12} or R_{23} in the example in Figure 3) whereas the routers forwarding the downstream flows are referred to as *downstream routers* (e.g. R_{01} or R_{11}). In the same way, the wireless data links are referred to as upstream or downstream.

3.3 Time Division Cluster Scheduling

In general, the radio channel is a shared communication medium where more than one node can transmit at the same time. In cluster-tree WSNs, messages are forwarded from cluster to cluster until reaching the sink. The time window of each cluster is periodically divided into an *active portion* (AP), during which the cluster-head enables data transmissions inside its cluster, and a subsequent *inactive portion*, during which all cluster nodes may enter low-power mode to save energy resources. Note that each router must be awake during its active portion and the active portion of its parent router. To avoid collisions between clusters, it is mandatory to schedule the clusters' active portions in an ordered sequence, that we call *Time Division Cluster Schedule* (TDCS). In other words, the TDCS is equivalent to a permutation of active portions of all clusters in a WSN such that no inter-cluster collision occurs. In case of single collision domain, the TDCS must be non-overlapping, i.e. only one cluster can be active at any time. Hence, the duration of the TDCS's cycle is given by the number of clusters and the length of their active portions. On the contrary, in a network with multiple collision domains, the clusters from different non-overlapping collision domains may be active at the same time. However, finding a TDCS that avoids clusters' collisions in a large-scale WSN with multiple collision domains is a quite complex problem, which is left out of this paper. On the other side, it is easy to see that the duration of overlapping TDCS's cycle is shorter (some active portions can run simultaneously) than the duration of non-overlapping TDCS's cycle. Hence, the non-overlapping TDCS, which introduces some pessimism to the analysis of the large-scale WSNs with multiple collision domains, provides the worst-case bounds of WSN resources, which is the objective. For the sake of simplicity, in our analysis we assume only the non-overlapping TDCS.

Due to the cumulative flow effect, the amount of traffic increases in the direction of the sink such that the maximum flow is reached in the cluster to which the sink is associated (e.g. $cluster_{11}$ in Figure 3). Hence, the duty-cycles of the clusters closer to the sink should be higher (i.e. longer APs) than the ones of the clusters that are farther from the sink, to ensure efficient bandwidth utilization [Koubaa et al. 2007]. Note that the ratio of active portion to the whole period is called *duty-cycle*.

The TDCS significantly affects the resource requirements and delay bounds in cluster-tree WSNs. The number of feasible non-overlapping TDCSs in a network with n routers is equal to the number of permutations given by n factorial ($n!$). Note that for each data flow originated in a given node, there is a corresponding best-case/worst-case TDCS that minimizes/maximizes the end-to-end delay of that flow, respectively. Thus, it is impossible to determine a general best-case or worst-case TDCS meeting the requirements of all data flows. On one hand, the best-case TDCS of a data flow originated in a node x comprises the consecutive sequence of active portions corresponding to the ordered sequence of the clusters traversed along the routing path from x to the sink. On the other hand, the worst-case TDCS (WC-TDCS) comprises the same ordered sequence of active portions, but in the reverse order, which means starting from the sink backward to the node x. The active portions of other clusters, which are not on the routing path, are appended before or after the previously formed sequence in arbitrary order such that a complete WC-TDCS for a given flow is produced. Using our methodology based on the balanced properties of cluster-tree topology model (i.e. balanced topology with balanced load), the WSN resources are dimensioned for non-overlapping WC-TDCS of a data flow originated in the end-node that is farthest from the sink (i.e. a flow along the longest routing path in a WSN). Within a WC-TDCS's cycle the messages belonging to this data flow go only one hop forward. Note that for a particular cluster-tree WSN, the application-specific TDCS, which achieves better performance bounds than WC-TDCS, can be found. However, we are interested in the worst-case dimensioning of general WSNs such that non-overlapping WC-TDCS is our objective.

Let us consider the example in Figure 3, where an end-node of router R_{24} sends sensory data to the sink (that is associated to router R_{21}), i.e. a flow along the longest routing path in the WSN. The best-case TDCS for this flow comprises the continuous sequence of active portions in the following order: AP_{24}, AP_{12}, AP_{01}, AP_{11}, where AP_i is the active portion of $cluster_i$. On the contrary, the WC-TDCS comprises the same sequence but in the reverse order: AP_{11}, AP_{01}, AP_{12}, AP_{24}. The active portions of other clusters (i.e. AP_{21}, AP_{22} and AP_{23}) are appended in arbitrary order such that the complete WC-TDCS is the sequence: AP_{22}, AP_{11}, AP_{01}, AP_{12}, AP_{24}, AP_{21}, AP_{23}, for example.

To reduce the resource requirements of the routers, we introduce the following *priority rule*: *"When a router handles the links in opposite directions (e.g. R_{01} and R_{11} in Figure 3), the incoming flows via upstream data links are served before the outgoing flow via downstream data link."* Using this rule, the end-to-end delay of a data flow can be reduced by one TDCS's cycle duration at the cluster which handles the links in both directions for a given data flow.

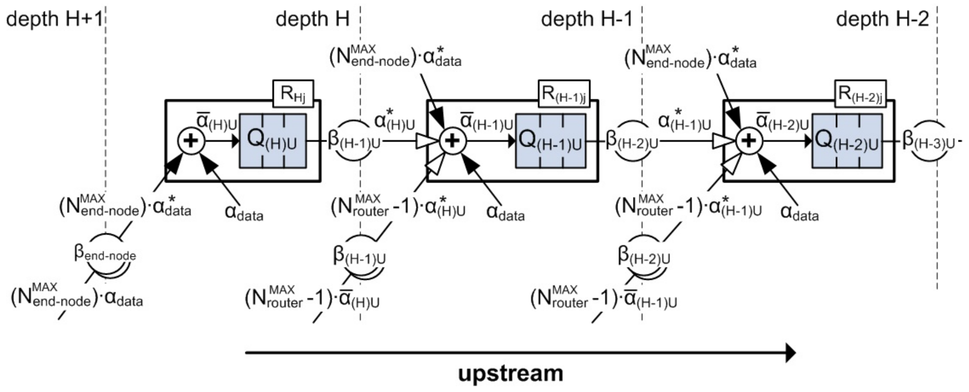


Fig. 5. The queuing system model for upstream flows.

4. DATA FLOW ANALYSIS

This chapter serves as a basic building block for Chapter 5. We derive recurrent equations of the input and output upstream/downstream flows as a function of the router's depth, considering the cluster-tree WSN model presented in Chapter 3. We assume that the end-nodes have sensing capabilities, but the sensing capability of the router nodes is optional. Therefore, we introduce a binary variable ω whose value is equal to 1 if routers have sensing capabilities; otherwise ω is equal to 0.

4.1 Upstream Data Flows

First, we evaluate the arrival curves of the total input upstream flows $\bar{\alpha}_{iU}$ and the upper bounds of the output upstream flows α^*_{iU} depth by depth, using the Network Calculus methodology, starting from depth H (i.e. the deepest routers). In our analysis, we consider the general queuing model for upstream data flows in Figure 5.

Analysis of Depth H+1

At depth $H+1$, there is no router, there are only end-nodes with sensory data flows constrained by the arrival curve α_{data}. A parent router at depth H guarantees the service curve $\beta_{end-node}$ to each of its end-nodes. Thus, according to Eq. (11) the output flow of each end-node is constrained by the output bound α^*_{data}.

Analysis of Depth H

At depth H, the total input flow of each router comprises the sum of the output flows of its $N^{MAX}_{end-node}$ end-nodes and, optionally, its own sensory data flow constrained by α_{data}. Thus, the arrival curve constraining the total input flow is expressed as:

$$\bar{\alpha}_H = \omega \cdot \alpha_{data} + N^{MAX}_{end-node} \cdot \alpha^*_{data} \tag{15}$$

As a result, according to Eq. (11) we get:

$$\bar{\alpha}_H = \left(N^{MAX}_{end-node} + \omega\right) \cdot \alpha_{data} + N^{MAX}_{end-node} \cdot r_{data} \cdot T_{end-node} \tag{16}$$

where $\bar{r}_H = \left(N^{MAX}_{end-node} + \omega\right) \cdot r_{data}$ is the resulting aggregate arrival rate of $\left(N^{MAX}_{end-node} + \omega\right)$ input data flows, and $\bar{b}_H = \left(N^{MAX}_{end-node} + \omega\right) \cdot b_{data} + N^{MAX}_{end-node} \cdot r_{data} \cdot T_{end-node}$ is the burst tolerance. Note that $\bar{\alpha}_H$ corresponds to the first two terms of the arrival curve (Eq. (12)) constraining the total input data flow $\bar{\alpha}_i$. These two terms are constant for upstream and downstream flows at each depth, hence we are using $\bar{\alpha}_H$ without any directional subscripts (i.e. U or D).

The total input flow upper bounded by $\bar{\alpha}_H$ is forwarded by a router at depth H to its parent router at depth $H-1$. This parent router guarantees a service curve $\beta_{(H-1)U} = R_{(H-1)U} \cdot \left(t - T_{(H-1)U}\right)^+$ to its child routers. Hence, according to Eq. (13), the output bound constraining the output upstream flow from a router at depth H is then expressed as $\alpha^*_{(H)U} = \bar{\alpha}_H \oslash \beta_{(H-1)U}$. As a result, applying Eq. (9) we obtain:

$$\alpha^*_{(H)U} = \bar{\alpha}_H + \sigma_{(H-1)} \tag{17}$$

where $\sigma_{(H-1)} = \bar{r}_H \cdot T_{(H-1)U}$.

Analysis of Depth H–1

The total input upstream flow of a router at depth $H-1$ comprises the output upstream flows of its child router in addition to the flow of its child end-nodes, and its own (optional) traffic. Thus, the arrival curve constraining the total input data flow is expressed as $\bar{\alpha}_{(H-1)U} = \left(\omega \cdot \alpha_{data} + N^{MAX}_{end-node} \cdot \alpha^*_{data}\right) + N^{MAX}_{router} \cdot \alpha^*_{(H)U}$. As a result, using Eqs. (15) and (17) we get:

$$\bar{\alpha}_{(H-1)U} = \left(N^{MAX}_{router} + 1\right) \cdot \bar{\alpha}_H + N^{MAX}_{router} \cdot \sigma_{(H-1)} \tag{18}$$

The total input flow upper bounded by $\bar{\alpha}_{(H-1)U}$ is forwarded by a router at depth $H-1$ to its parent routers at depth $H-2$. This parent router guarantees a service curve $\beta_{(H-2)U}$ to its child routers. Hence, according to Eq. (13), the output bound constraining the output upstream flow from a router at depth $H-1$ is then expressed as $\alpha^*_{(H-1)U} = \bar{\alpha}_{(H-1)U} \oslash \beta_{(H-2)U}$. As a result, applying Eqs. (3) and (18) we obtain:

$$\alpha^*_{(H-1)U} = \left(N^{MAX}_{router} + 1\right) \cdot \bar{\alpha}_H + N^{MAX}_{router} \cdot \sigma_{(H-1)} + \sigma_{(H-2)} \tag{19}$$

where $\sigma_{(H-2)} = \left(N^{MAX}_{router} + 1\right) \cdot \bar{r}_H \cdot T_{(H-2)U}$.

Analysis of Depth H–2

Similarly to the previous case, the arrival curve constraining the total input upstream flow of a router at depth $H-2$ is expressed as:

$$\bar{\alpha}_{(H-2)U} = \begin{pmatrix} \left((N^{MAX}_{router})^2 + N^{MAX}_{router} + 1\right) \cdot \bar{\alpha}_H + \\ (N^{MAX}_{router})^2 \cdot \sigma_{(H-1)} + N^{MAX}_{router} \cdot \sigma_{(H-2)} \end{pmatrix} \tag{20}$$

The output flow forwarded from a router at depth $H-2$ to its parent router at depth $H-3$, providing a service curve $\beta_{(H-3)U}$, is constraining by the output bound:

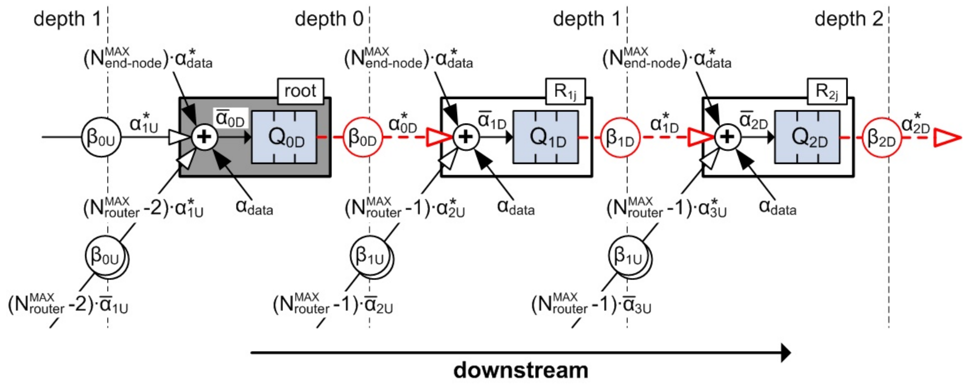


Fig. 6. The queuing system model for downstream flows.

$$\alpha^*_{(H-2)U} = \bar{\alpha}_{(H-2)U} + \sigma_{(H-3)} \tag{21}$$

where $\sigma_{(H-3)} = \left((N^{MAX}_{router})^2 + N^{MAX}_{router} + 1\right) \cdot \bar{r}_H \cdot T_{(H-3)U}$.

