Sensor Network Calculus – A Framework for Worst Case Analysis

Jens B. Schmitt[1] and Utz Roedig[2]

[1]Distributed Computer Systems Lab, University of Kaiserslautern, Germany
[2]Mobile & Internet Systems Laboratory, University College Cork, Ireland

Abstract. To our knowledge, at the time of writing no methodology exists to dimension a sensor network so that a worst case traffic scenario can be definitely supported. In this paper, the well known network calculus is tailored so that it can be used as a tool for worst case traffic analysis in sensor networks. To illustrate the usage of the resulting sensor network calculus, typical example scenarios are analyzed by this new methodology. Sensor network calculus provides the ability to derive deterministic statements about supportable operation modes of sensor networks and the design of sensor nodes.

1 Introduction

1.1 Motivation

Decisions in daily life are based on the accuracy and availability of information. Sensor networks can significantly improve the quality of information as well as the ways of gathering it. For example sensor networks can help to get higher fidelity information, acquire information in real time, get hard-to-obtain information and reduce the cost of obtaining information. Therefore it is commonly assumed that sensor networks will be applied in many different areas in the future and they can be viewed as an important part in the vision of ubiquitous/pervasive computing [1].

Application areas for sensor networks might be production surveillance, traffic management, medical care or military applications. In these areas it is crucial to ensure that the sensor network is functioning even in a worst case scenario. It must be clear that the sensor network can support all possible communication patterns that might occur in the network without being overloaded. If a sensor network is used for example for production surveillance it must be ensured that messages indicating a dangerous condition are not dropped. If functionality in worst case scenarios cannot be proven, people might be in danger and the production system might not be certified by authorities.

As it may be difficult or even impossible to produce the worst case in a real world scenario or in a simulation in a controlled fashion an analytical framework is desirable that allows a worst case analysis in sensor networks. Network calculus

[4] is a relatively new tool that allows worst case analysis of packet-switched communication networks. Network calculus has successfully been applied to model wired IP-based networks built on QoS technologies like Integrated Services or Differentiated Services [2],[3].

1.2 Goals

Our long-term goal is to develop a *sensor network calculus* that allows an analytical investigation of performance-related characteristics of wireless sensor networks. This paper represents the first step towards such a framework. As sensor networks differ in many aspects from traditional wired IP-based networks, existing results from traditional network calculus cannot be transferred directly. In particular, sensor networks have different constraints, such as battery powered nodes and dynamic topologies. It is necessary to incorporate these constraints in the framework, so that they can be analyzed. For these reasons the following is presented in this paper:

> *An analytical framework based on network calculus to dimension sensor networks, in particular taking into account the various trade-offs and interdependencies between node power consumption, node buffer requirements and information transfer delay.*

1.3 Problem Scope

Different applications running on top of a sensor network might have different requirements regarding the information extracted from the field. One important requirement is a bound on the maximum *information transfer delay* for data delivery. If information is delayed too long on the transport path, the application cannot use the information as it is considered out-dated. At each hop of the transport path, a message can be delayed. If a sensor node that generates a message is some hops away from the sink, the message delay accumulates over the hops. If several messages are delayed in one node at the same time, buffer space for the messages must be available. Thus, information transfer delay is correlated with the *buffer requirements* of a sensor node. The delay in each node is caused by two interdependent aspects. First, the delay depends on the traffic that a node has to process (*arrival rate*). Second, the node needs a specific amount of time to receive, process and send a message (*service rate*). The first aspect depends on the *network topology*, as the traffic that enters a node might be generated by several other nodes. The second aspect is dominated by the reception delay caused by the common usage of *duty cycles* which defines the *power consumption* of a sensor node.

