Sensor Network Calculus with Multiple Sinks

Jens B. Schmitt[1], Frank A. Zdarsky[1], Utz Roedig[2]

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

Abstract. Network calculus has been proposed and customized as framework for worst case analysis in wireless sensor networks in [1]. It has been demonstrated that this so-called sensor network calculus is an effective means to calculate maximum message transfer delays, maximum buffer requirements at the sensor nodes as well as lower bounds on duty cyles. Yet, so far only a single sink was accomodated for in the respective models. In this paper, we describe how sensor network calculus can be enhanced to also cover scenarios involving multiple sinks.

1 Introduction

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 that they can be viewed as an important part in the vision of ubiquitous/pervasive computing [2].

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. 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 [3] is a relatively new tool that allows worst case analysis of packet-switched communication networks. In [1] a framework for worst case analysis of wireless sensor networks based on network calculus is presented and called sensor network calculus. However, it can only be considered a first step towards a comprehensive worst case analysis framework for WSNs. One particular restriction is with respect to the number of sinks in the WSN: a single sink model is assumed. This is restrictive especially for larger scale WSNs. To have more sinks is beneficial for the information transfer delay as well as for the energy consumption of the sensor nodes since a sink can be reached faster respectively in less hops.

2 Sensor Network Calculus: The Single Sink Case

In this introductory section a single sink communication pattern 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 is explicitly neglected. 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. 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:

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

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

$$\bar{\alpha}_i = \alpha_i + \sum_{j=1}^{n_i} \alpha^*_{child(i,j)} \tag{2}$$

2.1 Maximum Sensing Rate Arrival Curve

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

$$\alpha_i(t) = p_i t = \gamma_{p_i,0}(t) \tag{3}$$

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. 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.

2.2 Rate-Latency Service Curve

Next, 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. More specific, the service curve depends on how the duty cycle and therefore the energy-efficiency goals are set.[1]

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.

A typical and well-known example of a service curve from traditional traffic control in a packet-switched network is

$$\beta_{R,T}(t) = R\,[t-T]^{+} \tag{4}$$

where the notation $[x]^{+}$ equals x if $x \geq 0$ and 0 otherwise. This is often also called a rate-latency service curve. 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 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. So, with some new parameters the following service curve at sensor node i is obtained:

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

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

[1] 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.

2.3 Network Flow Analysis

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:

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

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, $i = 1, \dots, N$, are given.
2. For all leaf nodes the output bound α_i^* can be calculated according to (6). 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 (6) 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 per node delay bounds D_i for each sensor node i can be calculated according to Theorem 3 (see Appendix):

$$D_i = h(\bar{\alpha}_i, \beta_i) = \sup_{s \geq 0} \{ \inf \{ \tau \geq 0 : \bar{\alpha}_i(s) \leq \beta_i(s + \tau) \} \} \tag{7}$$

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:

$$\bar{D}_i = \sum_{i \in P(i)} D_i \tag{8}$$

The maximum information transfer delay in the sensor network can then obviously be calculated as $\bar{D} = \max_{i=1,\dots,N} \bar{D}_i$. Note that this kind of analysis assumes FIFO scheduling at the sensor nodes which however should be the case in most practical cases.

3 Sensor Network Calculus: The Multiple Sink Case

The basic setting as well as the calculation procedure can be recovered from the single sink case. However, the network flow analysis is considerably more involved in the multiple sink case. When a specific flow of interest between a given source and sink shall be analysed, the effect of cross-traffic from other

Algorithm 1 Multiple Sink Network Flow Analysis

```
ComputeOutputBound(from node $i$, flows $\mathbb{F}$)
  forall pred($i$)
      $\alpha_i^{pred}$ += ComputeOutputBound(pred($i$), {flows to node $i$ from pred($i$)}$\cap\mathbb{F}$)
      $\alpha_{excl}$ += ComputeOutputBound(pred($i$), {flows to node $i$ from pred($i$)}$\setminus\mathbb{F}$)
  find $\theta^{opt}$ such that $\beta_i^{eff} = max_{\theta\geq 0}\left\{\beta_i\left(t\right) - \alpha_{excl}\left(t-\theta\right)\right\}$ is maximal
  return $\alpha_i^{pred} \oslash \beta_i^{eff}$

ComputeDelayBound (flow of interest $f$)
  ComputeOutputBound(sink of $f$, pred(sink of $f$))
  $delay^{total} = 0.0$
  forall nodes $i$ on the path from source to sink of $f$
      $delay_{total} += h \quad \alpha_i^{pred}, \beta_i^{eff}$
  return $delay_{total}$
```

flows to different sinks has to be taken into account. Here we can draw upon a known network calculus result telling us how to model the service curve of a FIFO node under cross-traffic from other flows [3]:

Theorem 1. *(FIFO Nodal Service Curve) Consider a FIFO node multiplexing two flows 1 and 2. Assume that the node guarantees a* strict *minimum service curve β to the aggregate of the two flows. Assume that flow 2 has α_2 as an arrival curve. Define the family of functions*

$$\beta_\theta^1\left(t\right) = \left[\beta\left(t\right) - \alpha_2\left(t-\theta\right)\right]^+ 1_{\{t>\theta\}}$$

Then for any $\theta \geq 0$ flow 1 is guaranteed a service β_θ^1 (if β_θ^1 is wide-sense increasing). Here, the indicator function $1_{\{cond\}}$ is defined as 1 if cond is true and 0 otherwise.

