Validating the Sensor Network Calculus by Simulations

Utz Roedig[1], Nicos Gollan[2], Jens B. Schmitt[2]

[1]InfoLab21, Lancaster University, Lancaster, UK
[2]disco | Distributed Computer Systems Lab, University of Kaiserslautern, Kaiserslautern, Germany

ABSTRACT

Network Calculus has been proposed and customized as a framework for worst-case analysis in wireless sensor networks (WSNs). It has been demonstrated that this so-called Sensor Network Calculus (SNC) is an effective network dimensioning tool as it allows us to calculate maximum message transfer delays and communication related energy consumption patterns *before* network deployment. So far it is unclear how the SNC calculated worst-case delay bounds compare to values experienced in real deployments. Our experiments presented in this paper show that an SNC worst-case delay prediction can be as little as 2.7% above the measured worst-case delay in a typical application scenario. Thus, it can be concluded that the SNC has a very practical relevance for dimensioning wireless sensor networks.

Keywords

Network Calculus, Sensor Network Calculus (SNC), Performance, Wireless Sensor Networks (WSN).

1. INTRODUCTION

Future application areas of wireless sensor networks (WSNs) might include industrial process automation, aircraft control systems or patient monitoring in hospitals. Such applications require predictable network performance in terms of message transfer delay. Timely analysis of sensor data must always be possible to decide if an action has to be performed. To achieve this goal, wireless sensor networks must be dimensioned properly. As over-provisioning is not a valid option in resource constrained wireless sensor networks a precise tool for network dimensioning is desirable.

In [8] a customization of the well-known Network Calculus is proposed as a dimensioning tool for wireless sensor networks. The so-called Sensor Network Calculus (SNC) can be used to dimension a sensor network such that message transfer delays are guaranteed to be *always* below an upper

worst-case bound[1]. Dimensioning implies in this context that specifications for the network structure, network traffic and node forwarding capabilities are determined. Indirectly, the power consumption patterns of nodes are specified as well since they depend on the nodes' forwarding capabilities. Thus, in summary, the SNC can be regarded as a potential candidate as WSN dimensioning tool.

Before the SNC can be considered for practical WSN dimensioning an important question must be addressed: *How realistic is the worst-case delay bound calculated using the SNC?* If message transfer delays in real networks are in most cases far lower than the worst-case upper bound the network was dimensioned for, a waste of resources occurs. In fact, an over-provisioned network would be the result; a situation that the use of the SNC should have prevented in the first place.

In this paper it is investigated how the SNC calculated worst-case message transfer delay compares to the message transfer delay measured in a simulated network deployment. Realistic WSN application scenarios are investigated to determine the usefulness of the SNC as a dimensioning tool.

2. RELATED WORK

While there has been a growing body of work on real-time aspects in WSNs (see for example [14] for a review), holistic system models on the interplay between performance characteristics and energy-efficiency targets as they are required for dimensioning purposes have not been addressed much. One notable exception is the work by Chiasserini and Garetto [3], which proposes a performance model based on Markov chains. The model relates performance characteristics such as data delivery delay and energy management parameters. However, for applications with stricter timing requirements such as average case analysis methodology is of limited use. What is needed for such WSNs is a analysis methodology like the SNC. In [8], the SNC framework for the analysis of WSNs has been proposed based on the network calculus [2]. It allows us to relate energy management parameters with performance characteristics under assumptions. In [9], this framework is extended to accommodate random topologies

---

[1]Besides the fact that data has to reach the destination in time it has to be ensured as well that sufficient data reaches the destination. A network dimensioning regarding message transfer reliability is not yet possible within the SNC framework and additional, complementary methods have to be applied. However, the SNC calculated delay bounds are still valid in a lossy, re-transmission free environment as lost packets will reduce traffic and always decrease the incurred delay.

for which certain topological parameters as, for example the maximum path length are known. Another extension of the SNC for WSNs with multiple sinks is presented in [13]. In [7], a methodology to analyze 802.15.4 cluster-tree WSNs has been proposed, again based on network calculus. Based on scheduling theory, [1] derives worst-case bounds on the capacity of real-time traffic that can be carried by a given WSN. None of the aforementioned studies have made an attempt to validate the respective analytical bounds yet.

3. SENSOR NETWORK CALCULUS

The sensor network calculus (SNC) [8] enables a worst-case analysis of data transport delays taking into account the various inter-dependencies between the sensor nodes' forwarding capabilities, the network traffic and the network topology. In particular, a node's forwarding capability correlates with its power consumption.

3.1 Basic Sensor Network Calculus

To apply the SNC, the network topology has to be known. For example a tree-structured network topology with a sink at the root and n_i sensor nodes can be used.