Most energy in a sensor node is used for communication and one very effective and generic way to solve the problem is to optimize power consumption on the data link layer by implementing duty cycles. Thereby, the receiver alters periodically between a power intensive idle mode (idle duration T_1) and a power saving sleep mode (sleep duration T_2) state. Thus, the duty cycle δ is defined as $\delta = T_1/(T_1 + T_2) \times 100[\%]$. To transmit, a sender has to catch the receiver in its idle phase. This is achieved by using a long preamble in front of each single message notifying the receiver of an incoming message. The need for the long preamble implies that the effective channel bandwidth is reduced and the information transfer delay is increased. The proportion between sleep and idle phase defines the power consumption of a node and the possible forwarding rate (service curve). The usage of duty cycles allows the lifetime of a node to be stretched from several days to over one year [6]. Therefore nearly all practical sensor networks in use today implement duty cycles as a method to extend the network lifetime [13],[8].

1.4 Outline

In the remaining paper it is shown how network calculus can be tailored and extended so that a worst case analysis of the relevant quantities in sensor networks is possible. Section 2 gives a brief summary of the network calculus and the basic sensor network calculus approach is presented. Section 3 shows in detail how the model is instantiated and applied. Section 4 presents some use cases that show how the model can be used to analyze common real-world scenarios. Section 5 presents related work and Section 6 concludes the paper.

2 Sensor Network Calculus

2.1 Background on Network Calculus

Network calculus is the tool to analyze flow control problems in networks with particular focus on determination of bounds on worst case performance. It has been successfully applied as a framework to derive deterministic guarantees on throughput, delay, and to ensure zero loss in packet-switched networks [4]. Network calculus can also be interpreted as a system theory for *deterministic* queueing systems, based on min-plus algebra. What makes it different from traditional queueing theory is that it is concerned with worst case rather than average case or equilibrium behaviour. It thus deals with bounding processes called arrival and service curves rather than arrival and departure processes themselves.

Next some basic definitions and notations are provided before some basic results from network calculus are summarized. In depth results can be found in [4].

Definition 1. *The input function $R(t)$ of an arrival process is the number of bits that arrive in the interval $[0,t]$. In particular $R(0) = 0$, and R is wide-sense increasing, i.e. $R(t_1) \leq R(t_2)$ for all $t_1 \leq t_2$.*

Definition 2. *The output function $R^*(t)$ of a system S is the number of bits that have left S in the interval $[0,t]$. In particular $R^*(0) = 0$, and R is wide-sense increasing, i.e. $R^*(t_1) \leq R^*(t_2)$ for all $t_1 \leq t_2$.*

Definition 3. *Min-Plus Convolution. Let f and g be wide-sense increasing and $f(0) = g(0) = 0$. Then their convolution under min-plus algebra is defined as*

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

Definition 4. *Min-Plus Deconvolution. Let f and g be wide-sense increasing and $f(0) = g(0) = 0$. Then their deconvolution under min-plus algebra is defined as*

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

Now, by means of the min-plus convolution, the arrival and service curve are defined.

Definition 5. *Arrival Curve. Let α be a wide-sense increasing function such that for $t < 0$. α is an arrival curve for an input function R iff $R \leq R \otimes \alpha$. It is also said that R is α-smooth or R is constrained by α.*

Definition 6. *Service Curve. Consider a system S and a flow through S with R and R^*. S offers a service curve β to the flow iff β is wide-sense increasing and $R^* \geq R \otimes \beta$.*

From these, it is now possible to capture the major worst-case properties for data flows: maximum delay and maximum backlog. These are stated in the following theorems.

Theorem 1. *Backlog Bound. Let a flow $R(t)$, constrained by an arrival curve α, traverse a system S that offers a service curve β. The backlog $x(t)$ for all t satisfies:*

$$x(t) \leq \sup_{s \geq 0} \{\alpha(s) - \beta(s)\} = v(\alpha, \beta) \tag{1}$$

$v(\alpha, \beta)$ is also often called the vertical deviation between α and β.

Theorem 2. *Delay Bound. Assume a flow $R(t)$, constrained by arrival curve α, traverses a system S that offers a service curve β. At any time t, the virtual delay $d(t)$ satisfies:*

$$d(t) \leq \sup_{s \geq 0} \{\inf\{\tau \geq 0 : \alpha(s) \leq \beta(s+\tau)\}\} = h(\alpha, \beta) \tag{2}$$

$v(\alpha, \beta)$ is also often called the horizontal deviation between α and β.

