Deterministic Worst-case Performance Analysis for Wireless Sensor Networks

Huimin She[1,2], Zhonghai Lu, Axel Jantsch, Li-Rong Zheng
[1]Dept. of Electronic, Computer and Software Systems
Royal Institute of Technology (KTH), Sweden
{huimin, zhonghai, axel, lirong}@kth.se

Dian Zhou
[2]ASIC & System State Key Lab.,
Dept. of Microelectronics, Fudan University, China
zhoud@fudan.edu.cn

***Abstract*—Dimensioning wireless sensor networks requires formal methods to guarantee network performance and cost in any conditions. Based on network calculus, this paper presents a deterministic analysis method for evaluating the worst-case performance and buffer cost of sensor networks. To this end, we introduce three general traffic flow operators and derive their delay and buffer bounds. These operators are general because they can be used in combination to model any complex traffic flowing scenarios in sensor networks. Furthermore, our method integrates variable duty cycle to allow the sensor nodes to operate at lower rates thus saving power. Moreover, it incorporates traffic splitting mechanisms in order to balance network workload and nodes' buffers. To show how our method applies to real applications, we conduct a case study on a fresh food tracking application, which monitors the food freshness in real-time. The experimental results demonstrate that our method can be either used to perform network planning before deployment, or to conduct network reconfiguration after deployment.**

I. Introduction

As advances in wireless communications and electronics, wireless sensor network (WSN) has become a promising technology with a wide range of applications, such as health care, supply chain management, structural monitoring, and military support [1]. In most of the applications, it is essential to ensure that the performance of sensor networks is predictable even in the worst case.

Recently, network calculus has been developed for worst-case performance analysis in packet switching networks [2]. With network calculus, some fundamental properties of packet-switched networks, such as delay bound and backlog bound, can be studied. Jens et al. [3] [4] extended this theory to sensor network calculus, which can be used as a tool for worst case traffic analysis in sensor networks. In [5], Anis et al. proposed a methodology for the modeling and worst-case dimensioning of cluster-tree sensor networks. In this paper, we also apply network calculus to examine the worst-case performance of sensor networks. However, our work differs from the previous work and makes significant improvement in the following aspects. Firstly, variable duty cycle (see section 2-C) is considered in our approach, thus providing a facility to make compromises between latency and power consumption according to application requirements. In [7], Sang et al. demonstrated that both energy saving and high performance can be achieved by conducting variable duty-cycle operations in media access control protocols. Secondly, we applied the network calculus theory to analyze three general traffic flow operators, which can be used to characterize any complex traffic flowing scenarios. The method can be applied to networks with any topologies as long as there is no loop. Thirdly, traffic splitting routing can be applied in this method. With traffic splitting mechanisms, a traffic flow is split into several sub-flows and each one is sent to the destination across different paths. Traffic splitting can be useful in improving the bandwidth efficiency, mitigating congestion, and increasing delivery reliability [8]. In addition, in [3] [4] [5], their works are based on a common assumption that the service rate is constantly bigger than the input data rate. The assumption may not always be reasonable in sensor networks since nodes should allow different operation rates for the best of power saving without compromising performance. Our method can incorporate the rate adjustment in nodes.

We have described the traffic splitting mechanisms in our previous work [6], from which we borrow many notations used in this paper. However, this paper differs from [6] on the following aspects. In [6], the work mainly focused on analyzing traffic splitting mechanisms; in this paper our main contribution is proposing a deterministic analysis method integrating variable duty cycle and three general traffic flow operators. Moreover, the topology of sensor network examined in [6] is a regular 2D mesh. In this paper, this limitation is lifted.

In this paper, we apply and extend the network calculus theory to the worst-case performance analysis of sensor networks. We introduce three general traffic flow operators and derive their characteristics using network calculus theory. Based on the traffic flow operators, we present a deterministic performance analysis method which integrates variable duty cycle operations and traffic splitting mechanisms. In order to show how the analysis method works, a case study of designing a sensor network for monitoring food freshness in real-time (see details in section 4-A) is conducted. The numerical results indicate that variable duty cycle operations and traffic splitting mechanisms have significant effects on improving the performance of sensor networks. Thus, requirements of different applications can be satisfied by selecting appropriate network parameters such as duty cycle, work period and splitting coefficient. Therefore, the analysis method provides a way for a sensor network designer to perform network planning prior to deployment, as well as to reconfigure network after design.