Next, the network traffic has to be described in terms of so-called arrival curves for each node. An arrival curve defines an upper bound for the input traffic of a node. Leaf nodes in the network have to handle traffic according to the sensing function they perform; for example, a node might sense an event and create a data packet at a maximum rate of one packet every second. This sensing pattern can be expressed as an arrival curve α_i. Non-leaf nodes handle traffic according to their own sensing pattern α_i and the traffic they receive from other nodes $j = 1, ..., n_i$. Thus, the arrival curve for the total input function $\bar{\alpha}_i$ for sensor node i is (according to [2]):

$$\bar{\alpha}_i = \alpha_i + \sum_{j=1}^{n_i} \alpha^*_{child(i,j)} \qquad (1)$$

The output of sensor node i, i.e. the traffic which it forwards to its parent in the tree, can be calculated as:

$$\alpha^*_i = \bar{\alpha}_i \oslash \beta_i = \left(\alpha_i + \sum_{j=1}^{n_i} \alpha^*_{child(i,j)} \right) \oslash \beta_i \qquad (2)$$

To calculate the output the so-called service curve β_i is used. The service curve specifies the worst-case forwarding capabilities of a node in terms of how long it takes to forward data to the next node; for example, a TDMA protocol might guarantee to forward data packets within $100ms$. The necessary forwarding delays are defined by the nodes forwarding characteristics which incorporate their power consumption patterns. The ratio of active to sleep phases of the transceiver defines the nodes possible forwarding speed *and* communication-related energy consumption. The ratio of sleep/active phases is often called *duty cycle*.

After specifying arrival and service curves, all flows $\bar{\alpha}_i$ (and α^*_i) have to be calculated using (1) and (2). For this calculation Algorithm 1 can be used. Thereafter, the local per-node delay bounds D_i for each sensor node i can be calculated according to a basic network calculus result given in [2]:

$$D_i = h(\bar{\alpha}_i, \beta_i) = \sup_{s \geq 0} \{ \inf\{ \tau \geq 0 : \bar{\alpha}_i(s) \leq \beta_i(s + \tau) \} \} \quad (3)$$

To compute the total information transfer delay $\bar{D}_i$ for a given sensor node i the per node delay bounds on the path $P(i)$ to the sink need to be added:

$$\bar{D}_i = \sum_{j \in P(i)} D_j \qquad (4)$$

The maximum information transfer delay in the sensor network can then obviously be calculated as $D = \max_{i=1,...,N} \bar{D}_i$. The whole procedure is called *total flow analysis* (TFA) because the traffic of all nodes is treated in an aggregate fashion. Please refer to [8] for further details on how to calculate performance bounds using the TFA.

Algorithm 1 Calculating the internal flows of a network.

1. Let us assume that arrival curves for the sensed input α_i and service curves β_i for sensor node i, $i = 1, \ldots, n$, are given.

2. For all leaf nodes the output bound α^*_i can be calculated as $\alpha^*_i = \alpha_i \oslash \beta_i$. Each leaf node is now marked as "calculated".

3. For all nodes only having children which are marked "calculated" the output bound α^*_i can be calculated according to (2) and they can again be marked "calculated".

4. If node 1 is marked "calculated" the algorithm terminates, otherwise go to step 3.

3.2 Advanced Sensor Network Calculus

While the TFA is a straightforward method to apply network calculus in the domain of wireless sensor networks, there is room for improvement with respect to the quality of the performance bounds which are calculated. In particular, we can use the result for the so-called Pay Multiplexing Only Once analysis (PMOO) described in [12], to compute an end-to-end service curve for a specific flow of interest from one sensor node to the sink. Due to the sink-tree structure of the network, all flows that join a flow of interest remain multiplexed until the sink, making it possible to calculate the total information transfer delay $\bar{D}_i$ for a given sensor node i by using a flow-specific end-to-end service curve. We call this method PMOO analysis (PMOOA) further on. It can be shown to deliver a tight bound for sink-trees [11].

The detailed algorithmic specification for the PMOOA is provided in Algorithm 2. The idea is to start at the sink node and deduct all flows from a node's service curve which have a different predecessor node from the flow which is currently under investigation. This is done until the source node is reached and the end-to-end service curve is completely constructed by those remaining service curves' convolution. With this PMOOA-based end-to-end service curve the end-to-end delay can then be calculated using standard network calculus results as for the TFA.

However, when compared to the addition of the nodal delay bounds as done by the TFA, this results in considerably

Algorithm 2 Simplified PMOO Analysis.

1. Let $M = \{E_1, \ldots, E_n\}$ be the set of edges the flow of interest is traversing on the way from its source to the sink. Each edge E_i has an incoming node N_{i-1} and an outgoing node N_i.

2. Let $\beta_{\text{eff}}^0 = \delta_0$ with

$$\delta_0(t) = \begin{cases} 0 & \text{if } t \leq 0 \\ \infty & \text{otherwise} \end{cases}$$

   δ_0 is the neutral element of the min-plus convolution.

3. For all $E_{1 \leq i \leq n} \in M$, add up all upper output bounds from incoming nodes $N \neq N_{i-1}$ (for $i = 0$ this means the sum of all incoming flows except the flow of interest) and update the effective service curve:

$$\beta_{\text{eff}}^i = \left[\left(\beta_{\text{eff}}^{i-1} \otimes \beta_{E_i} \right) - \sum_{N \neq N_{i-1}} \alpha_N^* \right]^+$$

   with the α_N^* according to equation 2.

4. $\beta_{\text{eff}} = \beta_{\text{eff}}^n$ is the effective service curve for the flow of interest.

---

less pessimistic bounds, because each interfering flow's burst has to be taken into consideration only once, which also justifies its name PMOOA.

