Fast Performance Simulation for Gossip-based Wireless Sensor Networks

Miloš Blagojević [1,2], Marc Geilen[1], Twan Basten[1,2], Majid Nabi[1], and Teun Hendriks [2]

[1] EINDHOVEN UNIVERSITY OF TECHNOLOGY, The Netherlands
[2] TNO-ESI, The Netherlands

Gossip-based Wireless Sensor Networks (GWSN) are complex systems of inherently random nature. Planning and designing GWSN requires a fast and adequately accurate mechanism to estimate system performance. As a first contribution, we propose a performance analysis technique that simulates the gossip-based propagation of each single piece of data in isolation. This technique applies to GWSN in which the dissemination of data from a specific sensor does not depend on dissemination of data generated by other sensors. We model the dissemination of a piece of data with a Stochastic-Variable Graph Model (SVGM). SVGM is a weighted graph abstraction in which the edges represent stochastic variables that model propagation delays between neighboring nodes. Latency and reliability performance properties are obtained efficiently through a stochastic shortest path analysis on the SVGM model using Monte Carlo (MC) simulation. The method is accurate and fast, applicable for both partial and complete system analysis. It outperforms traditional discrete-event simulation. As a second contribution, we propose a centrality-based stratification method that combines structural network analysis and MC partial simulation, to further increase efficiency of the system-level analysis while maintaining adequate accuracy. We analyzed the proposed performance evaluation techniques through an extensive set of experiments, using a real deployment and simulations at different levels of abstraction.

Keywords: Wireless Sensor Networks, gossiping, simulation, stochastic modeling, Monte Carlo, epidemic protocols, design-space exploration

1. INTRODUCTION

Wireless Sensor Networks (WSN) are complex systems of spatially distributed sensor nodes that communicate wirelessly, operate autonomously and perform some cooperative action. Due to unstable communication infrastructure, where communication resources are scarce and volatile, gossip protocols represent a good choice for WSN [1,2]. In a gossip-based data dissemination strategy, nodes store a randomized selection of received data samples and re-broadcast those data samples in a randomized way. Typically, gossip-based communication does not require much information about network topology and does not include complex route discovery algorithms. The probabilistic nature of gossiping provides an ad-hoc routing infrastructure and data disseminates via multiple alternative paths. Gossip-based protocols provide robustness in dissemination via increased communication redundancy.

The performance of Gossip-based WSN (GWSN) [1,2] can be assessed through Quality-of-Service (QoS) metrics such as *latency*, the time required to disseminate information, and *reliability*, the success rate of dissemination. These metrics together with the configuration parameters that impact them (node placement, radio, MAC, gossiping, application layer parameters) represent the GWSN design space. For large, heterogeneous GWSN where configuration parameters can be set individually per node, the size of the design space is huge.

Planning and designing GWSN requires an efficient mechanism to explore the design space and to choose optimal configurations. Exhaustive exploration is typically not a feasible alternative. Design-Space Exploration (DSE) could, for example, be performed by a genetic algorithm [3,4]. DSE requires a fast and adequately accurate method to evaluate the network QoS for specific configuration parameters.

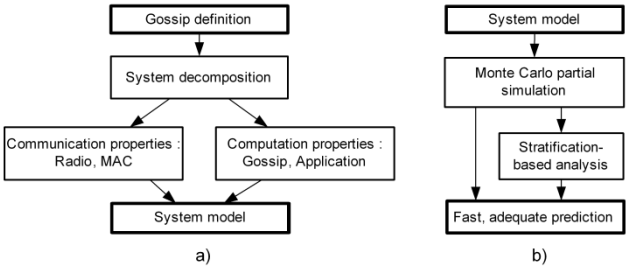

Figure 1. Method overview a) System model b) Performance evaluation.

A lot of modeling efforts in recent years have been focused on the evaluation of individual system aspects: radio [5, 6], data link [7,8], gossiping [9,10], hardware platform [11], and so forth. In this paper, we develop an approach that combines the individual aspect models into an integral system-level model (Figure 1a). Then, using the system model, we develop efficient performance evaluation techniques – Monte Carlo partial simulation and stratification-based analysis (Figure 1b).