The rest of this paper is organized as follows: section 2 introduces the system model of sensor networks and basic knowledge of network calculus theory. In section 3, three general traffic flow operators and the deterministic analysis method are presented. We present the wireless sensor network for fresh food tracking and give numerical results in section 4. Finally, conclusions are drawn in section 5.

II. MODELS

A. Sensor Netowrk System Model

We consider a static wireless sensor network consisting of multiple sensor nodes and one sink node. These sensors are randomly scattered in a field that needs to be sensed (Fig. 1). Sensor nodes periodically send their acquired data to the sink through multi-hop routing. A sensor node has the ability to sensing the environment and generating messages, as well as relaying messages for other nodes.

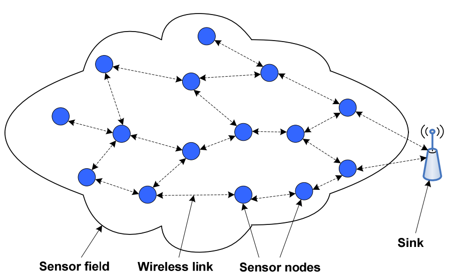

Fig. 1 A typical sensor network

In sensor networks, there are typically two kinds of traffic flows, which are upstream traffic flows (from sensor nodes to the sink) and downstream traffic flows (from the sink to a sensor node). Typically, critical messages are sent from sensors to the sink, i.e. upstream. The methods used to analyze upstream traffic flows and downstream traffic flows are similar. Therefore, our efforts concentrate on analyzing upstream traffic flows.

B. Traffic Model

To characterize the traffic generated by the sensor nodes, we model the arrival flow at a node using its cumulative traffic *R(t)*, defined as the number of bits coming from the flow in time interval [0, t] (*R*(0)=0). We assume that the cumulative traffic flow *R(t)* is constrained by a wide-sense increasing function α(t) , that is,

$$R(t) - R(s) \le \alpha(t-s); \forall t \ge 0, t \ge s \tag{1}$$

α(t) is called the *arrival curve* of *R(t)* [2] (Fig. 2). In this paper, we assume an *affine* arrival curve for all the sensor nodes, which is defined as $\alpha(t) = \rho \cdot t + \sigma$, where σ and ρ represent the burst tolerance (in units of data) and the rate (in units of data per unit time), respectively. Having α(t) as an arrival curve allows a source to send σ bits at once, but not more than ρ bits/s over the long run. Similarly, the output flow from a node could also be modeled by a cumulative function denoted by $R^*(t)$, which is defined as the traffic departing from the node in time interval [0, t]. The relation between the input flow *R(t)* and output flow $R^*(t)$ is expressed as (Fig. 2),

$$R^*(t) \ge \inf_{s \le t} (R(s) + \beta(t-s)); \forall t \ge 0 \tag{2}$$

where β(t) is defined as the *service curve* [2] provided by the sensor node, which is a wide sense increasing function with β(0) = 0.

Assume an arrival flow *R(t)*, constrained by arrival curve α(t), traverses a sensor node that offers a service curve β(t). Then, the delay bound *D(t)*, buffer bound *B(t)*, and output flow $R^*(t)$ can be derived according to the following lemmas. The proofs of these lemmas can be found in [2].