A further refinement over the basic SNC from the previous subsection can be made by exploiting the additional network calculus concept of a maximum service curve [4]. A maximum service curve $\bar{\beta}_i$ gives an upper bound on the service that can be provided by a node i. This helps to tighten the output of a node to

$$\alpha_i^* = \left(\bar{\alpha}_i \otimes \bar{\beta}_i \right) \oslash \beta_i$$

Next, we present specific instances of arrival and service curves as they are used in the remainder of the paper.

3.3 Maximum Sensing Rate Arrival Curve

A straightforward option in bounding the sensing input at a given sensor node is based on its maximum sensing rate which is either due to the way the sensing unit is designed or limited to a certain value by the sensor network application's task in observing a certain phenomenon. For example, it might be known that in a temperature surveillance sensor system, the temperature does not have to be reported more than once per second at most. To model such data arrivals in the SNC model, we now provide two options: a fluid and a discrete version. The *fluid model* for each source node i is given by the frequently used token-bucket arrival curve

$$\alpha_i(t) = \gamma_{r_i, s_i}(t) = \begin{cases} r_i t + s_i & t > 0 \\ 0 & otherwise \end{cases}$$

with s_i as the size of a single data packet and r_i as the maximum sensing rate of sensor i. The more accurate *discrete model* of it is given by a staircase function representing the discrete nature of the input in both packet size and sampling period. Mathematically, that curve is specified for each sensor node i as:

$$\alpha_i(t) = s_i \cdot \left(\left\lfloor \frac{r_i t}{s_i} \right\rfloor + 1 \right)$$

The more knowledge on the sensing operation and its characteristics is incorporated into the arrival curve for the sensing input the better the worst-case bounds become.
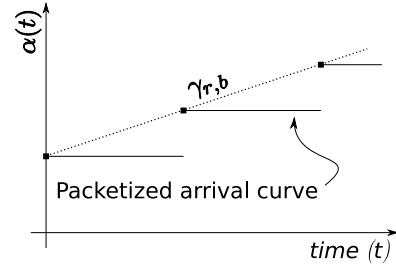

Figure 1: Maximum sensing rate arrival curve.

3.4 Minimum and Maximum Service Curve

The service curve depends on the way packets are scheduled in a sensor node which mainly depends on link layer characteristics. More specific, the service curve depends on how the duty cycle and therefore the energy-efficiency goals are set.

A typical and well-known example of a service curve from traditional traffic control in a packet-switched network is a rate-latency service curve [2]. The latency term nicely captures the characteristics induced by the application of a duty cycle concept. Whenever the duty cycle approach is applied there is the chance that sensed data or data to be forwarded just arrives after the last duty cycle is just over and thus a fixed latency occurs until the forwarding capacity is available again (for example, data is available after the TDMA slot just passed). In a simple duty cycle scheme this latency would need to be accounted for for all data transfers. For the forwarding capacity it is assumed that it can be lower bounded by a fixed rate which depends on transceiver speed, the chosen link layer protocol and the duty cycle. Ignoring the discrete nature due to packetization effects, we obtain the following *fluid version* of a minimum service curve at sensor node i:

$$\beta_i(t) = \beta_{f_i, l_i}(t) = f_i[t - l_i]^+ \tag{5}$$

where the notation $[x]^+$ equals x if $x \geq 0$ and 0 otherwise. Here f_i and l_i denote the forwarding rate and forwarding latency for sensor node i.

For the fluid version of the maximum service curve, we also assume a rate-latency curve, yet the latency must be less or equal and the forwarding rate greater or equal than the minimum service curve. In particular, for the remainder of the paper we assume the forwarding rate to be the same as for the minimum service curve and the latency to be 0, a frequent case, for example when using TDMA.

Again, we also consider discrete versions of minimum and maximum service curves to obtain more accurate models. More specifically, to model the behavior of time-slotted systems like TDMA more accurately than by a rate-latency curve, a curve as in Figure 2 can be used. Here, the latency incurred between the slots assigned to a node needs

to be modelled by a period where no service is accumulated, whereas during the assigned slots, the full medium rate can be used . L denotes the initial delay, s is the TDMA slot length, and E is the duration of a TDMA cycle, also called the *epoch*. For comparison, a rate-latency approximation is shown as curve β.
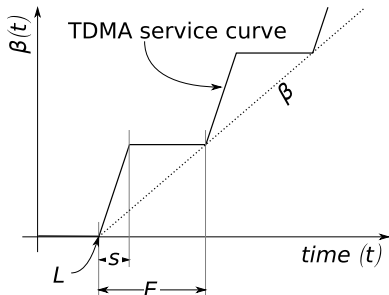

Figure 2: TDMA service curve

For the discrete version of the minimum service curve we thus assume exactly a curve as in Figure 2, whereas for the discrete version of the maximum service curve the initial delay is not incurred, so the curve is effectively shifted by L to the left.

3.5 DISCO Network Calculator

A tool called The *DISCO Network Calculator* for automatically performing the described SNC calculations is available [10]. The tool is especially useful if complex service and arrival curves have to be modeled. We used this tool to calculate the delay bounds in the evaluation section of the paper.

4. SNC NETWORK DIMENSIONING

The SNC introduced in the previous section can only be used to describe a real deployment if specific properties are fulfilled by the sensor network. In this section these properties are discussed in more detail and it is investigated how realistic sensor networks can fulfill these assumptions.