Analysis of General Depth i

By recurrence, we can easily prove that the arrival curve, constraining the total input upstream flow of each router at a given depth i, is expressed as follows:

$$\bar{\alpha}_{iU} = \sum_{j=0}^{H-i} (N^{MAX}_{router})^j \cdot \bar{\alpha}_H + \sum_{j=1}^{H-i} \left((N^{MAX}_{router})^j \cdot \sigma_{i+j-1}\right) \tag{22}$$

for $\forall i$, $0 \le i \le H$, where $\sigma_n = \sum_{k=0}^{H-(n+1)} (N^{MAX}_{router})^k \cdot \bar{r}_H \cdot T_{nU}$.

The output bound constraining the output upstream flow from each child router at depth i, receiving a service curve $\beta_{(i-1)}$ from a parent router at depth $i-1$, is expressed as:

$$\alpha^*_{iU} = \bar{\alpha}_{iU} + \sigma_{i-1} = \sum_{j=0}^{H-i} (N^{MAX}_{router})^j \cdot \bar{\alpha}_H + \sum_{j=0}^{H-i} \left((N^{MAX}_{router})^j \cdot \sigma_{i+j-1}\right) \tag{23}$$

for $\forall i$, $0 < i \le H$.

4.2 Downstream Data Flows

In this section, we evaluate the arrival curves of the total input downstream flows $\bar{\alpha}_{iD}$ and the upper bounds of the output downstream flows α^*_{iD} depth by depth, using the Network Calculus methodology, starting from depth 0 (i.e. the root). In our analysis, we consider the general queuing model for downstream data flows as illustrated in Figure 6.

Analysis of Depth 0

At depth 0, there is only one router, the root, and its total input data flow comprises the sum of the output flows of its end nodes, the sum of the output upstream flows of its $(N_{router}^{MAX}-1)$ child routers, and, optionally, its own sensory data flow constrained by α_{data}. Thus, the arrival curve constraining the total input data flow is expressed as $\bar{\alpha}_{0D} = \bar{\alpha}_H + \left(N_{router}^{MAX} - 1\right) \cdot \alpha_{1U}^{*}$. As a result, applying Eq. (23) we obtain:

$$\bar{\alpha}_{0D} = (N_{router}^{MAX})^{H} \cdot \bar{\alpha}_H + \left(N_{router}^{MAX} - 1\right) \cdot \delta_0 \tag{24}$$

where $\delta_0 = \sum_{j=0}^{H-1} \left((N_{router}^{MAX})^{j} \cdot \sigma_j\right)$.

The total input flow upper bounded by $\bar{\alpha}_{0D}$ is forwarded by the root to one of its child routers in the sub-tree where the sink is associated. The root guarantees a service curve $\beta_{0D} = R_{0D} \cdot (t - T_{0D})^{+}$ to this child router at depth 1. According to Eq. (3), the output flow forwarded from the root at depth 0 to a child router at depth 1 is then constrained by the output bound:

$$\alpha_{0D}^{*} = \bar{\alpha}_{0D} \oslash \beta_{0D} = \bar{\alpha}_{0D} + \tau_0 \tag{25}$$

where $\tau_0 = (N_{router}^{MAX})^{H} \cdot \bar{r}_H \cdot T_{0D}$.

Analysis of Depth 1

The total input downstream flow of a router at depth 1 comprises the output downstream flow of its parent router (i.e. the root, at depth 0) in addition to the flow of its child end-nodes/routers, and its own (optional) sensor data traffic. Thus, the arrival curve constraining the total input data flow is expressed as $\bar{\alpha}_{1D} = \bar{\alpha}_H + \left(N_{router}^{MAX} - 1\right) \cdot \alpha_{2U}^{*} + \alpha_{0D}^{*}$. As a result, applying Eqs. (23) and (25) we obtain:

$$\bar{\alpha}_{1D} = \begin{pmatrix} \left((N_{router}^{MAX})^{H} + (N_{router}^{MAX})^{H-1}\right) \cdot \bar{\alpha}_H + \\ \left(N_{router}^{MAX} - 1\right) \cdot \left(\delta_0 + \delta_1\right) + \tau_0 \end{pmatrix} \tag{26}$$

where $\delta_1 = \sum_{j=0}^{H-2} \left((N_{router}^{MAX})^{j} \cdot \sigma_{j+1}\right)$.

The total input flow upper bounded by $\bar{\alpha}_{1D}$ is forwarded by the router at depth 1 to one of its child routers in the sub-tree where the sink is associated. The parent router guarantees a service curve $\beta_{1D} = R_{1D} \cdot (t - T_{1D})^{+}$ to this child router at depth 2. According to Eq. (3), the output flow forwarded from the router at depth 1 to a child router at depth 2 is constrained by the output bound:

$$\alpha_{1D}^{*} = \bar{\alpha}_{1D} \oslash \beta_{1D} = \bar{\alpha}_{1D} + \tau_1 \tag{27}$$

where $\tau_1 = \left((N_{router}^{MAX})^{H} + (N_{router}^{MAX})^{H-1}\right) \cdot \bar{r}_H \cdot T_{1D}$.

Analysis of Depth 2

Similar to the previous case, the arrival curve constraining the total input downstream flow of the router at depth 2 is expressed as $\bar{\alpha}_{2D} = \bar{\alpha}_H + \left(N_{router}^{MAX} - 1\right) \cdot \alpha_{3U}^{*} + \alpha_{1D}^{*}$. As a result, applying Eqs. (23) and (27) we obtain:

$$\bar{\alpha}_{2D} = \begin{pmatrix} \left((N_{router}^{MAX})^{H} + (N_{router}^{MAX})^{H-1} + (N_{router}^{MAX})^{H-2}\right) \cdot \bar{\alpha}_H + \\ \left(N_{router}^{MAX} -\right) \cdot \left(\delta_0 + \delta_1 + \delta_2\right) + \tau_0 + \tau_1 \end{pmatrix} \tag{28}$$

where $\delta_2 = \sum_{j=0}^{H-3}\left((N_{router}^{MAX})^j \cdot \sigma_{j+2}\right)$.

The output flow from the router at depth 2, guaranteeing a service curve $\beta_{2D} = R_{2D} \cdot (t - T_{2D})^+$, to a child router at depth 3 is upper bounded by the output bound:

$$\alpha_{2D}^* = \bar{\alpha}_{2D} \oslash \beta_{2D} = \bar{\alpha}_{2D} + \tau_2 \tag{29}$$

where $\tau_2 = \left((N_{router}^{MAX})^H + (N_{router}^{MAX})^{H-1} + (N_{router}^{MAX})^{H-2}\right) \cdot \bar{r}_H \cdot T_{2D}$.

Analysis of General Depth i

By recurrence, we generalize the analysis for a general depth i. The arrival curve constraining the total input downstream flow of a router at a given depth i, for i = 0, , (H_{sink}-1), is then expressed as:

$$\bar{\alpha}_{iD} = \sum_{j=0}^{i}(N_{router}^{MAX})^{H-j} \cdot \bar{\alpha}_H + \left(N_{router}^{MAX} - 1\right) \cdot \sum_{j=0}^{i}\delta_j + \sum_{j=0}^{i-1}\tau_j \tag{30}$$

for $\forall i$, $0 \leq i \leq H$, where

$$\delta_n = \sum_{k=0}^{H-(n+1)} \left((N_{router}^{MAX})^k \cdot \sigma_{k+n}\right)$$

$$\sigma_n = \sum_{k=0}^{H-(n+1)} (N_{router}^{MAX})^k \cdot \bar{r}_H \cdot T_{nU}$$

$$\tau_n = \sum_{k=0}^{n}(N_{router}^{MAX})^{H-k} \cdot \bar{r}_H \cdot T_{nD}$$

The upper bound of the output downstream flow from a parent router at depth i, guaranteeing a service curve β_{iD}, towards its child router at depth $i+1$ is expressed as:

$$\begin{aligned}\alpha_{iD}^* &= \bar{\alpha}_{iD} + \tau_i = \\ &\sum_{j=0}^{i}(N_{router}^{MAX})^{H-j} \cdot \bar{\alpha}_H + \left(N_{router}^{MAX} - 1\right) \cdot \sum_{j=0}^{i}\delta_j + \sum_{j=0}^{i}\tau_j\end{aligned} \tag{31}$$

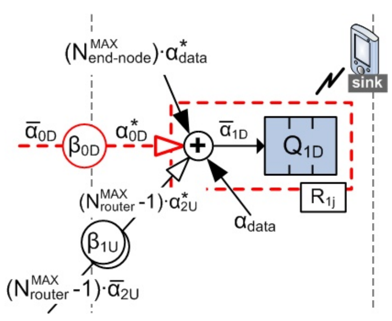
for $\forall i$, $0 \leq i < H_{sink}$.

Note that the sink can be associated to the router at a depth lower than the height of the cluster-tree, i.e. $H_{sink} < H$ (Figure 7a) or equal to the height of the cluster-tree, i.e. $H_{sink} = H$ (Figure 7b).

For $H_{sink} < H$, the arrival curve constraining the total input downstream flow is expressed as:

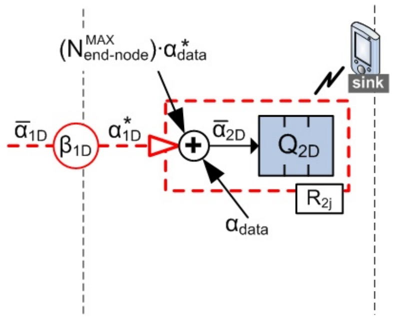
$$\bar{\alpha}_{(H_{sink})D} = \bar{\alpha}_H + N_{router}^{MAX} \cdot \alpha_{(H_{sink}+1)U}^* + \alpha_{(H_{sink}-1)D}^* \tag{32}$$

On the other hand, if $H_{sink} = H$, the arrival curve constraining the total input downstream flow is expressed as:



$(N^{MAX}_{end\text{-}node})\cdot\alpha^*_{data}$
sink
$\bar{\alpha}_{0D}$
β_{0D}
α^*_{0D}
$\bar{\alpha}_{1D}$
Q_{1D}
R_{1j}
$(N^{MAX}_{router}-1)\cdot\alpha^*_{2U}$
α_{data}
β_{1U}
$(N^{MAX}_{router}-1)\cdot\bar{\alpha}_{2U}$

(a) $H_{sink} = 1 < H$ (depth=1)



(b) $H_{sink} = 2 = H$ (depth=2)

Fig. 7. The locations of a sink router and correspondent data flows.

$$\bar{\alpha}_{(H_{sink})D} = \bar{\alpha}_H + \alpha^*_{(H_{sink}-1)D} \tag{33}$$

5. WORST-CASE NETWORK DIMENSIONING

Supporting time-sensitive WSN applications implies to predict and guarantee bounded (worst-case) end-to-end communication delays. To ensure bounded end-to-end delays and to avoid buffer overflow, network resources must be known in advance, and dimensioned along the path from a source to a sink. In this section, we derive the upper bounds on bandwidth and buffer requirements, and per-hop and end-to-end delays for both upstream and downstream data flows.

5.1 Per-Router Resources Analysis

We aim at specifying the minimum bandwidth of each upstream/downstream data links and the minimum buffer size at each router needed to store the bulk of data incoming through the router's inputs.

Bandwidth Requirements

Consider a parent router at depth i providing a service curve β_{iU} or β_{iD} to its child routers at depth $i+1$ in upstream or downstream direction, respectively (see Figure 3).

In the upstream case, the output flow of a child router at depth $i+1$ is constrained by the output bound $\alpha^*_{(i+1)U}$ and dispatched through the upstream link to its parent router at depth i. Thus, to ensure a bounded delay, the guaranteed amount of bandwidth R_{iU} must be greater than or equal to the outgoing data rate $r^*_{(i+1)U}$. As a result, by applying Eqs. (16) and (23) we obtain:

$$\begin{aligned} R_{iU} \geq r^*_{(i+1)U} = \bar{r}_{(i+1)U} = \sum_{j=0}^{H-(i+1)} (N^{MAX}_{router})^j \cdot \bar{r}_H = \\ \Omega_{iU}(H, N^{MAX}_{router}) \cdot \left(N^{MAX}_{end-node} + \omega\right) \cdot r_{data} \end{aligned} \tag{34}$$

for $\forall i$, $0 \leq i < H$.