Lemma 1. Delay bound

$$D(t) \le \inf\{\tau \ge 0 : \alpha(t) \le \beta(t+\tau)\} \tag{3}$$

Lemma 2. Backlog bound

$$B(t) \le \sup_{s \ge 0}\{\alpha(s) - \beta(s)\} \tag{4}$$

Lemma 3. Output flow: The output flow $R^*(t)$ is constrained by the arrival curve,

$$\alpha^*(t) = \sup_{s \ge 0}\{\alpha(t+s) - \beta(s)\} \tag{5}$$

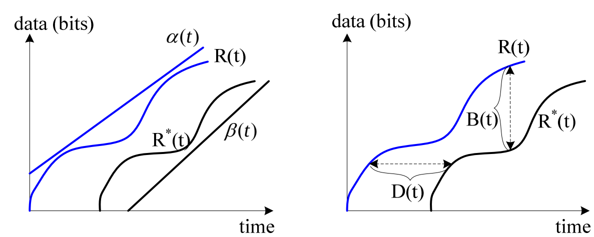

Fig. 2 Traffic Model: 1) The relationship between input traffic flow *R(t)*, output traffic flow $R^*(t)$, arrive curve α(t) and service curve β(t); 2) Delay bound and backlog bound.

The backlog is the amount of bits that are held inside the sensor node. The required buffer size of a sensor node is determined by the maximum backlog. The delay at time *t* is the time that would be experienced by a bit arriving at time *t* if all bits received before it are served before it. Graphically, the delay bound and backlog bound are the maximum horizontal and deviation distance between arrival curve α(t) and service curve β(t), respectively (Fig. 2).

C. Variable Duty Cycle

Waking up the nodes all the time is impossible in wireless sensor network since merely turning on the radio will soon deplete the node energy. To save energy, all the existing sensor networks employ a low duty-cycle operation with a periodic sleep and wakeup. In this paper, we also assume sensor nodes have two work modes which are *active* mode and *sleep* mode. Let the work period of all the sensor nodes be *T*, and duty cycle of sensor node i be λ. *Duty cycle* is defined as the percentage of time that the sensor node is active in a period. It can be expressed as a ratio or a percentage. For example, a sensor node with a 1 second work period, which consists of 0.1s active time and 0.9s sleep time, is said to have a duty cycle of 0.1 or 10%. Assume the time for node i to process the packets is τ. Further, let C denote the achievable link capacity. Therefore, sensor node i provides a *rate-latency service curve* $\beta(t)$, which is defined as,

$$\beta(t) = \lambda C[t - ((1-\lambda)T + \tau)]^{+}, \tag{6}$$

where λC and $(1-\lambda)T + \tau$ denote the service rate and delay, respectively. The expression $[x]^{+}$ is equal to x when $x>0$, and 0 otherwise.

III. A Deterministic Analysis Method

In this section, we present a deterministic method for worst-case performance analysis of sensor networks. The method is designed to analyze the *delay bound*, *backlog bound*, and delivery capability which is measured by *data delivery ratio* (see examples in section 4-B).

A. Analysis of Traffic Flow Operators

We defined three kinds of traffic flow operators: *traffic passing* operator, *traffic merging* operator, and *traffic splitting* operator (Fig. 3). These operators are general and can be used to describe any combined traffic flowing scenarios.

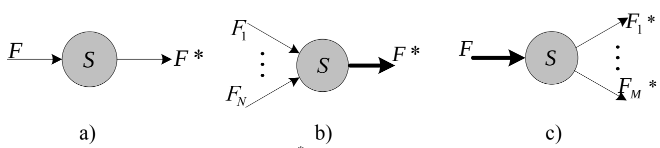

Fig. 3 Traffic flow operators. F, F^{*}, and S denote the input flow, output flow, sensor node, respectively. a) Traffic passing: one input flow and one output flow; b) Traffic merging: multiple input flows and one output flow; c) Traffic splitting: one input flow and multiple output flows.

1). Traffic passing

For the traffic passing operator, the sensor node has one input link and one output link (Fig. 3-a). As we mentioned above, the input traffic flow is constrained by arrival curve $\alpha(t) = \rho \cdot t + \sigma$, and service curve is defined as equation (6). Based on Lemma 1, 2, 3, we derived the delay bound, backlog bound, and output flow, respectively [2].

The delay bound is expressed as,

$$D = \frac{\sigma}{\lambda C} + (1-\lambda)T + \tau \tag{7}$$

And the backlog bound is,

$$B = \sigma + \rho(1-\lambda)T + \rho\tau \tag{8}$$

The output flow is constrained by,

$$\theta(t) = \rho' t + (\sigma + \rho(1-\lambda)T + \rho\tau), \tag{9}$$

where $\rho' = \min(\rho, \lambda C)$.