Theorem 1 now enables the derivation of sink-specific service curves for each node on the path between source and sink for a certain flow of interest. The detailed algorithms (yet still on a high level) to perform the network analysis for the multiple sink case are given in Algorithm 1. First we have to recursively compute the output bounds of all sensor nodes on the path from source to sink (starting the recursion at the sink). Here, we sum all traffic of flows joining the flow of interest and having the same sink as the flow of interest (that is the same step as in the single sink case). Next we sum all cross-traffic that joins the flow of interest but also leaves it again on the way to a sink different from that of the flow of interest. The reduction in the service curve guarantee by a given node on the path of the flow of interest is then calculated based on Theorem 1 using the cross-traffic (bounded by α_{excl}) and the nodal service curve β_i. This involves a search for the θ which gives an optimum effective service curve β_i^{eff} for the flow of interest (and its joining traffic from other sources towards the same sink). The mathematical details of how to find that optimum value are out of scope of this paper, but it may be mentioned that the actual calculation is simple to perfom. At the end of the output bound computation, the min-plus deconvolution of the overall traffic towards the sink of the flow of interest (bounded by α_i^{pred}) and the effective service curve of the given node β_i^{eff} is computed.

For each node on the path of the flow of interest the effective service curves β_i^{eff} and the bound on the overall traffic towards the sink of the flow of interest α_i^{pred} are stored. These are then used in the delay bound computation which now works its way up starting from the source of the flow of interest to its sink. At each node the horizontal deviation between α_i^{pred} and β_i^{eff} (i.e. the per-hop delay bound) is calculated and summed up resulting in the end-to-end delay for the flow of interest.

So now we can calculate the worst case delay characteristics for any flow of interest in the wireless sensor network at hand. Of course, if we wanted to find the absolute worst case delay in the wireless sensor network we could do the above calculation for each such flow of interest. Yet, not all of these calculations have to be done since some flows of interest would only be subflows of others and we could make the computation more efficient by only calculating the "longest" flows of interest.

4 Multiple Sink 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.

The goal of this section is to show how sensor network clalculus may be able to shed some light upon how the number of sinks affects the worst case message transfer delay in typical wireless sensor networks. The experimental setup is as follows: we assume a flat sqare of 100x100 m^2 on which the sensor nodes are randomly distributed (that means their x- and y-coordinates are chosen from a uniform random distribution over [0,100]). Each sensor has a transmission range of 20 meters. The routing from the sensors to the sinks is done based on shortest paths from sensors to sinks, each sensor node is associated with its nearest sink. The sinks are randomly chosen from the sensor nodes (which effectively releases them from being sensor nodes). Initially we create 100 nodes and then designate the respective number of sinks for the given experiment. The sensor node models we use mimic Mica-2 sensors [4] running TinyOS. In particular we assume a duty cyle of 1% which results in a latency of 1.096 s and a forwarding rate of 258 b/s. Furthermore, we assume a periodic sensing task: each sensor sends a 36 byte TinyOS packet every 30 seconds.

The results of the experiments with different number of sinks are shown in Figure 1. Here we show for 1, 5, and 10 sinks in the sensor field the worst case delay distribution over all possible flows of interest in the form of histograms where each bar gives the number of flows in intervals of duration 1 s. Note that for some flows there may not exist a finite delay (represented by the last bar in the histograms) since under worst case conditions the amount of incoming traffic of a node on the path of that flow may be higher than its forwarding rate. This