We call $\Omega_{iU}(H, N_{router}^{MAX})$ the *upstream bandwidth increase factor* at a given depth i. This factor increases with the depth and N_{router}^{MAX}, and it represents the ratio of the additional bandwidth that a router, at a depth i, must guarantee to each of its child routers in the upstream direction as compared to the bandwidth required by a router at depth H.

In the downstream case, the total input flow of the parent router at depth i is constrained by the arrival curve $\bar{\alpha}_{iD}$ and dispatched through a downstream link to its child router. Thus, to ensure a bounded delay, the guaranteed amount of bandwidth R_{iD} must be greater than or equal to the arrival rate of total input flow $\bar{r}_{iD}$. As a result, by applying Eqs. (16) and (30) we obtain:

$$\begin{aligned} R_{iD} \geq \bar{r}_{iD} = r_{iD}^{*} = \sum_{j=0}^{i} (N_{router}^{MAX})^{H-j} \cdot \bar{r}_H = \\ \Omega_{iD}(H, N_{router}^{MAX}) \cdot \left(N_{end-node}^{MAX} + \omega\right) \cdot r_{data} \end{aligned} \tag{35}$$

for $\forall i$, $0 \leq i < H_{sink}$.

Similarly to the previous case, we call $\Omega_{iD}(H, N_{router}^{MAX})$ the *downstream bandwidth increase factor* at a given depth i. This factor represents the ratio of the additional bandwidth that a router, at depth i, must guarantee to its child router in downstream direction as compared to the bandwidth required by a router at the depth H. Note that it is possible to determine the total number of routers in a cluster-tree WSN using the downstream bandwidth increase factor by having $i = H$, which is expressed as:

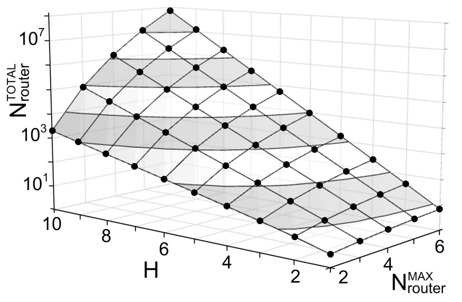
$$\Omega_{HD}(H, N_{router}^{MAX}) = N_{router}^{TOTAL} = \sum_{j=0}^{H} (N_{router}^{MAX})^{H-j} \tag{36}$$

Figure 8a presents the variation of the total number of routers N_{router}^{TOTAL} (i.e. the downstream bandwidth increase factor at depth H) as a function of the height of the tree H and the maximum number of child routers N_{router}^{MAX}.

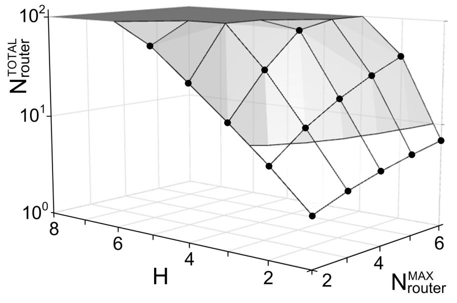
It can be observed that if N_{router}^{MAX} is high (e.g. equal to 5) the impact of the height H on the total number of routers is very significant. Depending on the total number of routers allowed when dimensioning the WSN, high values of the N_{router}^{MAX} parameter can be tolerated if the maximum height of the tree is limited. For instance, if the cluster-tree WSN cannot tolerate more than 10^2 routers (see Fig. 8b) all points in the X, Y, Z axis located below the plan defined by Z = 10^2 are potential solutions to determine the pair (H, N_{router}^{MAX}). For example, with this constraint, the height of the tree cannot exceed 2 if $N_{router}^{MAX} = 5$ or $N_{router}^{MAX} = 6$, while it can be set to 5 if $N_{router}^{MAX} = 2$.

Buffer Requirements

At each router, the incoming data must be stored in a buffer before it is dispatched. To avoid buffer overflow, in the upstream case, the buffer of an upstream router at depth i must be able to store all incoming data, constrained by the arrival curve $\bar{\alpha}_{iU}$, until it is dispatched through the upstream link to the parent router at depth $i - 1$. The required buffer size Q_{iU} of an upstream router at depth i must be at



(a) Total number of routers as a function of the height of the tree H and N_{router}^{MAX}.



(b) Feasible region for the total number of routers $N_{router}^{TOTAL} = 10^2$.

Fig. 8. Total number of routers, i.e. downstream bandwidth increase factor at depth H (logarithmic scale).

least equal to the burst tolerance b_{iU}^* of the output bound α_{iU}^* (see Fig. 2). Hence, according to Eq. (23) we get:

$$Q_{iU} = b_{iU}^* = b_{iU}^{*\ BURST} + b_{iU}^{*\ LATENCY} = \sum_{j=0}^{H-i} (N_{router}^{MAX})^j \cdot \bar{b}_H + \sum_{j=0}^{H-i} \left((N_{router}^{MAX})^j \cdot \sigma_{i+j-1} \right) \tag{37}$$

for $\forall i$, $0 \le i \le H$.

Observe that the upstream buffer requirement is the sum of two terms. The first term is the sum of burst tolerances of the sensory data flows b_{data} of all sensor nodes inside all sub-trees of a given router. The second term represents the cumulative effect of the service latency at each depth for upstream flows.

In the downstream case, the buffer of a downstream router at depth i must be able to store all incoming data, constrained by the arrival curve $\bar{\alpha}_{iD}$, until it is dispatched through the downstream link to a child router at depth $i+1$. The required buffer size Q_{iD} of the downstream router at depth i must be at least equal to the burst tolerance b_{iD}^* of the output bound α_{iD}^* (see Fig. 2). Hence, according to Eq. (31) we get:

$$Q_{iD} = b_{iD}^* = b_{iD}^{*\ BURST} + b_{iD}^{*\ LATENCY_{UP}} + b_{iD}^{*\ LATENCY_{DOWN}} = \sum_{j=0}^{i} (N_{router}^{MAX})^{H-j} \cdot \bar{b}_H + \left(N_{router}^{MAX} - 1\right) \cdot \sum_{j=0}^{i} \delta_j + \sum_{j=0}^{i} \tau_j \tag{38}$$

for $\forall i$, $0 \le i < H_{sink}$.

Observe that the buffer requirement is the sum of three terms. Similarly to the upstream case, the first term is related to the burst tolerance b_{data}, and the second term is related to the cumulative effect of the service latencies of upstream flows. The third term represents the cumulative effect of the service latency at each depth for downstream flows. In case of a sink router at depth H_{sink}, the buffer requirement must be greater than or equal to the burst tolerance $\bar{b}_{(H_{sink})D}$ of total input flow $\bar{\alpha}_{(H_{sink})D}$ given by Eq. (32) or Eq. (33).

5.2 End-to-End Delay Analysis

The worst-case end-to-end delay is the delay bound of a data flow transmitted along the longest path in the network. It can be computed using two approaches, as follows.

Per-hop End-to-End Delay

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. In the upstream case, according to Eq. (8) the delay bound between a child router at depth i and its parent router at depth $i-1$ guaranteeing service curve $\beta_{(i-1)U}$ is expressed as $D_{iU} = \bar{b}_{iU}/R_{(i-1)U} + T_{(i-1)U}$. On the other hand, in the case of the downstream flow, the delay bound between a parent router at depth i, which guarantees service curve β_{iD} to its total input downstream flow constrained by arrival curve $\bar{\alpha}_{iD}$, and its child router at depth $i+1$ is expressed as $D_{iD} = \bar{b}_{iD}/R_{iD} + T_{iD}$. Hence, the worst-case end-to-end delay is the sum of all per-hop delay bounds along the longest routing path, as follows:

$$D_{e2e} = D_{end-node} + \sum_{i=1}^{H} D_{iU} + \sum_{i=0}^{H_{sink}-1} D_{iD} \tag{39}$$

where $D_{end-node} = b_{data}/R_{end-node} + T_{end-node}$ is the delay bound between an end-node and its parent router.

This approach is a bit pessimistic, since the delay bound at each router is computed for the aggregation of all input flows. Tighter end-to-end delay bounds can be computed for individual flows, as described next.

Per-flow End-to-End Delay

The idea of this approach is to derive the service curves guaranteed to a particular individual flow F by the routers along the path, using the aggregate scheduling theorem in Eq. (10), and then deduce the network-wide service curve for flow F based on the concatenation theorem. Finally, according to Eq. (8), the end-to-end delay bound of a given flow F is computed using the network-wide service curve applied to the arrival curve of the input flow. The worst-case end-to-end delay is equal to the delay bound of a data flow along the longest routing path in the network. This technique has been used in [Lenzini et al. 2006].

Consider a consecutive sequence of routers along the longest routing path (e.g. from an end-node of router R_{24} to the sink router R_{21} in Fig. 3). The per-flow approach to the worst-case end-to-end delay is based on the following algorithm.

(1) Start from the sink router. If $H_{sink} = 0$ then the index variable $last = 0$ and the network-wide service curve $\beta_w = \beta_{lastU}$; else (i.e. $1 \leq H_{sink} < H$) $last = H_{sink} - 1$, $\beta_w = \beta_{lastD}$, and go to step 4.

(2) The service curve β_w is guaranteed to the aggregate of input upstream flows of an upstream router at depth $last+1$ (i.e. N_{router}^{MAX} flows from child routers at depth

$last + 2$, $N^{MAX}_{end-node}$ sensory data flows from end-nodes, and optional own sensory data flow). Using the aggregate scheduling in Eq. (10), the equivalent service curve β^{eq} is computed for the output flow of a router at depth $last + 2$ upper bounded by $\alpha^{*}_{(last+2)U}$.

(3) Using the concatenation theorem in Eq. (5), replace $\beta_w = \beta^{eq} \otimes \beta_{(last+1)U}$ since the concatenation is also service curve for the output flow of a router at depth $last + 2$. The length of the router's tandem is then reduced by one. Increase the variable $last = last + 1$. If $last = H - 1$, then go to step 7; else go to step 2.

(4) The service curve β_w is guaranteed to the aggregate of input upstream /downstream flows of the downstream router at depth $last$. Using the aggregate scheduling in Eq. (10), the equivalent service curve β^{eq} is computed for the input downstream flow of the downstream router upper bounded by $\alpha^{*}_{(last-1)D}$.

(5) Using the concatenation theorem in Eq. (5), replace $\beta_w = \beta^{eq} \otimes \beta_{(last-1)D}$ since the concatenation is also service curve guaranteed to the output downstream flow of the downstream router at depth $last - 1$.The length of the router's tandem is then reduced by one. Decrease the variable $last = last - 1$. If $last = 0$, then go to step 6; else go to step 4.

(6) The service curve β_w is guaranteed to the aggregate of input upstream flows of the root. Using the aggregate scheduling in Eq. (10), the equivalent service curve β^{eq} is computed for the output flow of an upstream router at depth $last + 1$ upper bounded by $\alpha^{*}_{(last+1)U}$. Then, using the concatenation theorem in Eq. (5), replace $\beta_w = \beta^{eq} \otimes \beta_{lastU}$ since the concatenation is also service curve for the output upstream flow of an upstream router at depth $last + 1$. Go to step 2.

(7) Using the aggregate scheduling in Eq. (10), the equivalent service curve β^{eq} is computed for the output data flow of an end-node at depth $H + 1$ upper bounded by α^{*}_{data}. Then, using the concatenation theorem in Eq. (5), the network-wide service curve $\beta_w = \beta^{eq} \otimes \beta_{end-node}$ guaranteed to the individual sensory data flow constrained by arrival curve α_{data} is computed.

(8) Compute the end-to-end delay bound, Eq. (8), using the network-wide service β_w applied to the arrival curve α_{data} upper bounding a sensory data flow.

In section 7.3, we experimentally prove that this latter approach provides tighter delay bounds than the former per-hop approach.

6. APPLICATION TO IEEE 802.15.4/ZIGBEE

So far, we have analyzed the general methodology for providing timing and buffer guarantees in cluster-tree WSNs with mobile sink behavior independently of any specific communication protocol. In this section, we show how to apply the aforementioned methodology to the specific case of IEEE 802.15.4/ZigBee cluster-tree WSNs.

6.1 Overview of the IEEE 802.15.4/ZigBee Protocols

The IEEE 802.15.4/ZigBee [IEEE-TG15.4 2006; ZigBee 2005] protocols have several appealing properties for WSNs. The IEEE 802.15.4 standard specifies the Physical and Data Link Layers, while the Network and Application Layers are defined by

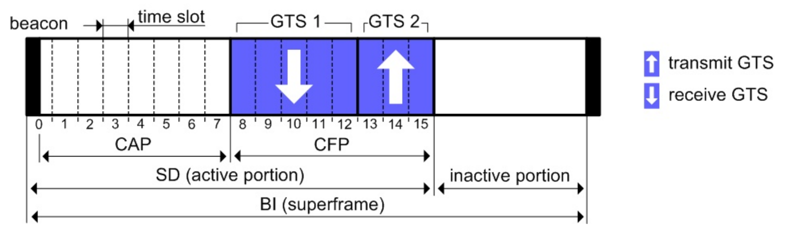


Fig. 9. IEEE 802.15.4 superframe structure.

the ZigBee specification. The MAC (Medium Access Control) supports the beacon-enabled or non beacon-enabled modes that may be selected by a central controller of the WSN, called *PAN coordinator*. We only consider the beacon-enabled mode, since it enables the provision of guaranteed bandwidth through the *Guaranteed Time Slot* (GTS) mechanism.