The main contributions of this paper are the following. First, we introduce a mathematical model which we call the Stochastic-Variable Graph Model (SVGM). It captures the dissemination properties of a single piece of data in the GWSN with a stochastic graph abstraction. The SVGM is used to drive a Monte Carlo (MC) simulation to analyze dissemination properties (latency and reliability) of specific pieces of data. Second, we propose a stratifica-

Corresponding author: Miloš Blagojević (mblagojevic@tue.nl). Address: Electrical Engineering Department, Eindhoven University of Technology, the Netherlands.

Manuscript received on 20-Dec-2012, revised on 17-Jul-2013 and accepted on 19-Sep-2013.

tion-based analysis that provides a good estimation of the system-level properties by analyzing selected pieces of data and extrapolating the results. Stratification-based analysis uses the results of MC partial simulation; it provides shorter evaluation times, while, as we show, it maintains good accuracy.

In the first part of the paper, we introduce the structure of a gossip-based WSN through a motivating case study. The important GWSN aspects that affect the performance are identified and analyzed. Stochastic and functional abstractions are introduced to model these aspects. Our goal is to provide fast and adequately accurate analysis. Stochastic models abstract from many protocol details, allowing shorter simulation times. Therefore, stochastic models are preferred over detailed functional ones, except when functional models provide a significant gain in accuracy. The nature of gossiping allows us to further improve evaluation speed by using a Stochastic-Variable Graph Model. SVGM is a weighted-graph abstraction of the GWSN in which edges are decorated with stochastic variables representing the time needed to successfully propagate information over individual network links. In the SVGM model, the propagation of a single piece of data is independent of other pieces of data and therefore can be analyzed in isolation.

Typically, sensor nodes in a GWSN periodically sample some physical phenomena and disseminate measurements through the network. Gossiping disseminates the same data simultaneously in multiple directions. In the second part of the paper, we show that the dissemination of a single piece of data can be efficiently analyzed through stochastic shortest path calculation on the SVGM using Monte Carlo simulation. The MC method analyzes the dissemination of each piece of data separately, i.e., it simulates only the part of the system related to the propagation of the specific data item. Such partial simulation provides an efficient solution in situations where one is interested in the performance of only a subset of data items (e.g., high priority data items or worst case analysis). Performance properties of the full system can be obtained by performing one MC partial simulation for each data item. Experiments show that our MC simulation technique outperforms traditional full-system discrete-event simulation. Furthermore, we show that a good estimation of the full-system properties can be obtained by performing MC partial simulations for only a small subset of properly chosen data items and then extrapolating the results. Data items are partitioned into *strata* containing data items that are expected to show similar performance, based on structural properties of the network connectivity. The strata are sampled and a subset of data items is chosen for simulation. The classification of data items into different strata is based on centrality measures that take into account topological properties of their source nodes.

In the final part of the paper, we evaluate the proposed performance estimation techniques in terms of their accuracy and speed. We use extensive simulations and a validation experiment with a real setup. The experiments were designed to provide comparisons at different levels of abstraction, i.e., real deployment, detailed protocol implementation (MiXiM simulation), and high-level discrete event simulation.

The paper is organized as follows. Section 2 discusses related work. Section 3 introduces GWSN through a motivating case study. Section 4 precisely phrases the problem that the paper addresses. In Section 5, we introduce the SVGM abstraction of GWSN, which forms the bases for our MC partial system simulation presented in Section 6. Section 7 introduces the stratification-based analysis. Section 8 presents the experimental evaluation. Section 9 concludes.

2. RELATED WORK

GWSNs characteristically exhibit non-deterministic and probabilistic behavior [12]. The local behavior of a single node is easy to model, but the complex interactions between a large number of nodes with such stochastic behavior are very hard to model. It is not surprising that the dominant technique for performance evaluation for GWSN is simulation. Most alternative methods either cannot cope with real-life system complexity (typically the analytic models) or have poor computational scalability (for instance model-checking approaches).