4.1 Network Structure

The SNC requires a network that forms a directed acyclic graph. This is the case in most sensor network deployments where data is collected at a central sink node. The resulting network topology in theses cases is a tree structure. It has to be noted that the SNC can also deal with cases where several sink nodes are used in the network (see [13]). In these cases the network is formed by combining several tree structures. The SNC cannot easily be applied in cases where all sensors communicate with each other and no clear direction of data flow is visible. For example, a scenario in which sensors in a field collaborate to track a vehicle would be a non-suitable scenario for the SNC.

Besides the need of having an acyclic graph structure, it is also required to know the shape of the network before the SNC can be applied. Ideally, the exact structure of the routing tree is known at the point the SNC is applied. The SNC can deal as well with a topology that is only known approximately (for example, only the maximum hop distance and maximum number of child nodes for each node are known, see [9] for details). However, in this case the SNC calculated worst-case bounds are further away from real-world experienced bounds. The SNC could therefore be applied in scenarios where the network structure can be planned carefully in advance.

A deployment fitting the aforementioned description could be, for example, a sensor network for process automation in a production plant. It is assumed that such scenarios consist of a relatively small number of nodes (a few dozen) which can be placed carefully. A non-fitting setting in contrast would be, for example, a scenario where thousands of nodes are dropped from a plane which need to organize autonomously.

4.2 Network Traffic

The first traffic requirement is given directly by the previously described topology requirements. Traffic is assumed to flow from the sensors towards the sink(s). For many sensor network applications such a traffic characteristic is the normal mode of operation. The SNC presented in Section 3 does not model traffic flowing from the sink towards the sensors. However, if the network uses a static mode of operation this limitation is not an issue. For example, all nodes might be set up statically to report their findings to the sink. If communication from the sink to the nodes is required (for example, for configuration tasks) it can be argued that such traffic is magnitudes smaller and less frequent than the traffic created by the sensor nodes. Thus, the SNC calculation would still be valid. Another option to deal with sink-node traffic is to ensure that this traffic has no impact on the sensor-sink traffic which can be achieved by using TDMA-based MAC schemes (see section 4.3).

Besides knowledge on the general direction of the traffic, the SNC requires knowledge about the traffic generated by each node. To apply the SNC, an arrival curve which denotes the worst-case characteristics of the traffic emitted by the node has to be given. The better the worst-case bound of the traffic can be described, the better the SNC calculated bounds on the message transfer delay are. In many sensor network deployments, the traffic emitted by the nodes can be described fairly accurately. It might be known that sensors report at fixed periods and thus worst-case traffic bounds and actual traffic are identical (the arrival curve describes not the worst-case, it describes the actual case).

Thus, the SNC is useful in scenarios where traffic patterns are reasonably known and bounds on the traffic can be given. The SNC is not very useful in scenarios with random and fully unpredictable traffic patterns.

4.3 Message Forwarding

The SNC is based on specific assumptions made regarding network elements. The definition of the service curve requires that a worst-case bound for the message forwarding delay can be given. Thus, a mapping between the SNC assumptions and real-world deployments requires the implementation of a deterministic message forwarding delay. The message forwarding delay can be decomposed into the message processing time and the time required to complete medium access control (MAC) protocol steps. An upper bound for the message processing time can be given if a deterministic operating system behavior is implemented. An upper bound for the required MAC access time can be given if a deterministic TDMA protocol is used. Suitable and realistic implementations of an operating system and MAC protocol are discussed in the following paragraphs.

Operating System. To achieve deterministic message processing, the operating system must be able to handle forwarding tasks in a deterministic way. Currently, event-driven operating systems such as TinyOS [15] are the preferred choice for wireless sensor networks as they are very power-efficient. Unfortunately, these systems are in their standard configuration not able to provide the necessary deterministic processing bounds. A TinyOS modification as described in [5] can be used to overcome this specific problem. If such a modified operating system is used within the sensor network, the required deterministic message processing can be implemented.

Medium Access Control. An upper bound for the forwarding delay can only be given if the MAC protocol has a deterministic behavior. This can be achieved by using a TDMA-based MAC protocol in the sensor network. The SNC cannot be used in a sensor network that implements a contention-based MAC protocol as no upper bound for the message transfer delay can be given. TDMA-based protocols are available for wireless sensor networks and can be used in practical deployments (for example [6]). The MAC protocol defines as well the duty cycle which defines the communication related power-efficiency of the sensor network.

5. SNC EVALUATION SETUP

The overall goal of the experimental evaluation is to compare SNC calculated message transfer delays and measured transfer delays in realistic deployments. This comparison is done in order to judge how useful the SNC is for dimensioning a real world network.

To obtain useful results it is necessary to carry out the experimental measurements in a controlled setting. It is required to know exactly how experimental parameters influence the measurements. In order to achieve this goal it was decided to carry out initial experiments in a simulation environment and not directly in a field trial. A simulation environment allows us to create a very realistic WSN behavior which is still fully controllable.

5.1 Setup

The network topology, traffic patterns and nodes' message forwarding capabilities are chosen such that they fit the SNC requirements given in Section 4. Thus it is ensured that the simulated sensor network can actually be implemented in practice.