In beacon-enabled mode, beacon frames are periodically sent by each cluster-head to synchronize nodes that are associated to it and to describe the structure of the superframe (Fig. 9). The superframe, corresponding to the *Beacon Interval* (BI), is defined by the time between two consecutive beacons, and includes an active portion and, optionally, a following inactive portion. During the inactive portion, each node may enter a low-power mode to save energy resources. The active portion, corresponding to the *Superframe Duration* (SD), is divided into 16 equally-sized time slots, during which data transmission is allowed. Each active portion can be further divided into a *Contention Access Period* (CAP) using slotted CSMA/CA for best-effort data delivery, and an optional *Contention Free Period* (CFP) supporting the time-bounded data delivery. Within the CFP, Guaranteed Time Slots can be allocated to a set of child nodes. The CFP supports up to 7 GTSs and each GTS may contain one or more time slots. Each GTS can be used to transfer data either in *transmit direction*, i.e. from child to parent node (upstream flow), or *receive direction*, i.e. from parent to child node (downstream flow). A GTS is activated upon request, where a node explicitly expresses the number of time slots that it wants to allocate from its parent. Hence, using this explicit GTS allocation, the maximum numbers of child routers and end-nodes (that require guaranteed bandwidth) are constrained as follows $N_{router}^{MAX} + N_{end-node}^{MAX} \leq 7$.

The *explicit GTS allocation*, which is natively supported by the IEEE 802.15.4/ZigBee protocols, has the advantage of being simple. However, the GTS resources may quickly disappear, since a maximum of 7 GTSs can be allocated in each superframe. Moreover, the explicit GTS allocation may be not efficient enough in terms of bandwidth utilization for data flows with low arrival rate, since the minimum guaranteed bandwidth of a GTS can be much higher than the arrival rate. To overcome these limitations, the *implicit GTS Allocation MEchanism* (i-GAME) was proposed in [Koubaa et al. 2008]. The i-GAME approach enables the use of a GTS by several nodes, while all their requirements (e.g. bandwidth, delay) are still satisfied. Hence, more than 7 child routers and end-nodes may be associated to a router. On the other hand, the implicit GTS allocation may enlarge the worst-case end-to-end delay.

The structure of the superframe is defined by two parameters, the *Beacon Order*

(BO) and the *Superframe Order* (SO) as follows:

$$\left.\begin{aligned} BI &= aBaseSuperframeDuration \cdot 2^{BO} \\ SD &= aBaseSuperframeDuration \cdot 2^{SO} \end{aligned}\right\} \quad \text{for } 0 \leq SO \leq BO \leq 14$$

where *aBaseSuperframeDuration* = 15.36 ms assuming the 2.4 GHz ISM frequency band with 250 kbps data rate. The beacon order and superframe order shall be equal for all superframes in a WSN [IEEE-TG15.4 2006], i.e. all clusters have the same duty-cycle.

The [IEEE-TG15.4 2006] supports three unlicensed frequency bands: 2.4 GHz (worldwide, 16 channels, 250 kbps), 915 MHz (North America and and some Asian countries, 10 channels, 40 kbps) and 866 MHz (Europe, 1 channel, 20 kbps). In this work, we only consider the 2.4 GHz band with 250 kbps data rate, which is also supported by the TelosB motes [Crossbow 2008], used in our experimental test-bed (refer to Section 7).

While IEEE 802.15.4 in beacon-enabled mode supports only star-based topologies, the ZigBee Specification has proposed its extension to mesh and cluster-tree based topologies. In the particular case of ZigBee cluster-tree networks, a PAN (or ZigBee) Coordinator is identified as the root of the tree and forms the initial cluster. The other routers join the cluster-tree in turn by establishing themselves as cluster-heads, starting to generate the beacon frames for their own clusters. Note that each cluster is active during its SD and inactive during the rest of its BI (if $BO > SO$). To avoid inter-cluster collisions (messages/beacons transmitted from nodes in different clusters), an appropriate cluster scheduling policy must be followed (see Section 3.3). For the sake of simplicity, we assume that all clusters have the same duty-cycle, and the WC-TDCS is non-overlapping. Hence, the WC-TDCS is given by the non-overlapping sequence of equally sized SDs, and the duration of a WC-TDCS's cycle is equal to BI.

6.2 Guaranteed Bandwidth of a GTS Time Slot

Each GTS time slot has a portion used for effective data transmission and a portion of overheads (i.e. inter-frame spacing, and eventual acknowledgment and retransmissions). Consecutive frames are separated by *inter-frame spacing* (IFS). The IFS is equal to a SIFS (Short Inter-Frame Spacing) of a duration of at least 0.192 ms [IEEE-TG15.4 2006], for MAC frame's length smaller than or equal to *aMaxSIFSFrameSize* (= 144 bits [IEEE-TG15.4 2006]). Otherwise, the IFS is equal to a LIFS (Long Inter-Frame Spacing) of a duration of at least 0.64 ms [IEEE-TG15.4 2006], for a MAC frame greater than *aMaxSIFSFrameSize* and smaller than *aMaxPHYPacketSize* (= 1016 bits [IEEE-TG15.4 2006]), which is the maximum size of a MAC frame, called *MPDU* (MAC Protocol Data Unit). Note that the MPDU is prefixed with a 48-bit physical header [IEEE-TG15.4 2006]. The MPDU and physical header together form *PPDU* (Physical Protocol Data Unit) which is dispatched to a wireless channel. In practice, most WSN applications are likely to use frames that are smaller than the maximum allowed size. Thus, in order to achieve more accurate results we introduce the parameter $MPDU_{max}$ representing the user-defined maximum size of MAC frames.

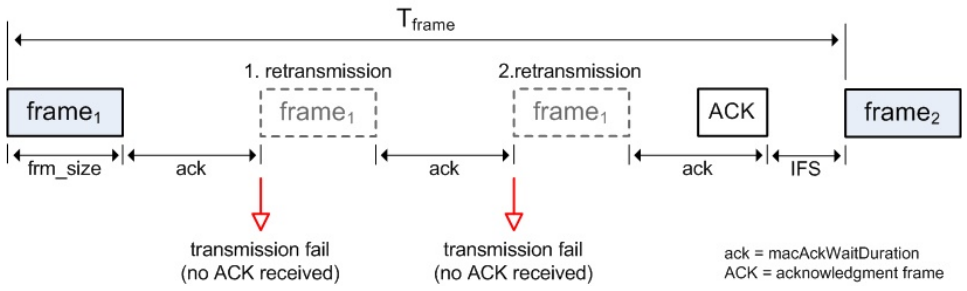


Fig. 10. The worst-case time required for an overall successful transmission of frame$_1$ ($macMaxFrameRetries = 2$).

IEEE 802.15.4 protocol supports acknowledgment and retransmission procedures to minimize communication errors resulting from the unreliable and time-varying characteristics of wireless channels. In case of acknowledged transmissions, the sender waits for the corresponding acknowledgment frame at most *macAckWaitDuration* (= 0.864 ms [IEEE-TG15.4 2006]). If an acknowledgment frame is received within $macAckWaitDuration$, the transmission is considered successful, and no further action regarding retransmission shall be taken. Otherwise, the data transmission (and waiting for the acknowledgment) is repeated up to a maximum of $macMaxFrameRetries$ (range 0–7, default value 3 [IEEE-TG15.4 2006]) times. If an acknowledgment frame is not received after $macMaxFrameRetries$ retransmissions, the transmission is considered failed. Each retransmission decreases guaranteed bandwidth for effective data transmission, increases communication delay and energy consumption (see Section 7.2) such that a fair trade-off between reliability and timeliness of data transmission must be found. Note that for unacknowledged transmission it is assumed that a frame is successfully received and no retransmission is performed (i.e. $macMaxFrameRetries = 0$).

The overall transmission, including the frame (i.e. PPDU), IFS and eventual acknowledgment and retransmissions, must be completed before the end of the current GTS. Otherwise, it must wait until the next GTS. Hence, a GTS can be wasted if no frame is available for transmission, or the remaining time at the end of the GTS is not enough to complete the overall transmission.

We derive the expression for the effective bandwidth (i.e. without overheads) guaranteed by one time slot in a given superframe (Eq. (43)), which is related to the worst-case data transmission. The worst-case time required for the overall successful transmission of a frame (i.e. the last retransmission succeeded - see Fig. 10) is then expressed as:

$$T_{frame} = \begin{pmatrix} (macMaxFrameRetries \cdot \Omega + 1) \cdot \\ (frm\_size/C + ack \cdot \Omega) + IFS \end{pmatrix} \qquad (40)$$

where frm_size is the user-defined maximum size of transmitted frame including MPDU and physical header (i.e. PPDU), C is the data rate (we assume 250 kbps), IFS is the inter-frame spacing, ack stands for $macAckWaitDuration$, $\Omega = 1$ for an acknowledged transmission or $\Omega = 0$ for an unacknowledged transmission.

The worst-case number of frames with user-defined maximum size that can be transmitted during one time slot is then expressed as:

$$N_{frame} = \left\lfloor \frac{TS}{T_{frame}} \right\rfloor \tag{41}$$

where TS is the duration of a time slot and is equal to SD/16. In the remaining time (i.e. $TS - N_{frame} \cdot T_{frame}$), a MAC frame smaller than $MPDU_{max}$ can only be transmitted if the overall transmission can be completed before the end of the GTS. The size of the last frame (PPDU), which can be transmitted within a given GTS, is then expressed as:

$$last\_frm\_size = \left(\frac{TS - N_{frame} \cdot T_{frame} - IFS}{macMaxFrameRetries \cdot \Omega + 1} - ack \cdot \Omega \right) \cdot C \tag{42}$$

If the size of the last frame is smaller than the minimum size of frame, then $last\_frm\_size = 0$.

Finally, assuming a full duty-cycle (i.e. $SO = BO$) the bandwidth guaranteed by one allocated time slot in a given superframe, for effective data transmission, is expressed as:

$$R_{TS}^{100\%} = \frac{N_{frame} \cdot frm\_size + last\_frm\_size}{SD} \tag{43}$$

6.3 Characterization of the Service Curve

Each parent router must reserve a GTS with enough time slots for each of its child nodes. For upstream data links, the resulting bandwidth of GTS, guaranteed by a parent router at depth i and given by N_{iU}^{TS} time slots in transmit direction, must be greater than or equal to the total input arrival rate $\bar{r}_{(i+1)U}$ of a child node at depth $i+1$. On the contrary, for downstream data links, a parent router at depth i must reserve a GTS with N_{iD}^{TS} time slots in receive direction to its child router at depth $i+1$ such that the resulting link bandwidth is greater than or equal to its total input arrival rate $\bar{r}_{iD}$. It results that:

$$N_{iU}^{TS} = \left\lceil \frac{\bar{r}_{(i+1)U}}{R_{TS}} \right\rceil \qquad N_{iD}^{TS} = \left\lceil \frac{\bar{r}_{iD}}{R_{TS}} \right\rceil \qquad N_{end-node}^{TS} = \left\lceil \frac{\bar{r}_{data}}{R_{TS}} \right\rceil \tag{44}$$

Note that $N_{end-node}^{TS}$ is the number of GTS time slots guaranteed to each end-node by its parent router. Hence, a GTS with N_i^{TS} time slots provides rate-latency service $\beta_{R_iT_i}$, where $R_i = N_i^{TS} \cdot R_{TS}$ is the guaranteed bandwidth for effective data transmission and T_i is the maximum latency that a data flow must wait to be served.

