Worst Case Modelling of Wireless Sensor Networks

Jens B. Schmitt
disco | Distributed Computer Systems Lab, University of Kaiserslautern, Germany
jschmitt@informatik.uni-kl.de

***Abstract*— At the current state of affairs it is hard to obtain a predictable performance from wireless sensor networks, not to mention performance guarantees. In particular, a widely accepted and established methodology for modelling the performance of wireless sensor networks is missing. In the last two years we have tried to make a step into the direction of an analytical framework for the performance modelling of wireless sensor networks based on the theory of network calculus, which we customized towards a so-called *sensor network calculus* [1]. We believe the sensor network calculus to be especially useful for applications which have timing requirements. Examples for this class of applications are factory control, nuclear power plant control, medical applications, and any alerting systems. In general, whenever the sensed input may necessitate immediate actions the sensor network calculus may be the way to go. In this paper we summarize these activities and discuss the open issues for such an analytical framework to be widely accepted.**

I. 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. Application areas for sensor networks might be production surveillance, traffic management, medical care, or military systems. 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, thus avoiding costly production outages. 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 performance analysis in sensor networks. Network calculus [2] 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*. This framework has further been extended towards random deployments [3] and the case of multiple sinks in [4]. The goal of this paper is to summarize these activities and show the usefulness of the sensor network calculus as well as opportunities for future work along this avenue.

II. SENSOR NETWORK CALCULUS: A BRIEF WALK-THROUGH

In this section we use the notation and the basic results provided in [1] (a summary of the most important notions of network calculus are given in the Appendix), furthermore a single sink communication pattern is assumed. 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 children's input, is

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

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

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

A. 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. This may either be due to the way the sensing unit is designed or due to a limitation on the sensing rate 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

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

Here, $\gamma_{r,b} = \begin{cases} rt+b & t>0 \\ 0 & t \le 0 \end{cases}$ denotes an affine arrival curve. This maximum sensing rate 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 performance bounds become.

B. 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.

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 idiosyncrasies 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 T nicely captures the characteristics induced by the application of a duty cycle concept, i.e., the sensor nodes periodically fall asleep for a certain amount of time if they are idle. Whenever the duty cycle approach is applied there is the chance that sensed data or data to be forwarded 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.

C. 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 (see Appendix):

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

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, $i = 1, \ldots, N$, 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 a basic network calculus result (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 $\bar{D} = \max_{i=1,\ldots,N} \bar{D}_i$. Note that this kind of analysis assumes FIFO scheduling at the sensor nodes, which however should be the case in most practical cases.

III. 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 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 curv*e 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*.

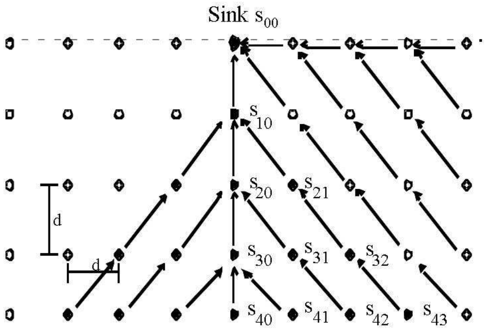

Fig. 1. Sensor Field with Grid Layout.

A. 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 analysed in which operation range a state of the art sensor node could be used to form the sensor field.

1) Topology and Routing: The sensor field is assumed to be a 9x9 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 (node s_{00}). 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 [6]. 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 analyses 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.

2) 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 (3) is used to model the upper bound of the sensing activity of each node in the sensor field. A homogeneous field is assumed, hence

$$\alpha_i(t) = pt = \gamma_{p,0}(t) \tag{9}$$

Each node additionally receives traffic from its child nodes according to the traffic pattern implied by the topology and the routing protocol (see Figure 1). Therefore, the arrival curve $\bar{\alpha}_i$ for the total input of a sensor node i is given by (2). Later it will be shown in detail how the relevant $\bar{\alpha}_i$ can be calculated.

3) 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.[1]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 (5) in Section II-B:

$$\beta_i(t) = \beta_{f,l}(t) = f(t-l)^+ = 258(t-1.096)^+[bit] \tag{10}$$

4) 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 II-C has to be used. First, the output bound α^*_{40} of the leaf node s_{40} has to be calculated using (9), (10) and (16):

$$\alpha_{40} = \gamma_{p,0},\ \beta_{40} = \beta_{f,l},\ \alpha^*_{40} = \alpha_{40} \oslash \beta_{40} = \gamma_{p,pl} \tag{11}$$

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 (11), (9), (10) and (6):

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

$$\alpha^*_{30} = \bar{\alpha}_{30} \oslash \beta = \gamma_{4p,7pl} \tag{12}$$

The calculation can now be repeated until node s_{10} is reached: ...

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

$$\alpha^*_{10} = \bar{\alpha}_{10} \oslash \beta = \gamma_{16p,50pl} \tag{13}$$

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 (14) and (7):

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

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

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

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

[1]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.

5) 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%[2], 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].

IV. Advanced Sensor Network Calculus

After the brief walk-through of the sensor network calculus basics and the illustrative example of its operation, we will discuss some of the more advanced techniques we have developed to further customize network calculus to the wireless sensor network setting as well as some of the applications of the framework we have proposed.