Analytic models focus on specific aspects of GWSN behavior and provide accurate results only under stringent assumptions that limit the modeling scope. In [9], mean-field analysis is used to evaluate a gossip protocol. It is shown that under certain conditions, when the number of sensor nodes in a finite area tends to infinity, the overall stochastic process can be approximated by a simple, compact deterministic process. An additional underlying assumption in this approach is that nodes are homogeneous, i.e., nodes have identical behavior. This assumption allows that nodes can be modeled with simple identical stochastic models. Such a model provides insight in the gossiping behavior; however, DSE and optimization of WSNs might require analysis of systems with non-homogenous settings of the configurable parameters of the nodes (e.g., transmission power, sampling time). Also, the non-homogenous effects as a result of the placement of the nodes and their interaction with the environment (obstacles, reflection, and refraction) are important for DSE. The performance evaluation method proposed in this paper supports DSE and allows performance evaluation of heterogeneous GWSN.

Another analytic model for gossip-based item dissemination is given in [13]. The model is derived under the assumption of lossless communication, where nodes exhibit homogeneous and uniform behavior over the network. In case the underlying system layers do not provide lossless communication, the nodes do not exhibit uniform behavior and the gossiping model has to be calibrated with experimental results [10]. Model calibration integrally captures the effects of communication aspects and their interaction with the gossiping protocol. In contrast, our modeling approach makes a clear separation between different system aspects, and allows that (if necessary) each of those system aspects can be calibrated independently (typically, the radio model is the one that requires calibration). This allows greater flexibility, and once the radio model is calibrated, performance can be estimated for different parameters of the protocols on top of the radio layer (e.g., MAC and gossiping parameters). The radio model behavior can be obtained directly from prototype deployments or from an analytic radio model

(e.g., in the form of a WSN design tool as described in [14]).

Another approach to gossiping performance evaluation is to use model-checking tools. A probabilistic model checking tool, Prism, has been used in several studies for formal verification and performance analysis of gossiping [15,16]. The model-checking methods suffer from the rapid increase in the number of global system states, commonly known as the state-space explosion problem. The computation overhead and lack of scalability is the biggest disadvantage of model-checking techniques. As a result, their application is limited to the analysis of small networks. To achieve better scalability, some methods focus only on the best- and worst-case performance [15].

Alternative methods to improve scalability combine probabilistic model checking with simulation techniques. In [17,16], Prism is used to perform formal analysis of gossiping in small network topologies with a goal to provide insight in the gossiping behavior (e.g., event probabilities and performance bounds). Then the obtained results are extended to larger networks using Monte Carlo techniques. The idea of incorporating small-scale properties in a large-scale simulation is similar to our approach. None of the model-checking approaches is sufficiently fast for DSE though; instead, in this paper, we use probabilistic models of the system layers. These models are integrated in the SVGM abstraction used to drive fast Monte Carlo simulations.

Discrete-event simulators are the most popular method for GWSN analysis. Often, WSN simulations are done by modifying some of the general network simulators (ns-2 [18], Omnet++ [19], Opnet [20]) to the WSN specifics. A WSN simulation framework for the Omnet++ simulator has been developed – MiXiM [21], which includes many libraries and detailed models for lower WSN protocol layers. The speed of MiXiM is acceptable for analyzing system behavior and gaining insight in the interaction between system parts, but it is also not sufficiently fast for DSE. We compare our methods with MiXiM simulations in Section 8.

Some of the ideas developed in this paper were presented in an early form in [22], which informally describes performance simulation in terms of a case study. The current paper generalizes these ideas to a hybrid modeling framework that uses a combination of probabilistic and functional abstractions for fast performance evaluation. In order to analyze dissemination properties of specific data items, we propose a MC partial simulation technique. Furthermore, we introduce novel ideas to efficiently estimate complete system performance using a combination of partial simulations and stratification-based network analysis methods. Finally, our experimental evaluation is more extensive, including a validation experiment with a real setup and experiments with irregular and heterogeneous network topologies.