As a system theory network calculus offers further results on the concatenation of network nodes as well as the output when traversing a single node. Especially the latter for which now the min-plus deconvolution is used will be of high importance in the sensor network setting as it potentially involves a so-called *burstiness increase* when a node is traversed by a data flow.

Theorem 3. *Output Bound. Assume a flow $R(t)$ constrained by arrival curve α traverses a system S that offers a service curve β. Then the output function is constrained by the following arrival curve*

$$\alpha^* = \alpha \oslash \beta \geq \alpha \tag{3}$$

Theorem 4. *Concatenation of Nodes. Assume a flow $R(t)$ traverses systems S_1 and S_2 in sequence where S_1 offers service curve β_1 and S_2 offers β_2. Then the resulting system S, defined by the concatenation of the two systems offers the following service curve to the flow:*

$$\beta = \beta_1 \otimes \beta_2 \tag{4}$$

2.2 Sensor Network System Model

In this paper the common class of single base station oriented operation models is assumed. Within the traffic that is modeled only the sensor reports are taken into account. Traffic generated from the base station towards the nodes (e.g. interests [16] to set up the network structure and configure the nodes) is explicitly not taken into account. This is considered feasible based on the assumption that the traffic flowing towards the sensors is magnitudes lower than traffic caused by the sensing events. Furthermore, it is assumed that the routing protocol being used forms a tree in the sensor network.[1]Hence N sensor nodes arranged in a directed acyclic graph are given.

Each sensor node i senses its environment and thus is exposed to an input function R_i corresponding to its sensed input traffic. If sensor node i is not a leaf node of the tree then it also receives sensed data from all of its child nodes $child(i,1),\ldots,child(i,n_i)$, where n_i is the number of child nodes of sensor node i. Sensor node i forwards/processes its input which results in an output function $R_i^*$from node i towards its parent node.

Now the basic network calculus components, arrival and service curve, have to be incorporated. First the arrival curve $\bar{\alpha}_i$ of each sensor node in the field has to be derived. The input of each sensor node in the field, taking into account its sensed input and its childrens input, is given by:

$$\bar{R}_i = R_i + \sum_{j=1}^{n_i} R^*_{child(i,j)} \tag{5}$$

Thus, the arrival curve for the total input function for sensor node i is given by:

$$\bar{\alpha}_i = \alpha_i + \sum_{j=1}^{n_i} \alpha^*_{child(i,j)} \tag{6}$$

Candidates for actual arrival curves on the sensed input are discussed in Section 3.1.

---

[1] If multiple sinks are assumed one simple way to use the following methodology would be to analyze each tree resulting from a given sink in isolation and later on additively combine interesting quantities as for example buffer requirements at a certain sensor node.

Second, the service curve has to be specified. The service curve depends on the way packets are scheduled in a sensor node which mainly depends on link layer characteristics (see Section 1.3). More specific, the service curve depends on how the duty cycle and therefore the energy-efficiency goals are set.[2] Again the discussion of actual candidates for sensor node service curves is deferred to Section 3.2 when the whole sensor network calculus framework has been presented. Finally, the output of sensor node i, i.e. the traffic which it forwards to its parent in the tree, is constrained by the following arrival curve:

$$\alpha_i^* = \bar{\alpha}_i \oslash \beta_i = \left(\alpha_i + \sum_{j=1}^{n_i} \alpha_{child(i,j)}^* \right) \oslash \beta_i \tag{7}$$

In order to calculate a network-wide characteristic like the maximum information transfer delay or local buffer requirements especially at the most challenged sensor node just below the sink (which is called node 1 from now on) an iterative procedure to calculate the network internal flows is required:

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 according to (3). 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 (7) and they can again be marked "calculated".
4. If node 1 is marked "calculated" the algorithm terminates, otherwise go to step 3.