We have seen in the previous sections how the single sink communication pattern typically found in wireless sensor networks was used to iteratively work out the internal traffic flow bounds inside the network and use these to calculate delay bounds in an additive fashion. However, one of the strengths of network calculus is its powerful concatenation result, which allows in general to achieve better bounds when a tandem of servers is first min-plus convoluted to a single system compared to an additive analysis of the separate servers. This concatenation result is not directly applicable in a wireless sensor network scenario even when only considering the simple single sink case. Therefore, we have generalized the concatenation result for general feedforward networks in [5], introducing a principle called “pay multiplexing only once” which makes optimal use of sub-paths shared between flows and achieves improvements over the additive bounds, which may be on the order of magnitudes depending on the scenario. A further extension of the basic sensor network calculus, which we also describe in [5], is the integration of maximum service curves into the sensor network calculus, which allows to improve the bounds on the network-internal flows and thus in turn lowers the performance bounds, again often very considerably. All these techniques, among other general network calculus techniques, have been implemented in the so-called DISCO Network Calculator. As we believe that tool support is of great importance for a wide acceptance of the sensor network calculus we provide the DISCO Network Calculator in the public domain[3].

Apart from trying to push the sensor network calculus forward methodically, we have also illustrated how to apply it for several design and control issues in wireless sensor networks. In [1] we have shown how a buffer dimensioning of the sensor nodes may be performed based on the worst case analyses of sensor network calculus such that no information is lost due to buffer overflow inside the network. Furthermore, we also discussed in [1] how different choices of duty cycles affect the information transfer delay. In [3], we considered the case of a randomly deployed sensor network and how this further dimension of uncertainty can be factored into the sensor network calculus. In particular we discussed how constraints from topology control may be used to improve the performance bounds from the sensor network calculus. Thus, we proposed to guide topology control decisions based on the sensor network calculus models. In [4] we used the advanced sensor network calculus result discussed in the previous paragraph to investigate scenarios with multiple sinks. In particular we demonstrated how sensor network calculus can be used to dimension the number of sinks as well as their placement in the sensor field.

V. Open Issues and Future Work Items

While we believe the sensor network calculus to have potential, there are still many open issues and correspondingly opportunities for future work. One immediate issue arising from the use of a deterministic analytical framework is the question how to capture the inherently stochastic nature of wireless communications. Here, we plan to integrate lately upcoming stochastic extensions of network calculus [7], which however again need to be customized for the sensor network case. Another issue is how to take in-network processing as is frequently proposed for wireless sensor networks into account. In [8] we have proposed a network calculus that allows for the scaling of data flows. This development should enable modelling of typical in-network processing techniques as for example aggregation of information. Furthermore, it should also be possible to accomodate the mobility of sensor nodes and/or sinks. As in many scenarios this is a kind of

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

[3]See http://disco.informatik.uni-kl.de/content/Downloads.

controlled mobility there is hope to capture even this difficult characteristic of advanced wireless sensor network scenarios.

Apart from these fundamental issues for the sensor network calculus, it is also important to demonstrate its usefulness in further applications. At the moment we design a task admission control scheme based on sensor network calculus for sensor networks that may have several concurrent tasks. Another work item could be a scheme where sleeping nodes are activated such that certain performance bounds can still be satisfied. 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] J.-Y. Le Boudec and P. Thiran. Network Calculus - A Theory of Deterministic Queueing Systems for the Internet. Springer, LNCS 2050, 2001.

[3] Jens Schmitt and Utz Roedig. Worst Case Dimensioning of Wireless Sensor Networks under Uncertain Topologies. In Proc. 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.

[4] Jens B. Schmitt, Frank A. Zdarsky, and Utz Roedig. Sensor Network Calculus with Multiple Sinks. In Proc. of IFIP Networking 2006, Workshop on Performance Control in Wireless Sensor Networks, Coimbra, Portugal. Springer LNCS, May 2006.

[5] 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.

[6] B. Karp and H. T. Kung, "GPSR : greedy perimeter stateless routing for wireless networks," in Mobile Computing and Networking, pp. 243-254, 2000.

[7] Yuming Jiang. A Basic Stochastic Network Calculus. In Proc. ACM SIGCOMM 2006, Pisa, Italy, September 2006.

[8] Markus Fidler and Jens B. Schmitt. On the Way to a Distributed Systems Calculus: An End-to-End Network Calculus with Data Scaling. In Proc. ACM SIGMETRICS/Performance 2006 (SIGMETRICS'06), St. Malo, France, June 2006.

APPENDIX: BACKGROUND ON NETWORK CALCULUS

Network calculus is *the* tool to analyse 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. 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.

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 $\alpha(t) = 0$ 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

$$x(t) \leq \sup_{s \geq 0} \{\alpha(s) - \beta(s)\} = v(\alpha, \beta) \tag{14}$$

$v(\alpha, \beta)$ 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

$$d(t) \leq \sup_{s \geq 0} \{\inf\{\tau \geq 0 : \alpha(s) \leq \beta(s+\tau)\}\} = h(\alpha, \beta) \tag{15}$$

$v(\alpha, \beta)$ is also often called the vertical 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

$$\alpha^* = \alpha \oslash \beta \geq \alpha \tag{16}$$

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:

$$\beta = \beta_1 \otimes \beta_2 \tag{17}$$