3. GOSSIPING WIRELESS SENSOR NETWORKS

In this section, we introduce gossip-based WSN through a motivating case study from the health-care domain that serves as a running example throughout the paper and as a platform for validating our methods in a real setup. We also start introducing the notation we use in the paper. For reference, a table with an overview of used notation is provided at the end of the document.

3.1 Motivating case study

The selected health-care application is developed by Roessingh Research and Development (RRD, www.rrd.nl) with the goal to assist patients suffering from COPD (Chronic Obstructive Pulmonary Disease) in daily living activities [37]. Sensor nodes deployed at known positions periodically measure various physical phenomena, such as pressure, movement or current. The network topology is rather static (mobility is limited to occasional movement of nodes attached to chairs, doors, etc.) and data needs to be collected at regular intervals. Through gossiping, measured data is delivered to each node in the network. The application may extract data from the GWSN at any of the network nodes.

The basic communication model of such a GWSN consists of four layers (physical, data-link, gossiping and application layer) as shown in Figure 2. In the rest of this section, the communication layers and architecture of a typical GWSN are analyzed in more detail using the case study.

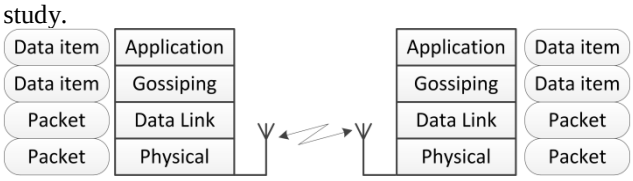

Figure 2. Layered architecture of a GWSN.

3.2 Application layer

Sensor nodes perform measurements with a regular time period, *sampling period* T_{smp}. The measurements are disseminated and stored in the form of *data items*. A data item is a piece of information that consists of some value m (the measurement of the phenomenon) and a version v (e.g., a sequence number or a time-stamp indicating when the measurement was taken). For simplicity, we assume that item versions are consecutive natural numbers starting from 1 (we use $\mathbb{N}$ to denote the natural numbers, not including 0). Besides that, each data item has a *key* number $k \in K$, where K denotes the set of keys, that uniquely identifies the source of the information contained in that data item, e.g., identifying the sensor where the measurement was taken. For simplicity of presentation, we assume that each node has one sensor attached to it and that nodes have unique ids that also function as data item keys.

Each node has its own information store for data items, called the *cache*. The size of the cache in the WSN is limited and a node can keep just one version of each data item. Note that the application layer may extract data items from the gossiping layer and store them as required. The cache content at node i is denoted with a cache function: $C_i: K \times T \hookrightarrow \mathcal{V}$, where $T \subseteq \mathbb{N}$ is the time domain expressed in terms of discrete communication rounds and $\mathcal{V} \subseteq \mathbb{N}$ is the set of version numbers. The notation $C_i(k,t) = v$ denotes that at time t in the cache of node i data item k is present with version v. In case data item k is not present, the function is undefined, which we denote as $C_i(k,t) = \perp$.

Note that the actual payload of a data item does not have any impact on the gossiping behavior and therefore it

does not play a role in the performance metrics used in this paper (see Section 4).

3.3 Gossiping layer

Gossip protocols are relatively simple communication protocols that disseminate information in a manner similar to gossiping in social networks. Randomly selected data items are repetitively broadcast to nodes within communication range. Gossiping does not include any error-control retransmission mechanism and provides no guarantees about information delivery. Nevertheless, the probabilistic selection of items to be transmitted provides that items can be retransmitted several times, resulting in an improved reliability and robustness. Gossiping is implemented through two policies, *item selection* and *cache update*.

Item selection is periodically initiated by the nodes. In each periodic communication round, a node transmits a *packet*. The *item selection* procedure selects data items to be transmitted in that packet. The number of data items that can fit to a single packet is denoted as N_S. Each time t that the item selection procedure is executed, a node i selects N_S data items from the information pool in its cache $Pool(i,t) = \{k \in K | C_i(k,t) \neq \perp\}$ and creates the packet payload (Figure 3). In the selection procedure implemented in our case study, for instance, one of the selected data items is always the local measurement, while the other $(N_S - 1)$ items are randomly selected from items present in the cache.