After the network internal flows are computed according to this procedure, the local worst case buffer requirements B_i and per node delay bounds D_i for each sensor node i can be calculated according to Theorem 1 and 2:

$$B_i = v(\bar{\alpha}_i, \beta_i) = \sup_{s \geq 0} \{ \bar{\alpha}_i(s) - \beta_i(s) \} \tag{8}$$

$$D_i = h(\bar{\alpha}_i, \beta_i) = \sup_{s \geq 0} \{ \inf \{ \tau \geq 0 : \bar{\alpha}_i(s) \leq \beta_i(s + \tau) \} \} \tag{9}$$

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:

$$\bar{D}_i = \sum_{i \in P(i)} D_i \tag{10}$$

The maximum information transfer delay in the sensor network can then obviously be calculated as

[2] The service curve might further depend on whether more advanced sensor network characteristics like in-network processing, e.g. for aggregation or even prioritization of some traffic is provided.

$$\bar{D} = \max_{i=1,\ldots,N} \bar{D}_i \tag{11}$$

Discussion Readers very knowledgeable in network calculus may wonder about the hop-by-hop calculation of the total delay as specified in (11) and whether it would not be possible to derive a network-wide service curve based on the concatenation result of Theorem 4. While due to the traffic aggregation inside the network the concatenation result cannot be applied directly, there is in fact a way to still derive a network-wide service curve based on modified service curves that take into account the effects of cross-traffic on a data flow [4]. However, the bounds achieved in this way are not necessarily lower than for (11). This depends on the actual parameters of arrival and service curves. Furthermore, we believe that the hop-by-hop calculation will lend itself better towards integrating in-network processing into future, more elaborate extensions of the model.

Often, sensor network applications may regard message transfer delay only as a constraint and primarily care about maximizing their lifetime. The length of the duty cycle, and thus the energy consumption properties of the sensor nodes, are incorporated into the service curve as will be discussed in Section 3.2. Hence, instead of calculating delay bounds and buffer requirements as described above, the calculations could also start with a given delay/buffer requirement and work out the length of the duty cycle and thus the power consumption level and therefore the network lifetime.

3 Instantiating the Model

Before progressing to some numerical examples the abstract sensor network calculus model needs to be instantiated with concrete arrival and service curves. In the following subsections these crucial aspects and their influence on the worst case behaviour of the system are discussed in a qualitative fashion before in Section 4 more quantitative results are presented.

3.1 Arrival Curve Candidates

Maximum Sensing Rate The simplest 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. The arrival curve for a sensor node i corresponding to simply putting a bound on the maximum sensing rate is given by

$$\alpha_i(t) = p_i t = \gamma_{p_i,0}(t) \tag{12}$$

Note that the assumption is made that each sensor node has its individual arrival curve respectively maximum sensing rate.

This arrival curve can be used in situations where all sensor nodes are set up to periodically report the condition in a sensor field. Thereby each sensor has a maximum possible rate with which the sensing information can be reported.

Average Sensing Rate Depending on the sensor network application the maximum sensing rate arrival curve might lead to very conservative bounds if the maximum sensing rate is only rarely the actual sensing rate. In this situation it would be much more useful if the arrival curve could be based on the average sensing rate. Additionally there should be permission of some short-term fluctuations if the sensing must be intensified for certain periods of high activity in the field. However, in order to avoid the use of the maximum sensing rate arrival curve it is crucial that the time during which the average sensing rate may be exceeded can be upper bounded. In many applications that should be possible since after some time the phenomenon will disappear again or has to be acted on such that it disappears again (e.g. in a sensor network that also comprises actuators). The arrival curve that captures the average sensing rate with short-term fluctuations for sensor node i is given by

$$\alpha_i(t) = s_i t + b_i = \gamma_{s_i,b_i}(t) \tag{13}$$

This affine arrival curve can be shown to be equivalent to the famous token/leaky bucket as it is known from traditional traffic control [4]. It allows sensing at a higher rate than s_i for short periods of time but in the long run only allows sensing at the average rate s_i.

This arrival curve can be used to describe situations in which sensors usually report with a low rate. If a phenomenon is detected in the vicinity of the sensor, the sensing rate is increased for a fixed amount of time.