The service latencies T_i depend on the TDCS such that their worst-case values are achieved for the non-overlapping WC-TDCS of a data flow along the longest routing path in a WSN. Since our methodology is based on the balanced properties of our cluster-tree topology model, the same service latency, equal to the worst-case

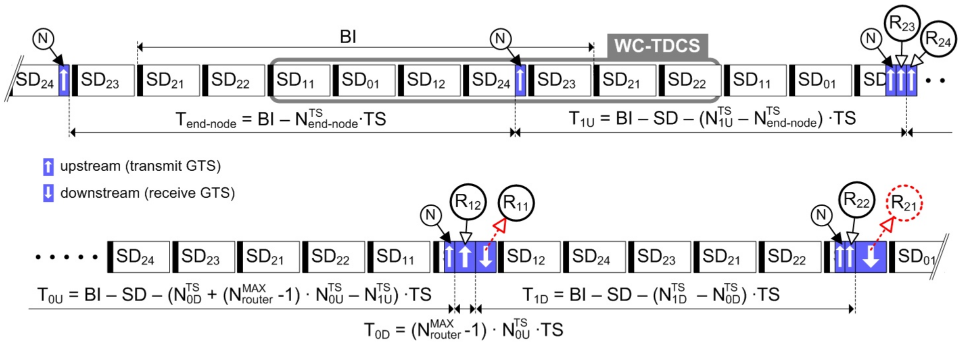

BI
WC-TDCS
SD24
SD23
SD21
SD22
SD11
SD01
SD12
N
R23
R24
R12
R11
R22
R21
upstream (transmit GTS)
downstream (receive GTS)
$T_{end\text{-}node} = BI - N^{TS}_{end\text{-}node} \cdot TS$
$T_{1U} = BI - SD - (N^{TS}_{1U} - N^{TS}_{end\text{-}node}) \cdot TS$
$T_{0U} = BI - SD - (N^{TS}_{0D} + (N^{MAX}_{router} - 1) \cdot N^{TS}_{0U} - N^{TS}_{1U}) \cdot TS$
$T_{1D} = BI - SD - (N^{TS}_{1D} - N^{TS}_{0D}) \cdot TS$
$T_{0D} = (N^{MAX}_{router} - 1) \cdot N^{TS}_{0U} \cdot TS$

Fig. 11. The worst-case service latencies of a flow along the longest routing path in the WSN related to the example in Fig. 3.

one at a given depth (i.e. along the longest path), is provided to all data links at a given depth in upstream/downstream direction. Let us consider the example in Fig. 3, where an end-node of router R_{24} sends sensory data to the sink associated to the router R_{21} (i.e. a flow along the longest routing path). The corresponding WC-TDCS may be given by the following sequence of Superframe Durations, for example: SD_{11}, SD_{01}, SD_{12}, SD_{24}, SD_{23}, SD_{21}, SD_{22} (Fig. 11). The worst-case service latencies at each depth, except depth 0, are given by the distance between the active portions of consecutive clusters on the longest routing path to the sink. At depth 0, the priority rule (Section 3.3) is applied. The duration of WC-TDCS's cycle is equal to the time which spans between two consecutive active portions of the same cluster (i.e. BI).

Note that the service latencies of any application-specific or overlapping TDCS will be equal or shorter than the latencies of non-overlapping WC-TDCS. In these cases, the worst-case latency at a given depth is equal to the longest latency in upstream/downstream direction at this depth, which does not to be equal to the one along the longest routing path in a WSN.

According to Fig. 11, the worst-case service latency guaranteed to a flow over an upstream data link at a given depth is expressed as:

— the latency guaranteed by a router to its end-node:

$$T_{end-node} = \text{BI} - N^{TS}_{end-node} \cdot TS$$

— the latency guaranteed by a router at depth i to a child router at depth $i+1$, for $\forall i$, $0 < i < H$:

$$T_{iU} = \text{BI} - \text{SD} - \left(N^{TS}_{iU} - N^{TS}_{(i+1)U}\right) \cdot TS$$

— the latency guaranteed by the router at depth 0 to the child router at depth 1:

$$T_{0U} = \text{BI} - \text{SD} - \Big(N^{TS}_{0D} + \left(N^{MAX}_{router} - 1\right) \cdot N^{TS}_{0U} - N^{TS}_{1U}\Big) \cdot TS$$

On the other hand, the worst-case service latency guaranteed to a flow over a downstream data link at a given depth is expressed as:

— the latency guaranteed by a router at depth 0 to the child router at depth 1 (priority rule, Section 3.3):

$$T_{0D} = \left(N_{router}^{MAX} - 1\right) \cdot N_{0U}^{TS} \cdot TS$$

— the latency guaranteed by a router at depth i to the child router at depth $i+1$, for $\forall i$, $0 < i < H_{sink}$:

$$T_{iD} = \mathrm{BI} - \mathrm{SD} - \left(N_{iD}^{TS} - N_{(i-1)D}^{TS}\right) \cdot TS$$

6.4 IEEE 802.15.4/ZigBee Cluster-Tree WSN Setup

For our experimental scenario, we consider a simple cluster-tree WSN corresponding to the configuration where $H = 2$, $N_{end-node}^{MAX} = 1$, $N_{router}^{MAX} = 2$. For the sake of simplicity, only end-nodes are equipped with sensing capability (i.e. $\omega = 0$) and generate data flows bounded by the arrival curve α_{data}. We assume $SO = 4$, which is the minimum value that is possible to use without resulting into synchronization problem [Cunha et al. 2008], using open-ZB protocol stack [Cunha et al. 2007] over TinyOS [TinyOS 2008] and MICAz/TelosB motes. This constraint results from the non-preemptive behavior of the TinyOS operating system. According to Eq. (36), the total number of routers is equal to 7. Hence, BO must be set such that at least seven SDs with $SO = 4$ can fit inside the BI without overlapping. In general, we obtain:

$$BI \geq \Omega_{HD}(H, N_{router}^{MAX}) \cdot SD \Leftrightarrow BO_{min} = \left\lceil \log_2\left(\Omega_{HD}(H, N_{router}^{MAX}) \cdot 2^{SO}\right) \right\rceil \quad (45)$$

As a result for $SO = 4$, the minimum BO is equal to 7, such that a maximum of $2^7/2^4 = 8$ SDs can fit in one BI. The maximum duty-cycle of each cluster is then equal to $2^{SO}/2^{BO} = 1/8 = 12.5\%$. Note that to maximize the lifetime of a WSN, the lowest duty-cycles must be chosen (IEEE 802.15.4 supports duty-cycles under 1%). As a result, the inactive portion is extended, and the nodes may stay in low-power mode longer to save energy resources. On the other hand, low duty-cycles enlarge end-to-end delays. Hence, long lifetime is in contrast to the fast timing response of a WSN, so a trade-off must be found. In our example with $SO = 4$, we can get the duty-cycles: 12.5% ($BO = 7$), 6.25% ($BO = 8$), 3.125% ($BO = 9$), and so on.

According to [IEEE-TG15.4 2006], the minimum CAP (i.e. *aMinCAPLength* parameter), which ensures that commands and best-effort data can still be transferred when GTSs are being used, is equal to 7.04 ms, assuming the 2.4 GHz ISM band, which corresponds to 1 time slot with $SO = 4$. Note that the CAP requires minimum 8, 4, 2 or 1 time slots with $SO = 0, 1, 2$ or 3, respectively. The remaining slots can be allocated for GTSs. Hence, the maximum CFP length is equal to $L_{CFP} = 15$ time slots. With this constraint, a router cannot reserve more than L_{CFP} time slots for 7 GTSs maximum, i.e. for its $N_{end-node}^{MAX}$ end-nodes and N_{router}^{MAX} child routers. Assuming that each end-node requires allocation of a GTS with $N_{end-node}^{TS}$ time slots (i.e. $r_{data} \leq N_{end-node}^{TS} \cdot R_{TS}$) from its parent router. Then, each child router can allocate a GTS with the maximum number of time slots equal to $\lfloor (L_{CFP} - N_{end-node}^{TS} \cdot N_{end-node}^{MAX}) / N_{router}^{MAX} \rfloor$. According to Eqs. (34) and (35), the arrival rate r_{data} must be limited in order not to exceed the maximum bandwidth that a parent router can reserve. Obviously, due to the cumulative flow effect, the maximum bandwidth will be required either by the child routers of the root, in case the sink is associated to the root (i.e. $H_{sink} = 0$), or by the sink router, in other cases (i.e. $1 \leq H_{sink} \leq H$).

Thus, for $H_{sink} = 0$, the bandwidth guaranteed by the root to its child routers at depth 1 is expressed as:

$$R_0 = \left\lfloor \frac{L_{CFP} - N_{end-node}^{TS} \cdot N_{end-node}^{MAX}}{N_{router}^{MAX}} \right\rfloor \cdot R_{TS}$$

As a result, applying Eq. (34), we obtain the maximum arrival rate of the sensory data flow as:

$$r_{data}^{MAX} = \left\lfloor \frac{L_{CFP} - N_{end-node}^{TS} \cdot N_{end-node}^{MAX}}{N_{router}^{MAX}} \right\rfloor \cdot \frac{R_{TS}}{\left(\sum_{j=0}^{H-1} (N_{router}^{MAX})^j\right) \cdot \left(N_{end-node}^{MAX} + \omega\right)} \tag{46}$$

for $H_{sink} = 0$.

On the other hand, for $1 \leq H_{sink} \leq H$, the corresponding link bandwidth guaranteed by the parent router at depth $(H_{sink} - 1)$ to the sink router at depth H_{sink} is equal to:

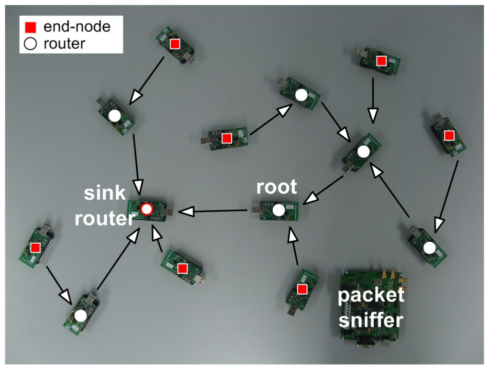
$$R_{(H_{sink}-1)} = \left\lfloor \frac{L_{CFP} - N_{end-node}^{TS} \cdot N_{end-node}^{MAX}}{N_{router}^{MAX}} \right\rfloor \cdot R_{TS}$$

As a result applying Eq. (35), we obtain the maximum arrival rate of the sensory data flow as:

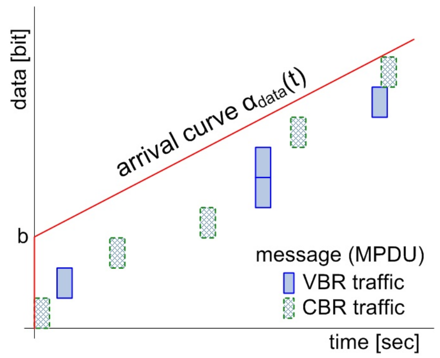
$$r_{data}^{MAX} = \left\lfloor \frac{L_{CFP} - N_{end-node}^{TS} \cdot N_{end-node}^{MAX}}{N_{router}^{MAX}} \right\rfloor \cdot \frac{R_{TS}}{\left(\sum_{j=0}^{H_{sink}-1} (N_{router}^{MAX})^{H-j}\right) \cdot \left(N_{end-node}^{MAX} + \omega\right)} \tag{47}$$

for $\forall H_{sink}$, $1 \leq H_{sink} \leq H$.

The average arrival rate r_{data} of sensory data flow must be lower than r_{data}^{MAX} in any case. The value of burst b_{data} is selected according to the burstiness of sensory data.



(a) The test-bed deployment for $H_{sink} = 1$.



(b) The sensory data traffic.

Fig. 12. The test-bed deployment and sensory data traffic upper bounded by arrival curve α_{data}.

7. PERFORMANCE EVALUATION

In this section, we compare the analytical results based on Network Calculus that we proposed in this paper with the experimental results obtained through the use of IEEE 802.15.4/ZigBee technologies. The analytical results are computed using a MATLAB model [Jurcik 2008], and the experimental results are obtained using a test-bed based on the TelosB motes [Crossbow 2008].

7.1 Network Setup

The experimental test-bed (illustrated in Fig. 12a) consists of 7 clusters and 14 TelosB motes running the TinyOS 1.x [TinyOS 2008] operating system with open source implementation of the IEEE 802.15.4/ZigBee protocol stack [Cunha et al. 2007]. The TelosB is a battery powered wireless module with integrated sensors, IEEE 802.15.4 compliant radio, antenna, low-power 16-bit RISC microcontroller, and programming capability via USB. For debugging purposes, we have used the Chipcon CC2420 packet sniffer [Chipcon 2008] that provides a raw list of the transmitted packets, and the Daintree Sensor Network Analyzer (SNA) [Daintree Networks 2008] that provides additional functionalities, such as displaying the graphical topology of the network.

Note that, in practice, this experimental deployment could span over a wider region than the one illustrated in Fig. 12a, provided that every end-node and child router is within radio range of its parent router (TelosB radio range is around several tens meters). Number of end-nodes associated to each router can also be higher (not all nodes might need guaranteed bandwidth).

We configured the application running on the sensor nodes to generate 5 bytes at the data payload of every message. Hence, the maximum size of the MAC frame is equal to $MPDU_{max}$ = 208 bits (i.e. MAC header = 72 bits, FCS = 16 bits, network header = 64 bits, and data payload = 56 bits). Note that the maximum size of frame is then equal to 256 bits (i.e. $MPDU_{max}$ + physical header). The minimum size of frame is equal to 200 bits (i.e. physical header, MAC header, FCS and network header). Note that all devices in the WSN have unique 16-bit short addresses assigned by the PAN Coordinator during the association process.