In the case that $\lambda C < \rho$, the backlog will increase endlessly if the input traffic flows into the node continuously. To avoid this, measures should be taken either to increase the service rate or constrain the input data rate. Otherwise, packets will be dropped when the buffer is full. From the expression of the service curve, we can see that the service rate can be increased by increasing the duty cycle.

2). Traffic merging

For the traffic merging operator, multiple traffic flows merge into one traffic flow at the sensor node (Fig. 3-b). In this case, it is important that a service discipline should be applied to allocate the bandwidth. The service disciplines are used to control the order in which packets are served, and determine how packets from different connections interact with each other. In [10], Zhang described several service disciplines for packet-switching networks, for example, Delay Earliest-Due-Date, Virtual Clock, Fair Queuing. However, different service disciplines fit for different applications. The service discipline for sensor networks should be as simple as possible since the hardware resource in a sensor node is very limited. Therefore, we take the following two disciplines for bandwidth allocation in sensor networks. The first one is called *rate-proportional allocation* strategy, and the other one is called *weight-proportional allocation* strategy. In rate-proportional allocation, the bandwidth is allocated proportional to the data rate of each flow; while in weight-proportional allocation, each flow is assigned a weight value w_i, and the bandwidth is allocated according to the weight values. In fact, the previous allocation strategy can be regarded as a special case of the latter when all the w_i equals 1.

As shown in Fig. 3-b, we assume there are N input traffic flows, each of which is denoted as F_i. The output is an ensemble of traffic flows. Let F_i^{*} denote the output flow corresponding to F_i, and C_i denote the bandwidth allocated to the traffic flow F_i. Assume the arrival curve and service curve are the same as those in section 3.1-A. Then, C_i in the two allocation strategies is calculated as expression (10). Based on Lemma 1, 2, 3, the delay bound, backlog bound, and output flow are derived as expression (11), (12), (13), respectively [2].

$$C_i = \frac{w_i \rho_i}{\sum_{j=1}^{N} w_j \rho_j} \lambda C \tag{10}$$

The delay bound of flow i is,

$$D_i = \frac{\sigma_i}{C_i} + (1-\lambda)T + \tau \tag{11}$$

And the backlog bound is expressed by,

$$B = \sum_{i=1}^{N} \sigma_i + ((1-\lambda)T + \tau)\sum_{i=1}^{N} \rho_i \tag{12}$$

The output flow F_i^{*} is constrained by,

$$\theta_i(t) = C_i t + \sigma_i + \rho_i(1-\lambda)T + \rho_i\tau \tag{13}$$

3). Traffic splitting

In order to balance network workload, a traffic flow may be split into multiple flows as shown in Fig. 3-c. Let the splitting factor be γ_j ($j = 1...M$) and $\sum \gamma_j = 1$, where M denotes the number of output paths. We assume the node has infinite input and output capacity. The arrival curve and service curve are the same as those in section 2.2-B. Then, we derived the delay bound, backlog bound, and output flow according to Lemma 1, 2, and 3 [2].

The delay bound is,

$$D_j = \frac{\gamma_j \sigma}{\lambda C} + (1-\lambda)T + \tau \quad (14)$$

And the backlog bound is,

$$B = \sigma + \rho(1-\lambda)T + \rho\tau \quad (15)$$

Output traffic flow F_j^* is constrained by,

$$\theta_j = \min(\gamma_j \rho, \lambda C)t + \gamma_j(\sigma + \rho(1-\lambda) + \rho\tau) \quad (16)$$

For the traffic splitting operator, there could be cases that $\rho > \lambda MC$, which means the output bandwidth can not satisfy the requirements. To avoid this, either the input data rate should be constrained or the service rate should be enhanced. Otherwise, the data loss rate will be increasing.