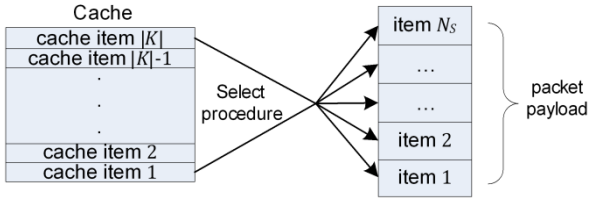

Figure 3. Item selection procedure

The *cache update* procedure is executed each time a node receives a packet. The typical update procedure is simple: as soon as a node receives a newer version of a data item the old one is overwritten.

3.4 Data-link layer

Data items selected for transmission are encapsulated in MAC packets. The MAC protocol defines a transmission time schedule for each packet. MAC efficiency depends on its ability to create a communication schedule that (1) avoids collisions, i.e., simultaneous transmissions by nodes in communication range and (2) provides schedule matching, i.e., at the moment when a packet is transmitted, the receiver's radio is in the listening mode.

In our case study, we use gMAC [23], a TDMA-based MAC protocol, designed for gossiping. The communication schedule is organized in TDMA frames consisting of assigned communication slots to provide collision-free communication. To reduce energy consumption, the communication takes place only in a small part of the frame, the *active period*, and each node transmits only one packet per TDMA period. Scheduling is similar to the LMAC protocol [24] and each node gets a transmission slot unique within the 2-hop neighborhood. gMAC divides the active period in N_L listening segments. In each round, a node randomly chooses to listen to only one of those segments. As a result, energy consumption is reduced, but only a subset of neighbor nodes can receive a transmitted packet. There is no error control mechanism at the MAC layer, and a packet is transmitted only once.

3.5 Physical layer

The GWSN case study is implemented on the MyriaNed WSN platform [25]. A MyriaNode is the basic building block of MyriaNed. It uses a 2.4GHz packet radio (Nordic nRF24L01) with 32 byte packets, and a data rate of 2Mbps.

The wireless medium is inherently unreliable and occasionally packets get lost. Packet reception success can be seen as a stochastic process depending on the various parameters of the physical layer, such as transmission frequency, packet size, antenna radiation properties, distance between nodes, interference in the radio channel, and obstacles in the environment.

4. PROBLEM STATEMENT

In this section we define the problem in more detail. We define the system-level quality metrics that provide an insight in the system behavior.

4.1 Data item dissemination

We first define notation that allows us to capture the dissemination process of data items in the network. The gossip process for a given data item (k,v) with key k, and version v starts at the moment $T_S(k,v)$, when information source k generates data item (k,v).

Definition 1. The *arrival time* $T_E(j,k,v)$ of data item (k,v) at node j, is the number of the communication round in which data item arrives for the first time at node j.

$$T_E(j,k,v) = \inf\{t | C_j(k,t) = v\} \quad (1)$$

We use $T_E(j,k,v) = \infty$ to denote the situation where data item (k,v) never arrives at node j. In that case we say that it is *lost*.

Definition 2. Delivery time $D_j(k,v)$, denotes the time necessary to deliver version v of data item k from its source node to node j. It is defined as the time between the moment when the version of data item k is generated, $T_S(k,v)$, until the moment it appears in the cache of node j, $T_E(j,k,v)$:

$$D_j(k,v) = \begin{cases} T_E(j,k,v) - T_S(k,v) & T_E(j,k,v) \neq \infty \\ \perp & T_E(j,k,v) = \infty \end{cases} \quad (2)$$

Delay $D_j(k,v)$ is undefined ($\perp$) if an item is not delivered.