Network Topology. The network structure used in the experimental evaluation has a tree structure as shown in Fig. 3. The evaluated network consists of $n = 15$ sensor nodes. Static routing is used; child nodes route a message to their parent in the tree.
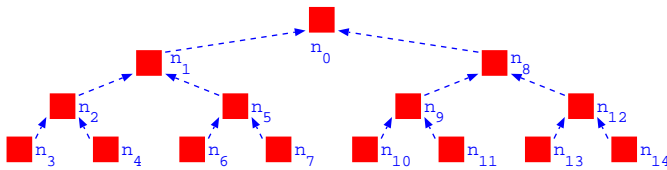

Figure 3: Network Structure

Message Forwarding. It is assumed that the operating system can prioritize message processing. Sensing tasks are executed when no packet processing is required. Packets are queued behind all other packets currently awaiting service (FIFO). An operating system behavior equivalent to the TinyOS kernel modification described in [5] is modeled to achieve deterministic behavior.

A simple TDMA based medium access control protocol is used in the network. The described protocol is based on ideas documented in [6] and can be used in a simple and well-controlled topology setting.

The base unit of the MAC protocol is the time unit *epoch* E. The epoch is divided into m time slots s_m. A number of $m \geq n + 1$ time slots is required. Each of the n sensors is assigned one time slot; the remaining time slot is used for contention-based access (for example broadcasts). A time slot is dedicated to exactly one node, thus, messages can be transmitted without collisions. Nodes must be awake for incoming messages during time slots allocated to their child nodes, their own time slot and for the contention based time slot. Thus, a node using this MAC protocol will have a duty cycle of $d = 4/m$ in a binary tree topology.

At startup of the network, time synchronization among nodes is necessary and all nodes are in listening mode. The sink node n_0 announces the start of the epoch E by sending an epoch message in time slot s_0. The child nodes receive the epoch message and thus are now synchronized with the epoch. Now the child nodes re-send the epoch message in their time slot. This process is repeated until every node received the epoch message. Thereafter, the TDMA scheme is set up and nodes can start forwarding messages to the sink without collisions. Periodically, the sink might refresh time synchronization by re-sending an epoch message.

For the experimental evaluation an epoch of $E = 100ms$ and $m = 20$ is used. Thus, each time slot s_m has a duration of $5ms$. We model a Chipcon CC2420 Zigbee transceiver with a transmission speed of $250kbps$ and a message size of $50byte$. Thus, a message transmission has a duration of $1.6ms$ which fits well in the selected slot size of $5ms$.

It has to be noted that the protocol can be used in realistic deployments as large fields are not assumed. Furthermore, time slots are larger than the actual message transmission time. Therefore, time for packet processing is available and it is possible to cater for jitter in time synchronization. The described MAC protocol does not scale to large deployments. However, it is simple and efficient in small scenarios and can be implemented in a practical deployment.

The message forwarding used in the experimental setup can be bounded in the SNC by a rate-latency service curve as described by Equation (5). The forwarding rate is $f_i = 4kbit/s$ and the forwarding latency for node i is $L_i = E - s_i = 95ms$.

Network Traffic. The traffic flows from the sensor nodes towards the sink. It is assumed that every node within the network creates periodic sensor reports with a frequency of p. In a realistic sensor network deployment it cannot be assumed that these reports will be generated in each node at exactly the same time. Thus, the experiment time is divided in slots of size $T = 1/p$ duration. For the evaluation, a time value of $T = 1s$ is used. In each slot each node creates one message at a randomly selected time between 0 and δ (with $0 \leq \delta < T$). In the remaining paper δ is called *jitter*.

For small δ, all nodes in the field report their measurement at roughly the same time towards the sink. This could be the case if for example all sensors are triggered by the same observed phenomenon or because they are set-up to report findings periodically.

The network traffic used in the experimental setup can be bounded in the SNC by a maximum sensing rate arrival curve as described in section 3.3. For small δ all nodes in the field are closely synchronized and worst-case situations described by the SNC may be expected to be observed.

6. SNC EXPERIMENTAL EVALUATION

The evaluation setup described in the previous section is used to evaluate the usefulness of the SNC calculations.

The jitter parameter δ in the traffic description is used as variable parameter in the experiments. The SNC can be used to calculate the worst-case bounds for the message transfer delay. By using δ it is possible to "tune" the experimental setup towards a worst-case behavior. It seems reasonable to assume that more packets close to the SNC calculated worst-case delay will be observed for small δ as all nodes become active at roughly same time. If δ is large, few packets will experience the worst-case delay as network activity is spread over time.

6.1 SNC Calculated Message Transfer Delay

The worst-case message transfer delay is experienced in the experimental setup by messages transmitted from leaf nodes in the topology (see Fig. 3). The problem was set up using maximum sensing rate arrivals of one $50B$ packet every second with a $50B$ burst and service with a $95ms$ latency, $5ms$ slot duration and a medium rate[2] of $10kB/s$. The *DISCO Network Calculator* was used to automatically perform the necessary SNC calculations.