B. The Deterministic Analysis Method

In this section, we present the deterministic analysis method as a whole. It works as follows:

1. According to the topology of the sensor network and the routing algorithm, obtain the routing paths of each traffic flow.
2. Based on the general traffic flow operators and analysis methods proposed in section 3.1, construct traffic flowing scenarios using the three general operators. Then compute the output flow, delay bound and backlog bound for each traffic flow starting from the source node.
3. Calculate the end-to-end delay bound. There are two ways to compute the end-to-end delay bound. The first method is summing up the per-hop delay together. The other method was proposed by Lenzini et al. [11]. The main idea of this method is to derive an equivalent service curve for a given traffic flow based on the network calculus theory. And then the end-to-end delay bound is calculated using the equivalent service curve. Both approaches can be applied in our analysis method.

C. An Example

We show how the deterministic analysis method works through an example. Assume there are two flows F_1 and F_2, which are sourced from node *a* and *b*, respectively (Fig. 4). F_1^c and F_2^c denote the corresponding output traffic flow of F_1 and F_2 at node *c*. Assume the traffic models are $F_1 \sim (\sigma_1, \rho_1)$ and $F_2 \sim (\sigma_2, \rho_2)$, where σ_1 and σ_2 describe the burstiness, and ρ_1 ρ_2 denote the data rate [9]. Let the duty cycle and work period of node *i* (*i*=*a*…*g*) be λ_i and *T*, respectively. Let the link capacity be *C*.

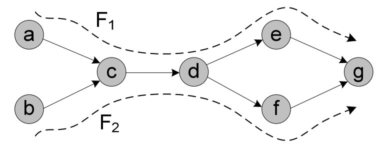

Fig. 4 An example of traffic flows

At node *c*, there are two input links and one output link. Then, the output flow, delay bound and backlog bound can be calculated according to the method described in section 3-A-2. Assume the allocation strategy is rate-proportional allocation strategy. Then,

$$F_1^c \sim (\sigma_1 + \rho_1(1-\lambda_c)T + \tau, \lambda_c C\rho_1/(\rho_1+\rho_2)) \quad (17)$$

$$F_2^c \sim (\sigma_2 + \rho_2(1-\lambda_c)T + \tau, \lambda_c C\rho_2/(\rho_1+\rho_2)) \quad (18)$$

$$D_1^c = \sigma_1(\rho_1+\rho_2)/(\rho_1\lambda_c C) + (1-\lambda_c)T + \tau \quad (19)$$

$$D_2^c = \sigma_2(\rho_1+\rho_2)/(\rho_2\lambda_c C) + (1-\lambda_c)T + \tau \quad (20)$$

And the backlog of node *c* is,

$$B_c = \sigma_1 + \sigma_2 + (\rho_1+\rho_2)((1-\lambda_c)T + \tau) \quad (21)$$

At node *d*, there are one input link and two output links. All the calculation can follow the method described in section 3-A-3. Use the same method, at node *e* and *f*, the output flows, delay bound and backlog bound can be calculated recursively according to the method described in section 3-A-1.

After the above calculation, the burstiness and data rate of the output of flow F_1 at the sink can be derived as expressions (22) and (23), respectively.

$$\sigma_1^* = \sigma_1 + \rho_1 T[(1-\lambda_c) + (1-\lambda_d) + (1-\lambda_e) + 3\tau] \quad (22)$$

$$\rho_1^* = \min(\lambda_e C, \lambda_d C, F_1^c(\rho)) \quad (23)$$

where $F_1^c(\rho)$ denotes the data rate of flow F_1 at node *c*.

The end-to-end delay bound can be calculated by adding the individual delay at each node together. For flow F_2, the results can be computed using the same method. From the results, we can see that the output data rate is mainly limited by the bottleneck link. Therefore, if the delay bound and backlog bound can not satisfy the requirements of applications, the duty cycle needs to be dynamically adjusted.

IV. Application and Numerical Results

A. Wireless Sensor Network for Fresh Food Tracking

In European market, approximately 10% of the whole cargo of fruits and vegetables coming from different parts of world is deteriorated during the transportation process. This leads to a loss of billions of dollars per year [12]. With the rapid development of sensor network techniques, the loss can be mitigated by deploying a sensor network to track the freshness status of these kinds of food in real-time.