Definition 3. Delivery success, $U_j(k,v)$, of item (k,v) to node j is 1 if the data item arrives at node j and 0 otherwise:

$$U_j(k,v) = \begin{cases} 0 & \text{if } T_E(j,k,v) = \infty \\ 1 & \text{otherwise} \end{cases} \quad (3)$$

4.2 Gossiping properties

The quality of gossip-based information disseminationcan is characterized with two essential properties: the number of nodes that ultimately receive the data items and

the time needed for the data items to spread. In this work, these two properties are evaluated through two metrics, referred to as *reliability* and *latency*, respectively. These properties are defined precisely in the following subsections respectively.

4.2.1 Latency

Latency is a metric that quantifies the time (the number of communication rounds) necessary to distribute a data item. We consider three different levels of latency metrics: *point-to-point*, *point-to-multipoint* and *network latency*.

Point-to-point latency $L_j(k)$ denotes the average time necessary to deliver a version of data item k from its source node k to some node j, averaged over all versions that reach node j.

Definition 4. The *point-to-point latency* $L_j(k)$ is defined as the sample mean of the delivery times of all versions that reach j. It is formally defined as follows.

$$L_j(k) = \lim_{N_{smp}\to\infty} \frac{\sum_{1\le v\le N_{smp}, U_j(k,v)=1} D_j(k,v)}{\sum_{1\le v\le N_{smp}} U_j(k,v)} \quad (4)$$

Definition 5. The *point-to-multipoint latency* $L^D(k)$ characterizes the time necessary to deliver data item k to the set D of destinations of interest. It is defined as the average value of point-to-point latencies for data items k over the set D:

$$L^D(k) = \frac{1}{|D|}\sum_{j\in D} L_j(k) \quad (5)$$

(It assumes that the nodes in D are reached by items k, so they have a well-defined point-to-point latency.)

Definition 6. Thirdly, the *network latency* $L^{K,D}$ is the time to deliver an item averaged over all data item keys of interest K. It is defined as the average point-to-multipoint latency over the set K of data item keys, for a given set D of destinations of interest:

$$L^{K,D} = \frac{1}{|K|}\sum_{k\in K} L^D(k) \quad (6)$$

4.2.2 Reliability

Reliability is defined as the fraction of successfully delivered data items. If a node does not receive some version of a data item, we consider that version lost for that specific node. Similar to latency, we define three metrics: *point-to-point reliability*, *point-to-multipoint reliability* and *network reliability*.

Definition 7. Point-to-point reliability $R_j(k)$ denotes the fraction of versions of data item k delivered successfully to node j. It is defined as the ratio of the number of versions of data item k that appear in the cache of node j, over the total number of generated versions. Point-to-point reliability is formally defined as follows.

$$R_j(k) = \lim_{N_{smp}\to\infty} \frac{1}{N_{smp}} \sum_{v=1}^{N_{smp}} U_j(k,v) \quad (7)$$

Point-to-multipoint reliability $R^D(k)$ and network reliability $R^{K,D}$ are defined in the same way as their latency counterparts.

4.3 Other properties

Lifetime denotes the time necessary for nodes in the network to deplete their energy source and thus depends on the energy consumption. It is an important performance metric for WSN design. The power consumption depends on the transmission power settings of individual nodes, packet length, number of slots in the active period, etc. The type of gossiping networks analyzed in this paper exhibits periodic behavior, where parameters that affect power consumption do not change over time. In such a case, estimation of node power consumption is straightforward and therefore lifetime is not a metric of interest in this paper.

Redundancy in the communication is another relevant gossiping metric. In general redundancy can be: 1) transmission redundancy, showing how many times a node has transmitted a data item (that is already received by all neighbors); 2) receiving redundancy, showing how many times a node has received the same data item. For latency and reliability estimation all that matters is the moment when a node has received a data item for the first time. For each node, we have to simulate all data item transmissions until all neighbors have received that data item. Further (redundant) transmissions do not affect the reliability and latency metrics, and do not need to be simulated. In order to estimate redundancy, for each node, simulation of data item transmissions should continue until the data item is replaced in the cache (i.e., until it is overwritten with a newer version). Such a modification is straightforward and we do not address the redundancy metric further in this paper.