Discussion The set of sensible arrival curve candidates is certainly larger than the arrival curves described above. 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. We consider it a strength of the sensor network calculus framework that it is open with respect to arbitrary arrival curves. On the other hand, the options presented above may be sufficient in a large number of sensor network scenarios.

3.2 Service Curve Candidates

The service curve captures the characteristics with which sensor data is forwarded by the sensor nodes towards the sink. It abstracts from the specifics and idiosyncracies of the link layer and makes a statement on the minimum service that can be assumed even in the worst case.

Rate-Latency Service Curve A typical and well known example of a service curve from traditional traffic control in a packet-switched network is given by

$$\beta_{R,T}(t) = R(t - T)^+ \tag{14}$$

where the notation $(x)^+$ denotes x if $x \geq 0$ and 0 otherwise. This is often also called a rate-latency service curve and results from the use of many popular packet schedulers (for example Weighted Fair Queueing (WFQ) [11]) many of which can be generalized as guaranteed rate or latency rate schedulers [9], [10]. While for sensor networks there may often be neither a necessity nor the resources (e.g. energy, computational power, memory capacity) for a sophisticated scheduling algorithm like WFQ, the class of rate-latency service curves is still very interesting. This is due to the fact that 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 (of the next hop!) is just over and thus a fixed latency occurs until the forwarding capacity is available again. In the simple duty cycle scheme presented in Section 1.3 this latency would need to be accounted for for all data transfers since the preamble length is fixed, in schemes where the data is repeated for a certain amount of time and a feedback from the receiver signals when the sender can stop this repetition, the latency would really represent a worst case scenario as just mentioned. 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. So, with some new parameters the following service curve at sensor node i is obtained:

$$\beta_i(t) = \beta_{f_i,l_i}(t) = f_i(t - l_i)^+ \tag{15}$$

Here f_i and l_i denote the forwarding rate respectively forwarding latency for sensor node i.

4 Sensor Network Calculus at Work

In this section some numerical examples for the previously presented sensor network calculus framework are described. These examples are chosen with the intention of describing realistic and common application scenarios, yet they are certainly simplifying matters to some degree for illustrative purposes. As mentioned in the previous sections, the sensor network calculus framework allows, from a worst case perspective, to relate the following local characteristics:

- *Sensing Activity*: this parameter is described in the framework by the *arrival curve* concept;
- *Buffer Requirements*: the buffer requirements of each node are described by the *backlog bound;*

to the following global characteristics:

– *Information Transfer Delay*: the delay in each node is described by the *delay bound;*
– *Network Lifetime*: the energy consumption is described by the *duty cycle* represented in the *service curve.*

The goal in using sensor network calculus is to determine specific values for these characteristics for a given application scenario. The scenario itself is characterized by further constraints such as *topology* and *routing.*

4.1 Basic Scenario

The intention of this example is to analytically explore the possible range of the characteristics discussed above in a realistic scenario. Thereafter it is analyzed in which operation range a state of the art sensor node could be used to form the sensor field.

Topology and Routing The sensor field is assumed to be a grid, the distance between the sensors is d. Fig. 1 shows the lower half of a grid shaped sensor field with the base station (sink) located in its center. The size of the field is $8d \times 8d$, containing $N = 80$ sensors each with an idealized transmission range of $\sqrt{2}d$.

For the routing protocol, the Greedy Perimeter Stateless Routing (GPSR) protocol is used [12]. All nodes in GPSR must be aware of their position within a sensor field. Each node communicates its current position periodically to its neighbors through beacon packets. In the given static scenario, these beacons have to be transmitted only once. Upon receiving a data packet, a node analyzes its geographic destination. If possible, the node always forwards the packet to the neighbor geographically closest to the packet destination. If there is no neighbor geographically closer to the destination, the protocol tries to route around the hole in the sensor field. This routing around a hole is not used in the described topology. In Fig. 1 the resulting structure of the communication paths is shown.