In the scenario of real-time fresh food (e.g. meat, vegetable, fruits) tracking, sensors are deployed in the boxes filled with fruits, vegetables and meat in a truck carriage (Fig. 5). Since the possible causes of food deterioration are microbiological infestation and improper environmental condition, four kinds of sensors can be used in our application, which are *humidity sensor, temperature sensor,* CO_2 *sensor,* and O_2 *sensor*. These sensors are responsible for collecting the corresponding information of food. All the data collected by sensors are sent to a base station, which is put on the top of the truck. The base station then transmits the data to a remote server through GPRS networks and Internet. Thus an expert at the remote server side can read and analyze the data in real-time. If something is wrong or abnormal actions have happened, he can send instructions to the base station to take measures, such as lowering the temperature of the cooling system or sprinkling water onto fresh vegetables and fruits, to protect the food from becoming deteriorated. In addition, there is a wired connection between the base station and the driver monitor. So the driver can also read the information collected by the network and take proper measures if necessary. The size of sensor networks applied in this application depends on the size of trucks. For small trucks, a 2-hop or 3-hop sensor network is enough. While for large trucks, a network of more hops is needed.

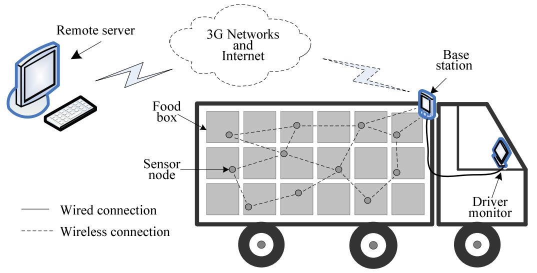

Fig. 5 A sensor network for real-time fresh food tracking

B. Numerical Experiments and Results

We have realized our deterministic analysis method using Matlab. The parameters used in the numerical experiment are as follows. We assume a sensor network was generated by randomly putting a number of sensors in the food boxes, which are located in a truck carriage (Fig. 5). The base station acts as the sink. Therefore, a wireless sensor network with an irregular topology is set up. According to Mica2 mote[1] [13], we assume the link capacity C is 38.4 *kbps* and work period T is *1.096s* [4]. We assume the packet size is 288 bits. The standard reporting frequency of each sensor is assumed to be 0.1 Hz, i.e. the sensor node sends one packet in every ten seconds, leading to a date rate of 28.8 bits/s. The burst size is assumed to be the amount of data generated in two seconds. In the experiments, the traffic load is changed by varying the reporting frequency from 0.1 Hz to 1 Hz. Moreover, we assume there is no collision in the network since the effect of collision is independent of traffic merging and splitting.

[1] The Mica2 mote is a mote module used for low-power wireless sensor networks (see http://www.xbow.com)

Note that at a sensor node, its input data rate can be higher than its service rate due to a lower duty cycle configuration. If this happens, data loss may occur when the backlog buffer is full. Apparently, data loss is a big concern. To capture this in our experiments, we define *data delivery ratio* as the amount of data received by the sink versus that of data sent by the sources.

To study how duty cycle impacts the performance of sensor networks, we conduct several numerical experiments. The number of nodes is 30. Fig. 6 shows that the packet delivery ratio decreases with traffic load increasing. And the packet delivery ratio can be enhanced by increasing the duty cycle. In Fig. 7, we can see that the end-to-end worst-case delay increases when the traffic load increases. With the same traffic load, the delay decreases with duty cycle increasing. Therefore, when the data delivery ratio or the worst-case delay can not meet the requirement of applications, the duty cycle needs to be increased. Meanwhile, the power consumption can be more important than delivery ratio and delay. In these cases, low duty cycle operations can be taken to save energy.

As we mentioned in previous sections, traffic splitting mechanisms play an important role in load balancing. In order to study the efficiency of traffic splitting strategies, we compared the data delivery ratio and average per-hop delay in different splitting strategies, which are *no traffic splitting* (NoTS), *averagely traffic splitting* (AvTS) and *randomly traffic splitting* (RaTS). Averagely traffic splitting means that the traffic is averagely split at each sensor nodes, while randomly traffic splitting means that the traffic is split with random probabilities. In the following simulations, the duty cycle of each sensor is 0.04. Fig.8 shows that when the traffic load is bigger than 0.5, the data delivery ratio drops dramatically when the traffic is not split. Compared with that in NoTS, the data delivery ratios in AvTS and RaTS are enhanced by 17.1% and 14.8%, respectively. In Fig. 9, the average per-hop delay in AvTS and RaTS is lower than that in non-traffic-splitting cases, with improvement 9.65% and 7.81%, respectively.