The first calculations were done using fluid versions of arrival- and service curves, with the service curve modeling a sustained rate of $100kB/s$, $10kB$ per $100ms$. A total flow analysis (TFA) summing up the per-hop delays yielded a delay bound of $5190ms$. Next, the PMOO analysis was used, again with fluid models of arrival and service curves. The PMOOA resulted in a delay bound of $2367.5ms$, obviously a considerable improvement. To model the data flows more accurately, the curves were changed to their discrete versions as discussed in sections 3.3 and 3.4. A PMOO analysis using these discrete versions of the curves resulted in a delay bound of $1490ms$, while the TFA result decreased to $4100ms$. Finally, by incorporating the maximum service curve concept, i.e. by modeling an upper bound on the service provided by the servers, the PMOO bound was further reduced to $890ms$.[3] Hence, we can observe a very significant improvement of the delay bound for this scenario using the advanced SNC concepts of PMOOA and the maximum service curve as well as the discrete versions of arrival- and service curves.

In other words, if the network is dimensioned according to the setup described in the previous section, all messages can be delivered in less than $890ms$. If our application can deal with such a maximum delay our sensor network is di-

mensioned correctly. If the delay is too high, we might want to increase the forwarding rate (e.g. by reducing the epoch E of the MAC layer). If the delay is lower than required we might want to decrease the forwarding capabilities of the nodes which would result in better energy savings (the duty cycle will improve).

The open question is now how close really observed delays are to the calculated bound of $890ms$. In addition, it is important to find out how many packets will actually experience this worst-case delay. These remaining questions will be addressed by the measurements described in the next subsection.

6.2 Measured Message Transfer Delays

The parameter δ is used as variable parameter in the experiments. The message transfer delay d_i of all packets is recorded during the experiments. The delay is measured from the time the message is generated until it reaches the sink node n_0. An experiment duration is 600 seconds. Thus, with all 14 nodes reporting once a second, 8400 messages are generated during one experiment[4]. Values between $\delta = 0ms$ and $\delta = 500ms$ are used as jitter within the experiments.

Fig. 4 shows the delay histogram for $\delta = 0ms$. In this case, all nodes in the field report at exactly the same time, at the beginning of an epoch E. As a result, the delay distribution is very homogeneous as forwarding queues in the network are always filled in the same way. Basically, the network is free of any random event.

The measured worst-case delay is $d_{max} = 640ms$. This worst-case delay is experienced by 600 messages during the experiment. These messages are generated by node n_{14}. A message generated by n_{14} takes 6 epochs E to travel to node n_8 and being scheduled for transmission as messages from n_{14} queue behind all other messages generated in the branch. Finally from node n_8 to n_0 $40ms$ are added as node n_8 uses time slot number 8 (slot size $5ms$).
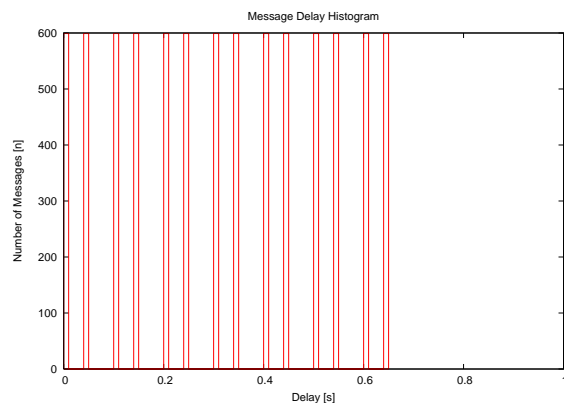

Figure 4: Delay histogram; fully synchronized network ($\delta = 0ms$).

As expected, all measured delays are below the SNC calculated worst-case bound of $890ms$. The calculated worst-case bound of $890ms$ is 39% above the measured worst-case delay of $640ms$. The measured worst-case delay is experienced by 7.1% of all messages. In addition, 28.6% of the messages

---

[2]The medium rate is chosen to reflect the transmission of at most one $50B$-packet per slot.

[3]The TFA did not improve when the maximum service curve was used in the TDMA model.

---

[4]Not all 8400 messages might reach the sink as some messages might still be in transit when the simulation ends.

experience a delay between $445ms$ and $890ms$ (a window describing messages which experience more than half of the SNC calculated worst-case bound).

Fig. 5 shows the delay histogram for $\delta = 50ms$. In this case, all nodes have a slight offset when generating messages. Again, all measured delays are below the SNC calculated worst-case bound of $890ms$. The calculated worst-case bound of $890ms$ is 20.4% above the measured delay of $739ms$ (This message was generated by node n_{11}). It has to be noted that the measured worst-case delay is significantly higher than for $\delta = 0$. Thus, the fully synchronized network with $\delta = 0$ does not represent - as one might think - the worst possible scenario. A slight jitter in message generation patterns enables unfavorable scheduling situations. In this experimental run, 36.6% of the messages experience a delay between $445ms$ and $890ms$.
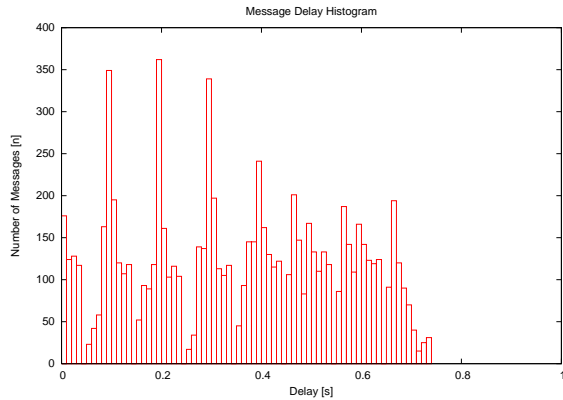

Figure 5: Delay histogram; synchronized network ($\delta = 50ms$).