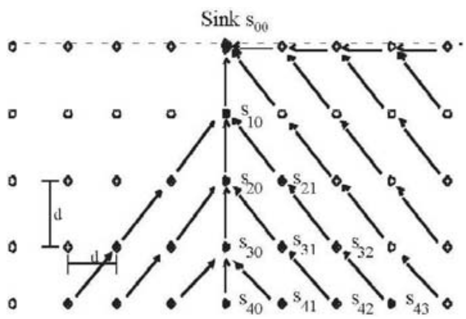

Fig. 1. Sensor Field with Grid Layout.

Sensing Activity It is assumed that the sensor field is used to collect data periodically from each of the sensors. Each sensor can report with a maximum report frequency of p. Thus, the maximum sensing rate arrival curve described by (12) is used to model the upper bound of the sensing activity of each node in the sensor field. A homogeneous field is assumed, hence

$$\alpha_i(t) = pt = \gamma_{p,0}(t) \tag{16}$$

Each node additionally receives traffic from its child nodes according to the traffic pattern implied by the topology and the routing protocol (see 1). Therefore, the arrival curve $\bar{\alpha}_i$ for the total input of a sensor node i is given by equation (6). Later it will be shown in detail how the relevant $\bar{\alpha}_i$ can be calculated.

Network Lifetime To achieve a high network lifetime a duty cycle of $\delta = 1\%$ is set for the nodes in the network. As a sensor node, the Mica-2 [13] platform is assumed. Mica-2 supports a link speed of 19.2 kbit/s. The minimum idle time of the transceiver is $T_1 = 11$[ms] (3ms to begin sampling, 8ms minimum preamble length), the corresponding sleep time is $T_2 = 1085$[ms]. Thus, a maximum packet forwarding rate of 0.89[packets/s] ($f = 258$[bit/s]) can be achieved.[3]The resulting latency for the packet forwarding is $l = T_1 + T_2$. This packet forwarding scheme can be described by the rate-latency service curve as described by equation (15) in Section 3.2:

$$\beta_i(t) = \beta_{f,l}(t) = f(t-l)^+ = 258(t-1.096)^+[bit] \tag{17}$$

Calculation After defining the scenario, the sensor network calculus framework can now be used to evaluate the characteristics of interest and their interdependencies. Goal of the calculation is to determine these characteristics at the sensor node with the worst possible traffic conditions. In this example this is the node s_{10}. If the characteristics in this node are determined and the node is dimensioned to cope with them, all other nodes in the field (assuming homogeneity) are dimensioned properly as well.

To calculate the total traffic pattern, the algorithm described in Section 2.2 has to be used. First the output bound α^*_{40} of the leaf node s_{40} has to be calculated using (16), (17) and (3):

$$\alpha_{40} = \gamma_{p,0},\ \beta_{40} = \beta_{f,l} = \beta,\ \alpha^*_{40} = \alpha_{40} \oslash \beta_{40} = \gamma_{p,pl} \tag{18}$$

The output bound for node s_{40} is also the output bound for the other leaf nodes (e.g. $\alpha^*_{40} = \alpha^*_{41} = \alpha^*_{42} = \alpha^*_{43}$). Now the output bounds for the nodes one level higher in the tree can be calculated using equation (18), (16), (17) and (7):

$$\bar{\alpha}_{30} = \gamma_{p,0} + 3\alpha^*_{40} = \gamma_{p,0} + 3\gamma_{p,pl} = \gamma_{4p,3pl},\ \alpha^*_{30} = \bar{\alpha}_{30} \oslash \beta = \gamma_{4p,7pl} \tag{19}$$

[3] Values are taken from the TinyOS code (CC1000Const.h). The packet length is 36 bytes, the preamble length for 1% duty cycle is 2654 bytes.