To show the performance of different network sizes, we devise the following experiments. The traffic load is set to be 0.6, and the duty cycle is set to be 0.04. Fig. 10 and Fig. 11 show the delivery ratio and the worst-case delay scale with the number of sensors, respectively. In Fig. 10, we can see that the delivery ratio decreases as the number of sensors increases. However, by adopting the traffic splitting strategies AvTS and RaTS, the data delivery ratio is enhanced by 20.5% and 17.7%, respectively. Moreover, Fig. 11 shows that the end-to-end worst-case delay is reduced by 28.8% and 25.2% in AvTS and RaTS, respectively.

To show the backlog variation with duty cycles and traffic splitting methods, we conduct two experiments. The traffic load is 0.5 for both figures, and the duty cycle is 0.08 for Fig. 13. B_min, B_ave, and B_max denote the minimum backlog, average backlog, and maximum backlog, respectively. These values are obtained from backlog bounds at all sensor nodes. In Fig. 12, we can see that the backlogs do not reduce much when duty cycle increases from 0.04 to 0.15. However, the average backlog is much smaller than the maximum backlog with the same duty cycle. In this example, adjusting duty cycle has smaller effect on reducing the backlogs. But by applying traffic splitting strategies, the maximum and average backlogs can be greatly reduced.

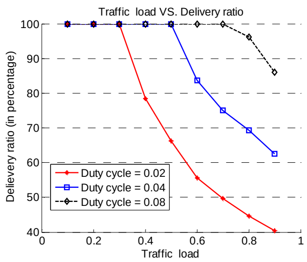

Fig. 6 Data delivery ratio with various duty cycles

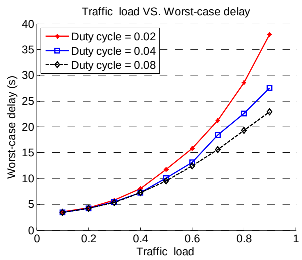

Fig. 7 Worst-case delay with various duty cycles

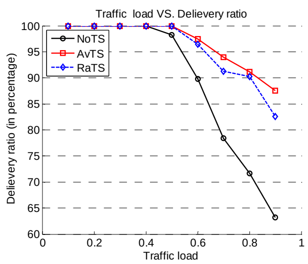

Fig. 8 Data delivery ratio in various split methods

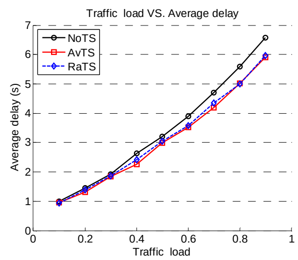

Fig. 9 Average delay in different split methods

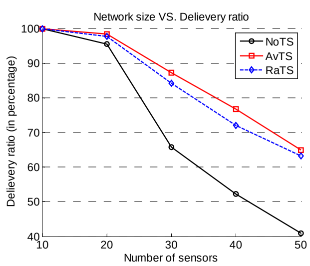

Fig. 10 Data delivery ratio vs. network sizes

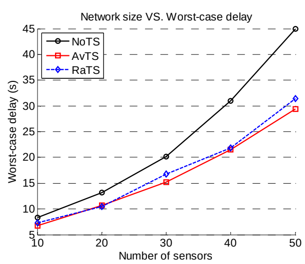

Fig. 11 End-to-end worst-case delay vs. network sizes

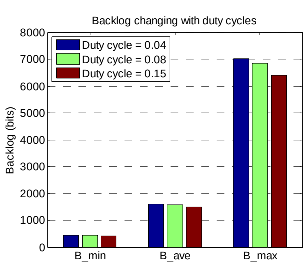

Fig. 12 Backlogs with different duty cycles

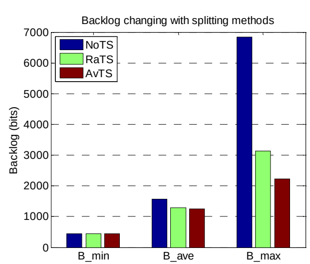

Fig. 13 Backlogs in different splitting strategies

V. Conclusions

In this work, we proposed a deterministic method for worst-case performance analysis in wireless sensor networks. Three general traffic flow operators are defined to model any traffic flowing scenarios. Based on the results from the basic operators, we presented the deterministic method, which adopts variable duty cycle operations and traffic splitting strategies. The method is applied to analyze the performance of the sensor network for fresh food tracking. With the numerical results, we show that (1) increasing duty cycle and splitting traffic flows can improve data delivery ratio and reduce the delay; and (2) variable duty cycle operations have less significant effect on reducing the max and average backlogs, while traffic splitting mechanisms can largely reduce the maximum backlog and average backlog. Hence, by adjusting the duty cycle and traffic splitting mechanisms, the performance requirements (such as delay, backlog and data delivery ratio) of different applications can be satisfied. Therefore, our proposed method not only provides an effective way for a designer to estimate the worst-case performance of sensor networks, but also can be used as a tool for network reconfiguration after design.