Fig. 6 shows the delay histogram for $\delta = 250ms$. In this case, nodes are not closely synchronized regarding message generation. All measured delays are below the SNC calculated worst-case bound of $890ms$. The calculated worst-case bound of $890ms$ is 6.6% above the measured worst-case delay of $835ms$ (This message was generated by node n_{11}). In this experimental run, 30.7% of the messages experience a delay between $445ms$ and $890ms$.
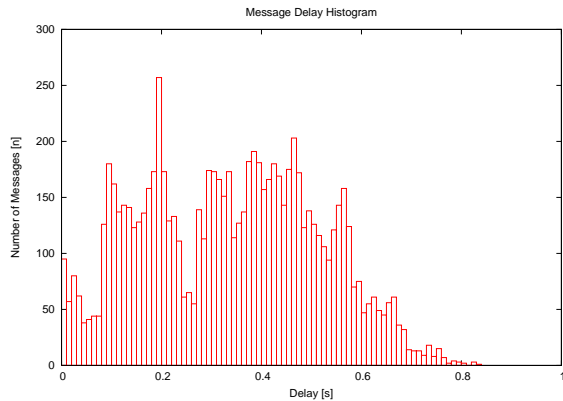

Figure 6: Delay histogram; un-synchronized network ($\delta = 250ms$).

Fig. 7 shows a summary of all experimental runs with different δ. The abscissa shows the simulation runs for different δ ($0ms < \delta < 500ms$); the ordinate depicts the delay. The figure shows the calculated worst-case delay of $890ms$, the worst-case delay observed during measurements and the 90%-quantile of measured delays. 10% of the delay measurements in the experiment run are above and 90% are below the 90%-quantile value. Thus, the 90%-quantile graph gives a quantitative indication of how close the measured delays are to the SNC calculated worst-case bound.

All measured delays are below the SNC calculated worst-case bound of $890ms$. Measured worst-case delay and calculated worst-case delay are closest for $\delta = 475ms$. Here, the calculated worst-case bound of $890ms$ is only 2.7% above the measured worst-case delay of $867ms$.

The graphs show that the worst-case is relatively independent of the jitter δ. Even for a high jitter a high worst-case delay can still occur. However, the 90%-quantile is dropping slightly with the amount of jitter introduced. This indicates that message transfer delays close to the calculated worst-case bounds are less likely to occur.
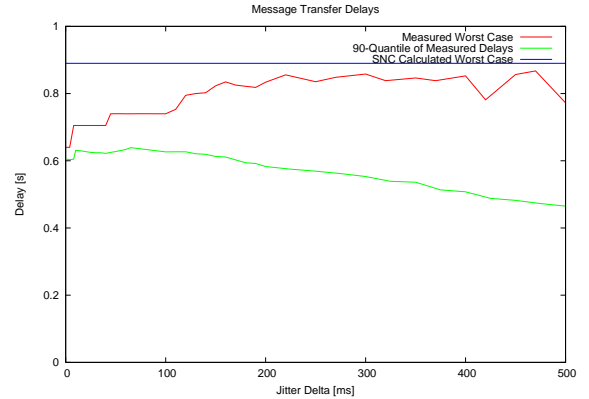

Figure 7: Measured worst-case delays in dependence of δ ($0ms < \delta < 500ms$).

Fig. 8 shows the percentage of messages that experience a delay between $445ms$ and $890ms$ in dependence of the jitter δ. These messages experience more than half of the SNC calculated worst-case bound. The graph shows that lower data transport delays are measured with increasing jitter.
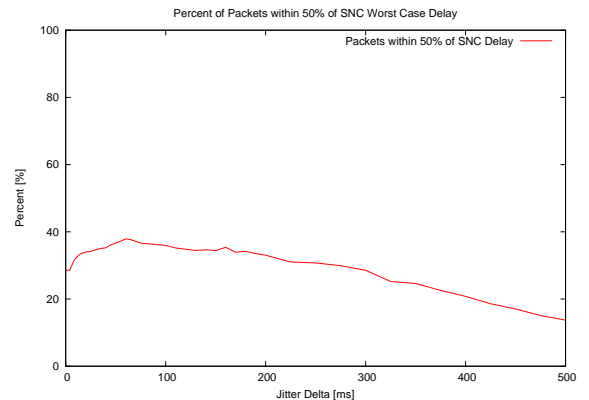

Figure 8: Percentage of messages with $455ms < d_i < 890ms$.

6.3 Findings

Our experiments show that the SNC calculated worst-case delay is close to the experienced delays in a realistic application setting. Actual measured delays in the experiment can be predicted by the SNC as close as 2.7% (for $\delta = 475ms$).