TinyOS 1.x flushes the reception buffer of the radio transceiver after processing the first arriving frame. Thus, the frames that arrive during the processing time of the first frame are discarded. This problem has been already reported and fixed in TinyOS 2.x. Since our implementation of IEEE 802.15.4/ZigBee protocol stack was built over TinyOS 1.x, we overcame the aforementioned problem by setting the inter-frame spacing (IFS) time (i.e. time between two consecutive frames) such that no frame arrives during the frame processing time. A value of IFS equal to 3.07 ms was measured for any size of frame.

According to Eq. (43), the effective bandwidth guaranteed by one time slot in a superframe with $SO = 4$ is equal to 3.125 kbps with 100% duty-cycle. Hence, in our experimental scenario with a 12.5% duty-cycle (i.e. $BO = BO_{min} = 7$), the effective bandwidth guaranteed by one time slot in a given superframe is equal to $R_{TS} = 3.125 \cdot 0.125 = 0.390$ kbps. Let us assume $N_{end-node}^{TS} = 1$. Then according to Eqs. (46) and (47), we obtain the maximum arrival rates of the sensory data flow as follows:

— $r_{data}^{MAX} = 455$ bps for $H_{sink} = 2$

— $r_{data}^{MAX} = 683$ bps for $H_{sink} = 1$

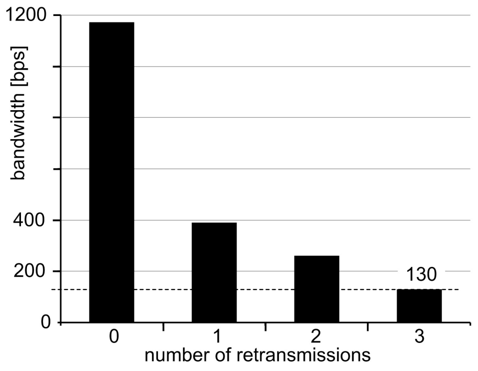
— $r_{data}^{MAX} = 911$ bps for $H_{sink} = 0$ (root)

As a result of $r_{data} \leq \min\left(r_{data}^{MAX}\right)$ and $r_{data} \leq R_{TS}$, we consider an average arrival rate equal to $r_{data} = 390$ bps, which corresponds to 3 frames (256-bit each) generated during one Beacon Interval (BI = 1.96608 sec). We assume that the burst tolerance is equal to $b_{data} = 576$ bits. Hence, each sensory data flow is bounded by arrival curve $\alpha_{data} = 576 + 390 \cdot t$. Note that Network Calculus based analytical model is bit-oriented, which means that sensory data are handled as a continuous bit stream with data rate r, while the experimental test-bed is frame-oriented, where data traffic is organized in frames of a given size. The frames can be generated at constant bit rate (CBR) or variable bit rate (VBR), but total data traffic must be upper bounded by the arrival curve α_{data} (Fig. 12b).

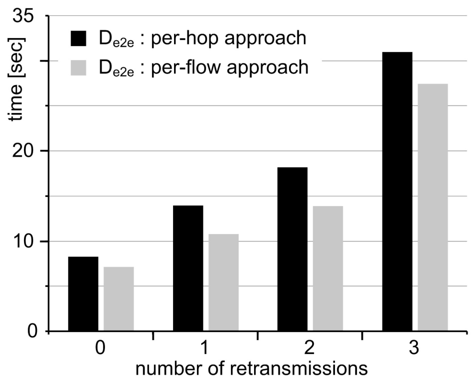
Finally, let us summarize the complete network setting:

$N_{router}^{MAX} = 2$	$r_{data} = 390$ bps
$N_{end-node}^{MAX} = 1$	$b_{data} = 576$ bits
$H = 2$	IFS = 3.07 ms
$SO = 4$ (SD = 245.76 ms)	$L_{CFP} = 15$
$BO = 7$ (BI = 1966.08 ms)	$\omega = 0$
$MPDU_{max} = 208$ bits	$macMaxFrameRetries = 0$

We assume the non-overlapping worst-case TDCS of a data flow along the longest routing path (i.e. from an end-node of router R_{24} to the sink in Fig. 3) given by the following sequence of Superframe Durations: SD_{11}, SD_{01}, SD_{12}, SD_{24}, SD_{23}, SD_{21}, SD_{22}. Note that we consider only unacknowledged transmissions.



(a) The guaranteed bandwidth of 1 time slot.



(b) The worst-case end-to-end delay.

Fig. 13. The worst-case delay and bandwidth as a function of the number of retransmissions.

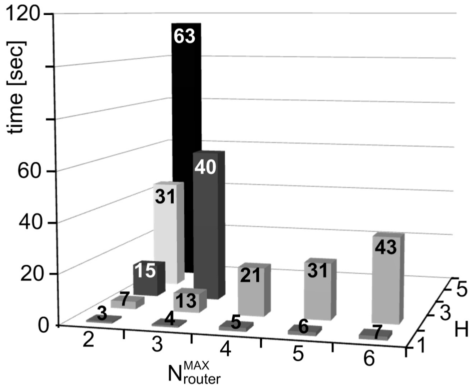
7.2 Analytical Evaluation

Number of Retransmissions vs. Timing Performance

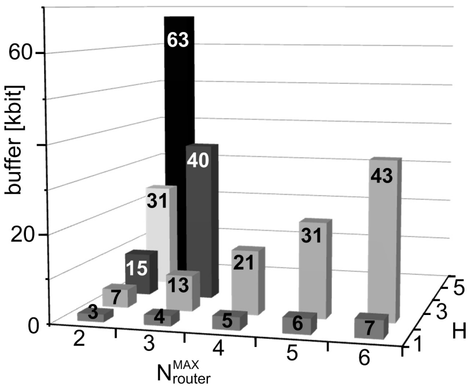
The unreliable and time-varying characteristics of wireless channels can be minimized using the acknowledgment and retransmission mechanisms. On the other side, each retransmission decreases guaranteed bandwidth and increases communication delay as depicted in Fig. 13. Figure 13a shows the guaranteed bandwidth of one time slot and Figure 13b the theoretical worst-case end-to-end delay as a function of the number of retransmissions (parameter $macMaxFrameRetries$) for $H_{sink} = 0$. The guaranteed bandwidth of one GTS time slot (Fig. 13a) is obtained using Eq. (43) multiplied by the duty-cycle, which is equal to 12.5%. It can be observed that the minimum guaranteed bandwidth of one time slot is equal to 130 bps when three retransmissions are enabled. To obtain comparable end-to-end delays, the same number of time slots must be allocated to each node when consider different number of retransmissions. Hence, the average arrival rate must be reduced to $r_{data} = 40$ bps. According to the IEEE 802.15.4 standard, the inter-frame spacing IFS is equal to LIFS or SIFS depending on the length of MAC frame. The other network settings are the same as mentioned in Section 7.1. The worst-case end-to-end delays obtained by per-flow approach introduces less pesimism than the per-hop approach, and end-to-end delays increase with the number of retransnmissions as shown in Fig. 13b. These results confirm our previous assumptions. Each retransmission enlarges the end-to-end delay by 58% on average, but also increases the reliability of data transmission.

Network Planning

Our methodology can be used for the planning of the cluster-tree topology as well. Let us consider the example of a convergecast application gathering sensory data at the root (i.e. $H_{sink} = 0$) and using the network settings as mentioned in Section 7.1. However, in this case, the largest feasible configuration of the worst-case cluster-tree topology is achieved for $N_{router}^{MAX} = 2$ and $H = 2$. This means that a feasible



(a) The worst-case end-to-end delay.



(b) The buffer requirement of the sink router.

Fig. 14. The worst-case delay and buffer requirement as a function of N_{router}^{MAX} and H.

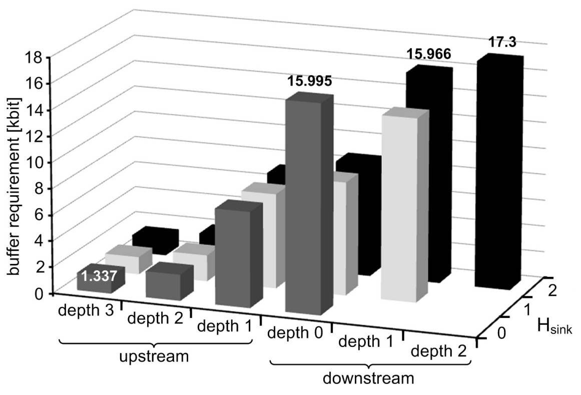
worst-case cluster-tree topology given by the parameters N_{router}^{MAX} and H satisfies the network constraints given by the other parameters, namely r_{data}, b_{data}, SO, BO, $MPDU_{max}$, IFS, L_{CFP}, ω, $macMaxFrameRetries$ and $N_{end-node}^{MAX}$.

To obtain more illustrative results, we reduce the length of the IFS to the minimum value defined by 802.15.4 standard (see Section 6.2), $r_{data} = 25$ bps, $SO = 2$, $L_{CFP} = 14$, and keep the other settings. Beacon Order BO is equal to the minimum value according to Eq. (45). Figure 14a presents the worst-case end-to-end delay and Figure 14b buffer requirement of the sink router as a function of the height of the tree H and the maximum number of child routers N_{router}^{MAX}. In other words, Figure 14 presents all feasible configurations of the worst-case cluster-tree topology, which satisfy a given network constraints. The numerical values at the columns represent the total number of routers (N_{router}^{TOTAL} Eq. (36)) in the network. It can be observed that there can be more feasible configurations for the same number of routers. For instance, the total number of 31 routers can be achieved with two configurations, namely $H = 2$ and $N_{router}^{MAX} = 5$ or $H = 4$ and $N_{router}^{MAX} = 2$. The buffer requirements at the sink router are almost the same for both configurations (22 kbits and 24.1 kbits, respectively), but the first configuration provides around half of the worst-case delay ($D_{e2e} = 22.76$ sec) compared with the second configuration ($D_{e2e} = 44.56$ sec). On the other side, the cluster-topology using the second configuration can spread out over a larger area due to the higher height H. So the network designer must find a trade-off for a given application-specific implementation.

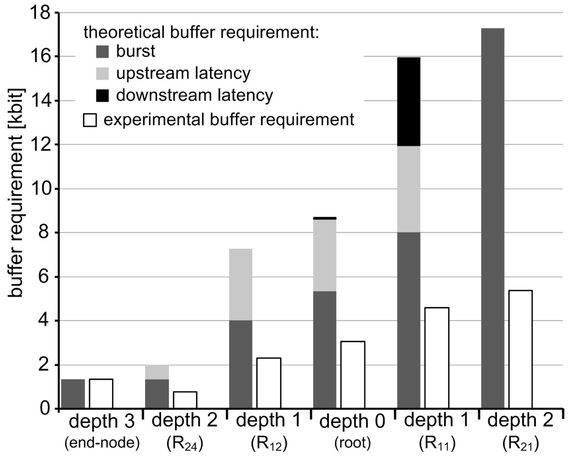
7.3 Experimental Evaluation

Buffer Requirements

Figure 15a presents the theoretical worst-case buffer requirement of the routers at given depths and as a function of the sink position. It can be observed that end-nodes have the smallest buffer requirement as they are the leaves of the tree, and that the buffer requirement grows in the direction of the sink router. Since



(a) The worst-case buffer requirement per router as a function of the depth and sink position.



(b) The theoretical vs. experimental buffer requirements.

Fig. 15. The worst-case buffer requirement.

the sink can be associate to any router in a WSN and in order to avoid buffer overflow, all routers at depth i should allocate a buffer of capacity equal at least to the maximum buffer requirement at a given depth i (e.g. all routers at depth 1 allocate a buffer of capacity equal to 15.966 kbits), which effectively demonstrates how these analytical results can be used by a network designer. Figure 15b shows the theoretical worst-case buffer requirements compared with the maximum values obtained through real experimentation, for $H_{sink} = 2$.

First, the theoretical buffer requirements are divided into three portions according their origin, as we have shown in Section 5.1. Observe that the cumulative effect of the burst is more important than the cumulative effect of the service latencies. The effect of the service latencies may be more important for other settings of b_{data} and r_{data}. So, the different settings of the sensory arrival curve affect the buffer requirements. The minor effect of the upstream service latency at depth 0

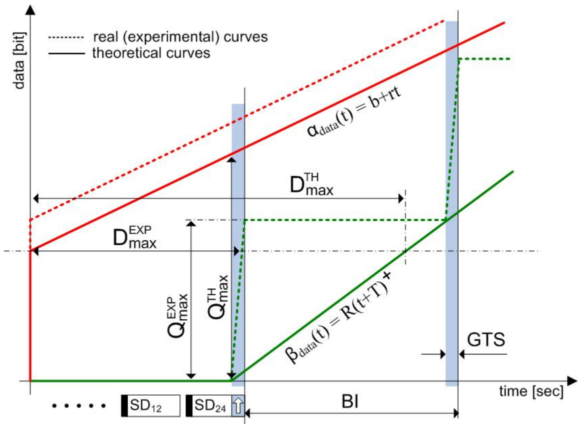


Fig. 16. Theoretical vs. experimental data traffic (related to Fig. 2).

is given by the priority rules (Section 3.3), such that the data arriving during the transmit GTS (i.e. upstream flow) are stored in the root until the receive GTS (i.e. downstream flow), at the end of the same SD, is active and data is dispatched (Fig. 11).