The calculation can now be repeated until node s10 is reached: ...

$$\bar{\alpha}_{10} = \gamma_{p,0} + 2\alpha^*_{21} + \alpha^*_{20} = \gamma_{16p,34pl},\ \alpha^*_{10} = \bar{\alpha}_{10} \oslash \beta = \gamma_{16p,50pl} \tag{20}$$

After the arrival curve for node s_{10} is calculated, the worst case buffer requirements B_{10} and the information transfer delay D can be calculated according to equation (8) and (9):

$$B_{10} = v(\bar{\alpha}_{10}, \beta) = 50pl$$

$$D_{10} = h(\bar{\alpha}_{10}, \beta) = l + \frac{34pl}{f},\ D_{20} = h(\bar{\alpha}_{20}, \beta) = l + \frac{13pl}{f}$$

$$D_{30} = h(\bar{\alpha}_{30}, \beta) = l + \frac{3pl}{f},\ D_{40} = h(\bar{\alpha}_{40}, \beta) = l$$

$$D = D_{40} + D_{30} + D_{20} + D_{10} = 4l + \frac{50pl}{f}$$

Discussion Now, after all nodes are calculated, it is possible to determine specific values for the characteristics of interest for the given application scenario. Furthermore it is possible to evaluate how these factors influence each other. As mentioned above, due to the channel speed and the selected duty cycle, the effective maximum forwarding speed is $f = 258$[bit/s]. The arrival rate of packets cannot be higher than the maximum forwarding speed. A higher arrival rate would result in an infinite queueing of packets. Therefore the sensing rate must be set such that $16p \leq f$. In the following, the highest possible integral sensing rate is assumed: $p = \lfloor f/16 \rfloor = 16$[bit/s]. This first result already shows the limits of this specific sensor field regarding its maximum sensing frequency. Translated in TinyOS packets with a standard size of 36 byte, the result shows that each sensor can only send a packet every 18 seconds.

The backlog bound at node s_{10} is now given by: $B_{10} = 50pl = 876.8$[bit]. This result can be translated into TinyOS packets with the standard size of 36 byte. In this case, $\lceil 3.04 \rceil = 4$ packets must be stored in the worst case in node s_{10}. As a Mica-2 node provides per default only a buffer space of one, a node modification would be necessary to support the described scenario in the worst case. The maximum information transfer delay is given by: $D = 4l + \frac{51pl}{f} = 7.85$[s].

To improve the backlog bound and the information transfer delay, the duty cycle used in the nodes can be modified. Of course the improvements have to be paid in this case with a higher energy consumption in the nodes and thus a shorter network lifetime. If the duty cycle is set to 11.5%[4], a maximum packet forwarding rate of 0.54[packets/s] ($f = 2488$[bit/s]) can be achieved. The resulting delay for the packet forwarding is $l = T_1 + T_2 = 11 + 85 = 96$[ms]. Now the following is obtained: $B_{10} = 50pl = 76.8$. In this case now, only 1 TinyOS packets needs to be stored in node s_{10} even under worst case conditions. The information transfer delay is now given by: $D = 4l + \frac{51pl}{f} = 0.41$[s].

[4] A duty cycle value offered by the TinyOS code for the Mica-2.

5 Related Work

To the knowledge of the authors, there are currently no tools available that allow a systematic and general analysis of the worst case traffic conditions in a sensor network. However, for the analytical investigation of very specific properties of a sensor network or sensor node prior work exists. Some research work proposes models that have the ability to predict the reliability of messages. In [14] for example it is shown how the reliability can be calculated and controlled by adjusting the message forwarding schemes. Another research field is the prediction of traffic patterns in sensor networks so that data aggregation can be performed. In [15] for example it is described how traffic patterns at nodes can be predicted and modified such that an efficient data aggregation is possible.

In this paper, network calculus is tailored for the purpose of establishing a very generic framework to analyze traffic patterns in a wireless sensor network. To our knowledge only one publication [17] exists in the context of wireless sensor networks that uses network calculus as analytical tool. The latter paper deals with the very different problem of developing a theoretically sound congestion control in distributed sensor networks. The authors make some basic observations on their flow controller using network calculus but do not consider to actually model the sensor network itself using network calculus which has been the goal of the paper at hand.

6 Conclusion and Outlook

An analytical framework based on network calculus to dimension sensor networks has been presented. Using real world examples it has been shown how to apply the framework for practical problems in sensor network dimensioning. More specifically, it has been demonstrated how the various trade-offs and interdependencies between node power consumption, node buffer requirements and information transfer delay can be described using the sensor network calculus framework.