More than 30% (for $10ms < \delta < 250ms$) of messages experience a delay between $445ms$ and $890ms$. This delay window describes messages which experience more than half of the SNC calculated worst-case bound. If the sensor network would be dimensioned without using a dimensioning tool such as the SNC and a provisioning error of 50% is made, about 30% of the messages will not be delivered in time and might have to be counted as loss. Losses of this magnitude, in addition to packet losses due to noisy channels, are too high for many sensor network applications that require performance guarantees. Dimensioning errors of 50% are easily possible if no useful dimensioning tool is available. Over-provisioning by several orders would reduce the risk of error but is not a viable option in resource constrained wireless sensor networks. Usually, over-provisioning would translate to higher energy consumption and, thus, shorter network lifetime.

7. CONCLUSION

The goal of this paper was to shed light on the question of whether the SNC can be a valuable tool for the dimensioning of WSNs. The presented results show that SNC predicted worst-case message transfer delay bounds are very close to measured delays in simulations of restricted, yet realistic WSNs deployment scenarios. For example, a difference of only 2.7% between calculated and measured worst-case delay was observed for one specific experiment and in all experiments it was never off by large. Thus, we validated that the SNC can be used as a reasonably accurate dimensioning tool for the class of wireless sensor network applications investigated in this paper. It should be noted that this positive answer to our basic question on the suitability of the SNC as a WSN dimensioning tool is also a result of the improvements for the basic SNC (PMOO analysis method, maximum service curve, discrete modeling) that we introduced in this paper.

We believe that the SNC can be used to derive practical design guidelines for a wireless sensor network that requires strict message transfer delay guarantees. The SNC can be used as dimensioning tool to avoid a) over-provisioned sensor networks b) undesired message transfer delays.

We plan to repeat the presented measurements using a practical deployment instead of a simulated sensor network environment. We expect that these real-world measurements will confirm the presented simulation results.

8. REFERENCES

[1] T.R. Abdelzaher, S. Prabh, and R. Kiran. On real-time capacity limits of multihop wireless sensor networks. In *RTSS '04: Proceedings of the 25th IEEE International Real-Time Systems Symposium*. IEEE, December 2004.

[2] Jean-Yves Le Boudec and Patrick Thiran. *Network calculus: a theory of deterministic queuing systems for the internet*. Springer-Verlag New York, Inc., New York, NY, USA, 2001.

[3] C.-F. Chiasserini and M. Garetto. Modeling the performance of wireless sensor networks. In *Proc. IEEE INFOCOM*, March 2004.

[4] R. L. Cruz. SCED+: Efficient management of quality of service guarantees. In *Proc. IEEE INFOCOM*, volume 2, pages 625–634, March 1998.

[5] Cormac Duffy, Utz Roedig, John Herbert, and Cormac J. Sreenan. Adding Preemption to TinyOS. In *Proceedings of the The Fourth Workshop on Embedded Networked Sensors (EmNets2007), Cork, Ireland*. ACM Press, June 2007.

[6] L. van Hoesel and P. Havinga. A lightweight medium access protocol (LMAC) for wireless sensor networks. In *INSS04*, Tokyo, Japan, June 2004.

[7] Anis Koubaa, Mario Alves, and Eduardo Tovar. Modeling and worst-case dimensioning of cluster-tree wireless sensor networks. In *RTSS '06: Proceedings of the 27th IEEE International Real-Time Systems Symposium*, pages 412–421, Washington, DC, USA, 2006. IEEE Computer Society.

[8] Jens Schmitt and Utz Roedig. Sensor Network Calculus - A Framework for Worst Case Analysis. In *Proceedings of IEEE/ACM International Conference on Distributed Computing in Sensor Systems (DCOSS'05), Marina del Rey, USA*, pages 141–154. Springer, LNCS 3560, June 2005. ISBN 3-540-26422-1.

[9] Jens Schmitt and Utz Roedig. Worst Case Dimensioning of Wireless Sensor Networks under Uncertain Topologies. In *Proceedings of 3rd International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt'05), Workshop on Resource Allocation in Wireless Networks, Riva del Garda, Italy*. IEEE, April 2005. ISBN 0-9767294-0-7.

[10] Jens B. Schmitt and Frank A. Zdarsky. The DISCO Network Calculator - A Toolbox for Worst Case Analysis. In *Proceedings of the First International Conference on Performance Evaluation Methodologies and Tools (VALUETOOLS'06), Pisa, Italy*. ACM, November 2006.

[11] Jens B. Schmitt, Frank A. Zdarsky, and Markus Fidler. Delay Bounds under Arbitrary Multiplexing. Technical Report 360/07, University of Kaiserslautern, Germany, July 2007.

[12] Jens B. Schmitt, Frank A. Zdarsky, and Ivan Martinovic. Performance Bounds in Feed-Forward Networks under Blind Multiplexing. Technical Report 349/06, University of Kaiserslautern, Germany, April 2006.

[13] Jens B. Schmitt, Frank A. Zdarsky, and Utz Roedig. Sensor Network Calculus with Multiple Sinks. In *Proceedings of IFIP NETWORKING 2006, Workshop on Performance Control in Wireless Sensor Networks, Coimbra, Portugal*, pages 6–13. Springer LNCS, May 2006. ISBN 972-95988-5-1.

[14] J.A. Stankovic, T. Abdelzaher, C. Lu, L. Sha, and J.Hou. Real-time communication and coordination in embedded sensor networks. *Proceedings of the IEEE*, 91:1002–1022, 2003.

[15] Tinyos. http://www.tinyos.net/, 2007.