The next observation confirms that the theoretical values upper bound the experimental values. The pessimism of the theoretical bounds is justified by the fact that the Network Calculus analytical model is based on a continuous approach (arrival and service curves are continuous) in contrast to the real stepwise behavior of flows and services (in the test-bed). In practice, the data is actually transmitted only during its GTS, while in the analytical model we consider a continuous data flow during the whole BI, since it represents the average rate and not the instantaneous rate. Figure 16 illustrates the problem and shows the arrival and service curves of a data flow sent by an end-node to its parent router. The burst of the outgoing data flow b^*_{data} (Eq. (11)) is equal to Q_{max}^{TH}, in case of the analytical model, or Q_{max}^{EXP}, in the experimental case. Due to the cumulative flow effect, the differences between theoretical (Q_{max}^{TH}) and experimental (Q_{max}^{EXP}) values of buffer requirement grow with depth. The rate-latency service curve used in our analysis results from a trade-off between computing complexity and pessimism.

The numerical values of theoretical worst-case as well as experimental maximum buffer requirements are summarized in Table II. The bandwidth requirements given by Eqs. (34) and (35), and the corresponding number of time slots are also presented. In Tables II and III, U means an upstream router at depth i or an upstream link to a router at depth i, and D means a downstream router or a downstream link from a router at depth i.

Delay Bounds

In Figure 17, we compare the worst-case, maximum and average values of per-hop delays bound in each router, and the end-to-end delay bounds for $H_{sink} = 2$. A first observation confirms that theoretical results upper bound the experimental results. The difference in theoretical worst case (D_{max}^{TH}) and experimental maximum

	depth		theoretical results (worst-case values)			experimental results (maximum values)
			R_i [kbps]	N_i^{TS}	Q_i [kbit]	Q_i [kbit]
$H_{sink} = 0$	0	U	1.7	3	**15.995**	5.376
	1	U	0.39	1	7.329	2.304
	2	U	–	–	2.008	0.768
$H_{sink} = 1$	0	D	1.56	4	8.667	3.072
		U	1.17	3	–	–
	1	D	–	–	14.02	5.376
		U	0.39	1	7.257	2.304
	2	U	–	–	2.008	0.768
$H_{sink} = 2$	0	D	1.56	4	8.667	3.072
		U	1.17	3	–	–
	1	D	2.34	6	**15.966**	4.608
		U	0.39	1	7.257	2.304
	2	D	–	–	**17.3**	5.376
		U	–	–	2.008	0.768
end-node			0.39	1	**1.344**	1.337

Table II. Buffer requirements: theoretical vs. experimental results.

(D_{max}^{EXP}) delays (Fig. 16) is given by the aforementioned continuous and stepwise behaviors of the analytical model and test-bed, respectively. The experimental delays comprise mainly the service latencies (Fig. 16) decreasing in the direction of the sink (Fig. 11). Hence, the maximum per-hop delays also decrease in the direction of the sink, as can be observed in Fig. 17. The reduced downstream delay at depth 0 results from the priority rule (Section 3.3). The end-to-end delays bounds are quite high, even though the b_{data} and r_{data} are low. This is mainly due to high value of $SO = 4$ (i.e. BI = 1.966 sec). Hence, the end-to-end delay bounds can be reduced using lower values of SO or higher bandwidth guarantees, using lower IFS, for example. Observe also that the worst-case end-to-end delay obtained by the per-flow approach introduces less pessimism than the delay from the per-hop approach (roughly 50% smaller: 27.13 s $\rightarrow$ 13.65 s, as presented in Table III).

Table III presents the worst-case, maximum and average numerical values of per-hop and per-flow delay bounds, and the end-to-end delays for different sink positions. Note that the average values were computed from a set of 15 runs, involving the transmission of 1155 frames each. The theoretical worst-case end-to-end delays are obtained as the sum of per-hop delays using Eq. (39), or by per-flow approach, which results in the family of service curves as a function of $\Theta \geq 0$. In our analysis we assume $\Theta = T + (b_2/R)$ as a trade-off between computation complexity and optimality. The determination of the optimal service curve, leading to the lowest worst-case delay, will be addressed in future work.

Duty-cycle vs. Timing Performance

In Section 6.4, we mentioned that to maximize the lifetime of a WSN, low duty-cycles are required. On the other hand, low duty-cycles enlarge the timing performance of a WSN. Our assumptions were confirmed as depicted in Fig. 18, which shows the theoretical worst-case and experimental maximum end-to-end delays as a

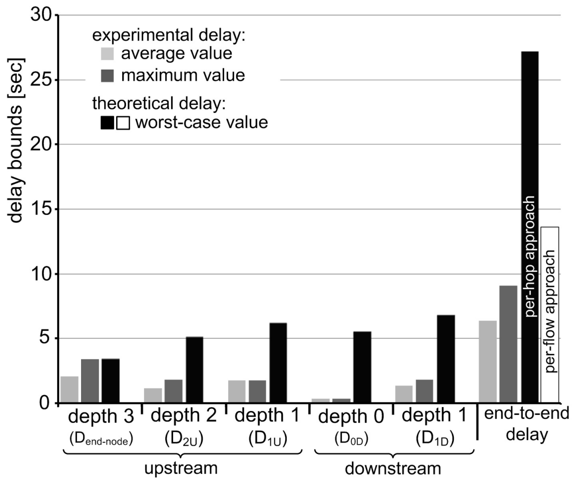


Fig. 17. The theoretical vs. experimental delay bounds.

	depth	theoretical results (worst-case values)	experimental results maximum	experimental results average
		D_i [sec]	D_i [sec]	D_i [sec]
$H_{sink} = 0$	1 U	6.257	1.764	1.308
	2 U	5.143	1.812	1.602
	D_{e2e}	14.82/**9.69**	**7.154**	4.952
$H_{sink} = 1$	0 D	5.547	0.104	0.099
	1 U	6.195	1.76	1.728
	2 U	5.143	1.809	1.602
	D_{e2e}	20.31/**10.53**	**7.251**	5.471
$H_{sink} = 2$	0 D	5.547	0.104	0.099
	1 D	6.814	1.812	1.321
	1 U	6.195	1.766	1.728
	2 U	5.143	1.814	1.135
	D_{e2e}	27.13/**13.65**	**9.074**	6.325
end-node D_{data}		3.425	3.402	2.042

Table III. Delay bounds: theoretical vs. experimental results.

function of duty-cycle for $H_{sink} = 0$. The value of SO is set to 4 and the decreasing duty-cycles are obtained by increasing BO. Note that for $SO = 4$, the minimum BO is equal to 7. To avoid the lack of bandwidth for smaller duty-cycles, the average arrival rate must be reduce to $r_{data} = 0.190$ kbps (note that $r_{data}^{MAX} = 0.195$ kbps for the smallest duty-cycle equal to 3.125%). The other network settings are the same as in previous experiments. The theoretical worst-case end-to-end delays are obtained by per-hop and per-flow approaches (Section 5.2). The observation again confirms that the theoretical results upper bound the experimental results, and the worst-case delay obtained by the per-flow approach offers less pessimism than the delay from the per-hop approach.

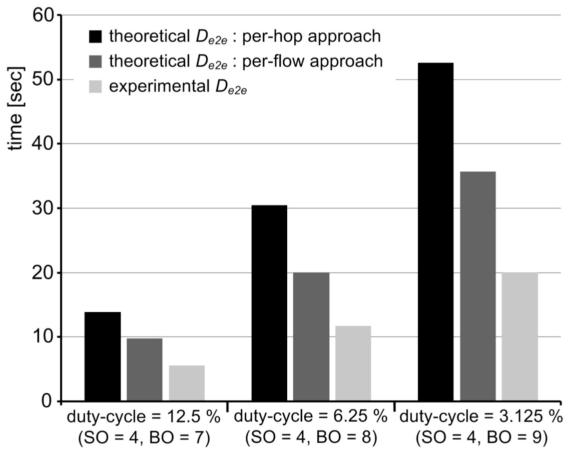


Fig. 18. The theoretical worst-case and experimental maximum end-to-end delays as a function of duty-cycle for $H_{sink} = 0$.

8. DISCUSSION ON MOBILITY SUPPORT

This section outlines several issues related to how to support the sink mobility in a cluster-tree WSN.

We assume that all network nodes (i.e. routers and end-nodes) are static; note that even the sink router is static - only the sink entity can be mobile. Generally, the physical topology of a WSN consists of the wireless links between every pair of nodes that are within transmission range of each other. Two nodes are within transmission range if they can inter-communicate bidirectionally without using any intermediate node. On the other hand, a logical topology is associated to a given physical topology and defines the active wireless links between nodes.

Considering the mobile sink behavior, the logical topology has to be adapted to allow sensory data to reach the new sink location. We distinguish between two approaches: *logical topology update* (Fig. 19a), applicable to the cases where the sink moves more frequently (i.e. in a continuous fashion), and *logical topology rebuilding* (Fig. 19b), applicable to the cases where the sink changes location less frequently (sporadically). Generally, the logical topology update approach should be chosen when the network inaccessibility time resulting from the logical topology rebuilding is greater than the maximum time between two consecutive sink movements (i.e. two consecutive sink associations with different sink routers).

8.1 The Logical Topology Update

When the sink moves very frequently, it might be more adequate to keep the logical network topology unchanged (the same original root and network deployment). Hence, the path of data flows in the direction of the sink has to be updated accordingly (dubbed *logical topology update*). The length of this updated path is equal to the number of hops between the previous and current sink router positions. In the worst-case, when the sink moves between two farthest routers of different sub-trees of the root, the length of the update path becomes twice the height of the cluster-

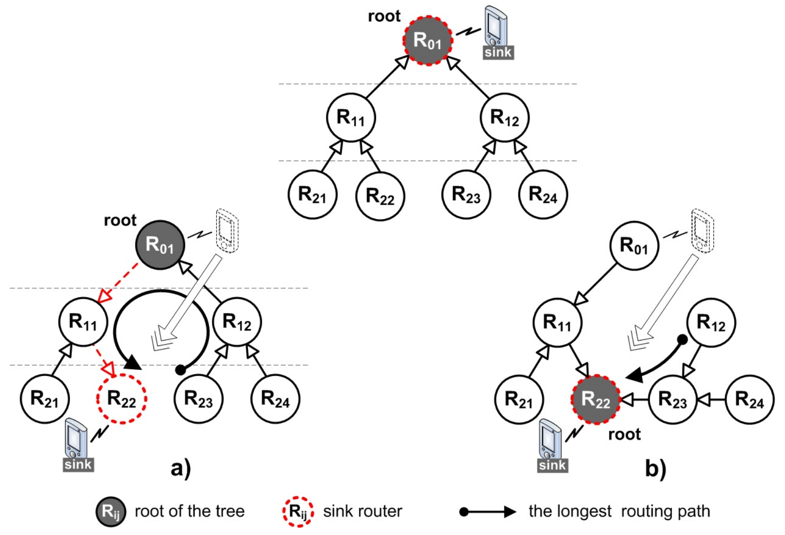


Fig. 19. Two approaches to mobile sink behavior: a) logical topology update; b) logical topology rebuilding.

tree. Thus, the network inaccessibility time will depend on the length of the update path and on the TDCS. In case of the worst-case TDCS, the inaccessibility time is expressed as the duration of the WC-TDCS's cycle multiplied by the length of the update path. This inaccessibility time is much smaller than the network inaccessibility time resulting from the logical topology rebuilding (see next), and also the energy consumption during the logical topology update is expected to be lower. On the other hand, the resource requirements (i.e. the resources guaranteed along the longest data path) in the updated logical topology are higher as compared to the rebuilt logical topology. It results in a balanced logical topology with unbalanced load requiring higher energy consumption and higher end-to-end delay bounds.

8.2 The Logical Topology Rebuilding

When the sink moves infrequently, it might be more adequate to rebuild the logical topology from scratch, according to the current position of the sink (dubbed *logical topology rebuilding*). The current sink router becomes the root of the tree and initiates the logical topology rebuilding of the cluster-tree WSN. The rebuilt cluster-tree topology may keep the same worst-case parameters or exceed some of them (e.g. the height of the tree). There can be more than one feasible cluster-tree topology. Consequently, this topology rebuilding procedure introduces higher network inaccessibility times and consumes more energy as compare to the previous approach. On the other hand, the run time resource requirements in this approach are lower as compared to the updated logical topology, resulting in reduced delay and buffer bounds. It results in a balanced logical topology with balanced load requiring lower energy consumption and lower end-to-end delay bounds.