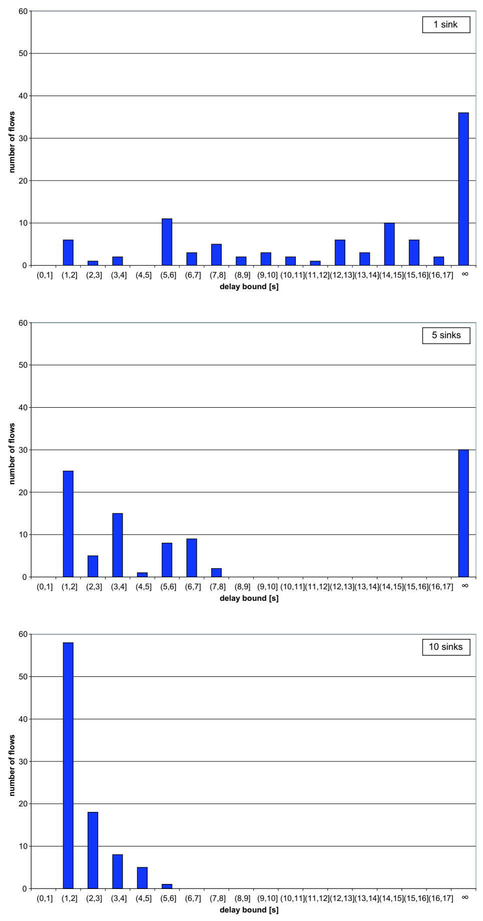

Fig. 1. Worst Case Delay Distribution for Different Numbers of Sinks.

is actually the case for 36 flows in the single sink scenario, but even when using 5 sinks there are still 30 flows with infinite delay bound. Only when we are using 10 sinks we receive finite delay bounds for all of the flows. Hence, we see that under these conditions sensor network calculus could help to correctly dimension the number of sinks in the wireless sensor network in order to control the message transfer delay performance of the network. In fact, in further experiments we found 6 sinks to be sufficient to ensure that no infinite delay bounds occured. It can be clearly seen from Figure 1 how the delay distribution improves with the number of sinks used in the sensor field. In fact, the average worst case delay improves from from 9.58 to 3.34 to 1.82 s for the 1, 5, and 10 sink scenario, respectively (not counting the flows with infinite delay bounds for the 1 and 5 sink case, of course). So again we see how sensor network calculus may advise us to find the number of sinks that we shall use in order to receive a certain message tranfer delay performance.

5 Related Work

In fact, there is not much work directly related to our proposal of using network calculus as a framework for the performance analysis of wireless sensor networks. Apart from our previous work in [1] and the work in [5] network calculus has so far not been used as tool in the context of wireless sensor networks. Actually, we hope that the community will eventually discover its great potential. The work described in [5] is a decidedly different issue of congestion control in sensor networks which however is also treated with the aid of network calculus.

6 Conclusion and Outlook

In summary, we believe that sensor networks will be used in the future for critical applications. In this case the sensor network must be properly dimensioned and controlled for all, even worst case, scenarios to ensure continuous and safe operation. The presented sensor network calculus framework is a first (promising) step towards such a proper dimensioning methodology of wireless sensor networks. In particular we have gone beyond [1] and showed how the multiple sink case in wireless sensor networks would be treated in the sensor network calculus framework.

There are many opportunities for future work. An incomplete list is the following: the stochastic nature of wireless communications needs to be incorporated into the models, mobility of sensor nodes and sinks should be accomodated, topology control and routing algorithms should be made aware of their effects on worst case characteristics, and so on. Apart from these issues the presented framework should also be validated by packet-level simulations in order to increase the fidelity in the predictive power of our models. Especially this last point deserves our immediate attention and is already currently under investigation.

References

1. Jens Schmitt and Utz Roedig. Sensor Network Calculus - A Framework for Worst Case Analysis. In Proc. of IEEE/ACM Int. Conf. on Distributed Computing in Sensor Systems (DCOSS'05), pages 141-154. Springer, LNCS 3560, June 2005.
2. M. Weiser. The computer for the 21st century. Scientific American, 265(3):94-104, September 1991.
3. J.-Y. Le Boudec and P. Thiran. Network Calculus - A Theory of Deterministic Queueing Systems for the Internet. Springer, LNCS 2050, 2001.
4. Crossbow Technology INC. Mica-2 Data Sheet.
5. J. Zhang, K. Premaratne and Peter H. Bauer. Resource Allocation and Congestion Control in Distributed Sensor Networks - a Network Calculus Approach. In Proc. of 15th Int. Symp. on Math. Theory of Networks and Systems, August 12-16, 2002.

Appendix: Background on Network Calculus[2]

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 and delay as well as 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, arrival and service curves 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 $R^* \geq R \otimes \beta$.*

[2] This appendix is not part of the paper but is provided for the convenience of reviewing.

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 2. *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:*

$$x(t) \leq \sup_{s \geq 0}\{\alpha(s) - \beta(s)\} = v(\alpha, \beta) \tag{9}$$

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

Theorem 3. *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:*

$$d(t) \leq \sup_{s \geq 0}\{\inf\{\tau \geq 0 : \alpha(s) \leq \beta(s + \tau)\}\} = h(\alpha, \beta) \tag{10}$$

$h(\alpha, \beta)$ 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 4. *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*

$$\alpha^* = \alpha \oslash \beta \geq \alpha \tag{11}$$

Theorem 5. *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:*

$$\beta = \beta_1 \otimes \beta_2 \tag{12}$$