For the future work, we intend to integrate fault tolerance into the analysis method. Furthermore, it is interesting to explore the optimized design space with given buffer sizes, performance requirements, energy constraints, and service strategies.

Acknowledgment

This work is supported by VINN Excellence iPack Center at Royal Institute of Technology (KTH), Sweden.

References

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, “A survey on sensor networks”, IEEE Comm. Magazine, pp. 102-114, 2002
[2] J. –Y. Le Boudec, and P. Thiran, “Network Calculus: A Theory of Deterministic Queuing Systems for the Internet”, Springer, LNCS 2050, 2004
[3] J. B. Schmitt, and Utz Roedig, “Sensor network calculus – A framework for worst case analysis”, IEEE/ACM International conference on Distributed Computing in Sensor Systems (DCOSS’05), USA, 2005
[4] J. B. Schmitt, F. A. Zdarsky, and Utz Roedig, “Sensor network calculus with multiple sinks”, In proc. of IFIP networking, workshop on performance control in wireless sensor networks, Portugal, 2006
[5] Anis Koubaa, M. Alves, and E. Tovar, “Modeling and worst-case dimensioning of cluster-tree wireless sensor networks”, the 27th IEEE International Real-Time Systems Symposium (RTSS’06), Brazil, 2006
[6] H. She, Z. Lu, A. Jantsch, L.-R. Zheng, D. Zhou, “Traffic splitting with network calculus for mesh sensor networks”, the 2007 International Workshop on Wireless Ad Hoc, Mesh and Sensor Networks (WAMSNet’07), Korea, 2007
[7] Sang Hoon Lee, Joon Ho Park, and Lynn Choi, “AMAC: Traffic-adaptive sensor network MAC protocol through variable duty-cycle operations”, In Proceedings of IEEE International Conference on Communications (ICC’07), Scotland, 2007
[8] S.-J. Lee, M. Gerla, “Split multipath routing with maximally disjoint paths in ad hoc networks”, IEEE International Conference on Communications (ICC’01), 2001
[9] R. Cruz, “A calculus for network delay, part I: Network elements in isolation”, IEEE Trans. on Info. Theory, vol. 37, pp. 114-121, 1991
[10] H. Zhang, “Service disciplines for guaranteed performance service in packet switching networks”, Proceedings of the IEEE, 1995
[11] L. Lenzini, L. Martorini, E. Mingozzi, and G. Stea, “Tight end-to-end per-flow delay bounds in FIFO multiplexing sink-tree networks”, Performance Evaluation Journal(Elsevier), vol.63, pp 956-987, 2006
[12] Billerud AB, “The Billerud book of maths”, http://www.billerud.com/
[13] Crossbow Technology Inc., “MICA2 Datasheet” http://www.xbow.com