9. CONCLUSIONS

This paper shows how to support time-bounded communications in cluster-tree Wireless Sensor Networks (WSNs). We tackled the worst-case analysis and dimensioning of cluster-tree WSNs assuming that the data sink can be static or mobile, i.e. can be associated to any router in the WSN. We proposed the worst-case system model, an analytical methodology (closed-form recurrent expressions) and a software tool that enable network designers to analyze and dimension these networks. In this way, it is possible to guarantee the routers' buffer size to avoid buffer overflows and to minimize clusters' duty-cycle (maximizing nodes' lifetime) still satisfying that given messages' deadlines are met.

We also showed how to instantiate our generic methodology in IEEE 802.15.4/ZigBee, which are promising technologies for WSN applications. Finally, we developed a multiple cluster test-bed based on Commercial-Off-The-Shelf technologies, namely TelosB motes [Crossbow 2008] running open-ZB protocol stack [Cunha et al. 2007] over TinyOS [TinyOS 2008]. This test-bed enabled us to assess the validity and pessimism of our worst-case theoretical results (buffer requirements and message end-to-end delays), by comparing these to the maximum and average values measured in the experiments.

Ongoing and future work includes improving the current methodology to encompass clusters operating at different duty-cycles and to provide a model that enables real-time control actions, i.e. the sink assuming the role of controlling sensor/actuator nodes.

APPENDIX

A. TABLE OF SYMBOLS

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

Symbol	Definition
$R(t)$	input cumulative function
$R^*(t)$	output cumulative function
$\alpha(t)$	arrival curve
$\beta(t)$	guaranteed service curve
$\alpha^*(t)$	output bound constraining the output function $R^*(t)$
D_{max}	delay bound
Q_{max}	backlog bound
$\odot$	min-plus deconvolution
$\otimes$	min-plus convolution
$\beta_1^{eq}(t,\Theta)$	equivalent service curve for flow 1
$1_{\{expr\}}$	$1_{\{expr\}}$ is equal to 1 if *expr* is true, and 0 otherwise.
$(x)^+$	$(x)^+ = \max(0,x)$
$\alpha_{b,r}(t)$	affine arrival curve with rate r and burst size b

$\beta_{R,T}(t)$	rate-latency service curve with rate R and latency T
H	height of the tree
$N_{end-node}^{MAX}$	maximum number of child end-nodes
N_{router}^{MAX}	maximum number of child routers
H_{sink}	maximum depth of the sink router
$\beta_{end-node}(t)$	rate-latency service curve guaranteed to end-nodes
$\alpha_{data}(t)$	affine arrival curve constraining sensory data
$\bar{\alpha}_i(t)$	affine arrival curve constraining the input function $R(t)$ of a router at a depth i
$\alpha_i^*(t)$	output bound constraining the output function $R^*(t)$ of a router at a depth i
$\beta_i(t)$	rate-latency service curve guaranteed by a router at depth i
ω	binary variable which is equal to 1 if routers have sensing capabilities; otherwise ω is equal to 0
Q_{iU}	the required buffer size of an upstream router at a depth i
D_{iU}	delay bound between a child router at depth i and its parent router at depth $i-1$
D_{iD}	delay bound between a parent router at depth i and its child router at depth $i+1$
D_{e2e}	the worst-case end-to-end delay

Table IV: Table of symbols.

REFERENCES

Abdelzaher, T., Prabh, S., and Kiran, R. 2004. On real-time capacity limits of multihop wireless sensor network. In *Proceedings of the 25th IEEE International Real-Time Systems Symposium (RTSS)*. IEEE Computer Society Press, Washington, DC, USA, 359–370.

Bai, H. and Atiquzzaman, M. 2003. Error modeling schemes for fading channels in wireless communications: A survey. *IEEE Communications Surveys and Tutorials 5,* 2 (Oct.), 2–9.

Boudec, J. L. and Thiran, P. 2004. *Network Calculus: A Theory of Deterministic Queuing Systems for the Internet Lecture Notes in Computer Science).* Springer-Verlag, New York, USA.

Caccamo, M., Zhang, L. Y., Sha, L., and Buttazzo, G. 2002. An implicit prioritized access protocol for wireless sensor networks. In *Proceedings of the 23rd IEEE Real-Time Systems Symposium (RTSS)*. IEEE Computer Society Press, Washington, DC, USA, 39–48.

Chipara, O., He, Z., Xing, G., Chen, Q., Wang, X., Lu, C., Stankovic, J., and Abdelzaher, T. 2006. Real-time power-aware routing in sensor networks. In *Proceedings of the 14th IEEE International Workshop on Quality of Service (IWQoS)*. IEEE Computer Society Press, Washington, DC, USA, 83–92.

Chipcon. 2008. C2420DK development kit datasheet. [Online]. Available: http://www.ti.com.

Crenshaw, T., Hoke, S., Tirumala, A., and Caccamo, M. 2007. Robust implicit EDF: A wireless mac protocol for collaborative real-time systems. *ACM Trans. on Embedded Computing Systems 6,* 4 (Sept.), 28.

Crossbow. 2008. TelosB mote datasheet. [Online]. Available: http://www.xbow.com.

Cunha, A., Koubaa, A., Severino, R., and Alves, M. 2007. Open-ZB: an open source implementation of the IEEE 802.15.4/ZigBee protocol stack on TinyOS. In *Proceedings of the 4th IEEE International Conf. on Mobile Ad-hoc and Sensor Systems (MASS).*

Cunha, A., Severino, R., Pereira, N., Koubaa, A., and Alves, M. 2008. ZigBee over TinyOS: implementation and experimental challenges. In *Proceedings of the 8th Portuguese Conf. on Automatic Control (CONTROLO).* UTAD, Portugal, 911–916.

Daintree Networks. 2008. Daintree sensor network analyzer (SNA). [Online]. Available: http://www.daintree.net.

Diestel, R. 2000. *Graph Theory.* Springer-Verlag, New York, USA.

Facchinetti, T., Almeida, L., Buttazzo, G. C., and Marchini, C. 2004. Real-time resource reservation protocol for wireless mobile ad hoc networks. In *Proceedings of the 25th IEEE International Real-Time Systems Symposium (RTSS).* IEEE Computer Society Press, Washington, DC, USA, 382–391.

Gandham, S., Dawande, M., Prahash, R., and Venkatesan, S. 2003. Energy efficient schemes for wireless sensor networks with multiple mobile base stations. In *Proceedings of the 46th IEEE Global Communications Conf. (GLOBECOM).* IEEE Computer Society Press, Washington, DC, USA, 377–381.

Gibson, J., Xie, G., and Xiao, Y. 2007. Performance limits of fair-access in sensor networks with linear and selected grid topologies. In *Proceedings of the 50th IEEE Global Communications Conf. (GLOBECOM).* IEEE Computer Society Press, Washington, DC, USA, 688–693.

He, T., Stankovic, J. A., Lu, C., and Abdelzaher, T. F. 2005. A spatiotemporal communication protocol for wireless sensor networks. *IEEE Trans. Parall. Distrib. Syst. 16,* 10 (Oct.), 995–1006.

Hu, Z. and Li, B. 2004. *to appear in Ad Hoc and Sensor Networks, Yang Xian and Yi Pan, Editors.* Nova Science Publishers, New York, USA.

IEEE-TG15.4. 2006. *Part 15.4: wireless medium access control (MAC) and physical layer (PHY) specifications for low-rate wireless personal area networks (LR-WPANs).* IEEE Computer Society.

In-Stat. 2009. Automatic Meter Reading (AMR) and smart energy to be the winning application for 802.15.4/ZigBee. [Online]. Available: http://www.instat.com/.

Jurcik, P. 2008. Matlab tool for the worst-case dimensioning of IEEE 802.15.4/ZigBee cluster-tree WSNs. [Online]. Available: http://www.open-zb.net/downloads.php.

Jurcik, P., Severino, R., Koubaa, A., Alves, M., and Tovar, E. 2008. Real-time communications over cluster-tree sensor networks with mobile sink behaviour. In *Proceedings of the 14th IEEE International Conf. on Embedded and Real-Time Computing Systems and Applications (RTCSA).* IEEE Computer Society Press, Washington, DC, USA, 401–412.

Koubaa, A., Alves, M., and Tovar, E. 2006a. Modeling and worst-case dimensioning of cluster-tree wireless sensor networks. In *Proceedings of the 27th Real Time Systems Symposium (RTSS).* IEEE Computer Society Press, Washington, DC, USA, 412–421.

Koubaa, A., Alves, M., and Tovar, E. 2006b. Modeling and worst-case dimensioning of cluster-tree wireless sensor networks: proofs and computation details. Tech. Rep. TR-060601, CISTER-ISEP research unit, Porto, Portugal.

Koubaa, A., Alves, M., Tovar, E., and A.Cunha. 2008. An implicit GTS allocation mechanism in IEEE 802.15.4 for time-sensitive wireless sensor networks: theory and practice. *Real-Time Systems Journal 39,* 1–3 (Aug.), 169–204.

Koubaa, A., Cunha, A., and Alves, M. 2007. A time division beacon scheduling mechanism for IEEE 802.15.4/ZigBee cluster-tree wireless sensor networks. In *Proceedings of the 19th Euromicro Conf. on Real-Time Systems (ECRTS).* IEEE Computer Society Press, Washington, DC, USA, 125–135.

Koubaa, A. and Song, Y. 2004. Evaluation and improvement of response time bounds for real-time applications under non-pre-emptive fixed priority scheduling. *Int. J. of Production Research 42,* 14 (July), 2899–2913.

Lee, C., Ekici, E., and Felemban, E. 2006. MMSPEED: Multipath multi-SPEED protocol for QoS guarantee of reliability and timeliness in wireless sensor networks. *IEEE Transactions on Mobile Computing 5,* 6 (June), 738–754.

Lenzini, L., Martorini, L., Mingozzi, E., and Stea, G. 2006. Tight end-to-end per-flow delay bounds in FIFO multiplexing sink-tree networks. *Performance Evaluation 63,* 9 (Oct.), 956–987.

Poe, W. and Schmitt, J. 2007. Minimizing the maximum delay in wireless sensor networks by intelligent sink placement. Tech. Rep. 362/07, University of Kaiserslautern, Germany.

Prabh, S. 2007. Real-time wireless sensor networks. Ph.D. thesis, Department of Computer Science, University of Virginia, VA, USA.

Prabh, S. and Abdelzaher, T. 2007. On scheduling and real-time capacity of hexagonal wireless sensor networks. In *Proceedings of the 19th Euromicro Conf. on Real-Time Systems (ECRTS).* IEEE Computer Society Press, Washington, DC, USA, 136–145.

Raman, B. and Chebrolu, K. 2008. Censor networks: a critique of "sensor networks" from a systems perspective. *ACM SIGCOMM Computer Communication Review 38,* 3 (July), 75–78.

Schmitt, J. and Roedig, U. 2005a. Sensor network calculus - a framework for worst case analysis. In *Proceedings of the 1st IEEE/ACM Conf. on Distributed Computing in Sensor Systems (DCOSS).* IEEE Computer Society Press, Washington, DC, USA, 141–154.

Schmitt, J. and Roedig, U. 2005b. Worst case dimensioning of wireless sensor networks under uncertain topologies. In *Proceedings of the 1st Workshop on Resource Allocation in Wireless NETworks (RAWNET).* IEEE Computer Society Press, Washington, DC, USA.

Schmitt, J., Zdarsky, F., and Thiele, L. 2007. A comprehensive worst-case calculus for wireless sensor networks with in-network processing. In *Proceedings of the 28th IEEE Real-Time Systems Symposium (RTSS).* IEEE Computer Society Press, Washington, DC, USA, 193–202.

Stankovic, J., Abdelzaher, T., Lu, C., Sha, L., and Hou, J. 2003. Real-time communication and coordination in embedded sensor networks. *Proceedings of the IEEE 91,* 7 (July), 1002–1022.

Stankovic, J., Lee, I., Mok, A., and Rajkumar, R. 2005. Opportunities and obligations for physical computing systems. *IEEE Computer 38,* 11 (Nov.), 25–33.

TinyOS. 2008. TinyOS open-source OS for wireless embedded sensor networks. [Online]. Available: http://www.tinyos.net.

Trdlicka, J., Johansson, M., and Hanzalek, Z. 2007. Optimal flow routing in multi-hop sensor networks with real-time constraints through linear programming. In *Proceedings of the 12th IEEE International Conf. on Emerging Technologies and Factory Automation (ETFA).* IEEE Computer Society Press, Washington, DC, USA, 924–931.

ZigBee. 2005. *ZigBee Specification, Version 1.0.* ZigBee Standards Organization.

Received Month Year; revised Month Year; accepted Month Year