In this paper, the focus has been set on network dimensioning. However, the presented framework can be used as well for e.g. on-line admission control. In such a case, the sensor network calculus framework can for example be used to dynamically decide if or if not an interest can be served by the sensor network. A further application might be in topology control, where the respective decisions may also be driven by how the system behaves in the worst case.

References

1. M. Weiser. The computer for the 21st century. Scientific American, 265(3):94-104, September 1991.
2. R. Braden, D. Clark, and S. Shenker. Integrated Services in the Internet Architecture: an Overview. RFC 1633, June 1994.

3. K. Nichols, S. Blake, F. Baker, and D. Black. Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers. Proposed Standard RFC 2474, December 1998.
4. J.-Y. Le Boudec and P. Thiran. Network Calculus - A Theory of Deterministic Queueing Systems for the Internet. Springer, Lecture Notes in Computer Science, LNCS 2050, 2001.
5. Chipcon SmartRF CC1000 Data Sheet. Chipcon, www.chipcon.com. 2002.
6. Crossbow Technology INC. Smart Dust Training Seminar. Tutorial at the 1st IEEE European Workshop on Wireless Sensor Networks (EWSN2004), Berlin, Germany, January 2004.
7. J. Polastre. Design and Implementation of Wireless Sensor Networks for Habitat Monitoring. Masters Thesis. University of California, Berkeley. Berkeley, CA. May 23, 2003.
8. A. Barroso, J. Benson, T. Murphy, U. Roedig, C. Sreenan, S. Bellis, K. Delaney, B. OFlynn. The DSYS25 Sensor Platform. Demo at the SenSys2004. http://www.cs.ucc.ie/~ur1/pages/publications/files/sensys04barroso.pdf
9. P. Goyal, S. S. Lam, and H. Vin. Determining End-to-End Delay Bounds in Heterogeneous Networks. In Proceedings of Network and Operating System Support for Digital Audio and Video, 5th International Workshop, NOSSDAV95, Durham, New Hampshire, USA. Springer LNCS 1018, April 1995.
10. D. Stiliadis and A. Varma. Latency-Rate Servers: A General Model for Analysis of Traffic Scheduling Algorithms. IEEE/ACM Transactions on Networking, 6(5):611624, October 1998.
11. Alan Demers, Srinivasan Keshav, and Scott Shenker. Analysis and Simulation of a Fair Queueing Algorithm. In Proceedings of ACM SIGCOMM, pp. 3-12, August 1989.
12. B. Karp and H. T. Kung, "GPSR : greedy perimeter stateless routing for wireless networks," in Mobile Computing and Networking, pp. 243-254, 2000.
13. Crossbow Technology INC. Mica-2 Data Sheet. Crossbow,
14. http://www.xbow.com.
15. B. Deb, S. Bhatnagar and B. Nath. ReInForM: Reliable Information Forwarding using Multiple Paths in Sensor Networks. In Proceedings of the 28th Annual IEEE conference on Local Computer Networks (LCN 2003), Bonn, Germany, October 2003.
16. U. Roedig, A. Barroso, and C. J. Sreenan. Determination of Aggregation Points in Wireless Sensor Networks. In Proceedings of the 30th Euromicro Conference (EUROMICRO2004), Rennes, France, IEEE Computer Society Press, August 2004.
17. C. Intanagonwiwat, R. Govindan and D. Estrin. Directed Diffusion: A Scalable and Robust Communication Paradigm for Sensor Networks In Proceedings of the Sixth Annual International Conference on Mobile Computing and Networks (MobiCOM 2000), August 2000, Boston, Massachusetts.
18. J. Zhang, K. Premaratne and Peter H. Bauer. Resource Allocation and Congestion Control in Distributed Sensor Networks - a Network Calculus Approach. In Proceedings of Fifteenth International Symposium on Mathematical Theory of Networks and Systems, University of Notre Dame, August 12-16, 2002.