4.4 GWSN-specific challenges

There are several factors specific to GWSN that make performance evaluation challenging:

1) Communication between nodes in WSN is intrinsically unreliable, or stochastic, due to the nature of the radio channel and data-link properties. The time necessary for a node to spread information to its neighbors is therefore not fixed, but a stochastic variable that depends on the system attributes.

2) Data items are simultaneously propagated in multiple directions, and multiple copies of the same data item are present at different locations in the network. The latency to reach some node is determined by the distribution path (chain of nodes) that the item followed until the first time the node receives it. In Figure 4, two possible distribution paths for information from the node in the bottom left corner to all other nodes are given. A connection between two nodes indicates that the item reached a node for the first time by a transmission from the other node. Both the distribution paths may be the result of the same stochastic process; due to stochastic variations in link delays the concrete paths turned out different.

3) Due to the random nature of GWSN, a newer version of the same data item may be distributed faster than an older one. This leads to suppressed item versions, where an old version is discarded and its further dissemination is interrupted. The interaction between different data item versions has an important effect on system performance, especially on reliability.

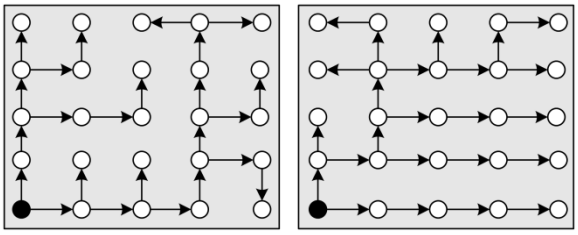

Figure 4. Distribution paths for two different data item versions that start from the lower left node.

The above challenges show that a performance evaluation method for GWSN has to consider the stochastic nature of WSN communication, the random distribution paths, and the interactions between different data item versions.

5. STOCHASTIC-VARIABLE GRAPH MODEL

In this section we introduce our mathematical abstraction of the dissemination process of a data item. We show how the relevant properties of GWSN can be captured in a stochastic-variable graph model (SVGM). A stochastic-variable graph $G_{SV}(V, E, X)$, is a connected, directed graph, with vertices V and edges E. The vertices represent WSN nodes and the edges represent direct links for data-item dissemination. The edge weights $X: E \rightarrow (\mathbb{N} \rightarrow [0,1])$ in the graph represent the distributions of the duration of information dissemination along the given edge (recall that time is measured in terms of discrete communication rounds). The edge weights are stochastic variables $X(i,j)$ with $(i,j) \in E$, where stochastic variable $X(i,j)$ models the propagation delay from node i to node j. An SVGM is used to capture the dissemination process of a specific version v of a data item k. In general, for each data item (k,v) an SVGM is instantiated, i.e., for each version v and key k stochastic variables $X_{k,v}(i,j)$, $i,j \in V$ are instantiated.

Stochastic variables describing propagation time have to capture the important system aspects (radio, data-link, gossiping and application) of a GWSN and the interaction between them. The gossiping properties are analyzed through a stochastic ensemble of weighted graphs sampled from the SVGM (Section 6). In this section, first, we introduce the stochastic models of relevant system aspects required for an SVGM. Second, we discuss the composition of the stochastic variables and describe the process of sampling weighted graphs from the SVGM. Finally, we illustrate how SVGM sampling provides a basis for performance simulation.

5.1 Stochastic models of GWSN system aspects

The stochastic variable $X(i,j)$ describing the time necessary to deliver a data item over link (i,j) integrates several simple stochastic models of different system aspects: communication (radio and MAC) and computation (gossiping and application).

5.1.1 Communication aspects of GWSN

The *communication* aspects of a WSN can be modelled by a connected directed weighted graph $G = (V, E, p)$, where the set V of vertices denotes the nodes in the network, the set of directed edges E captures the potential communication links, and $p: E \rightarrow [0,1]$ is an edge weight that captures the probability of successful communication over the links.

Definition 8. Success probability $p(i,j)$ specifies the probability that a packet sent by node i is successfully received by node j.

This probability models simultaneously the physical radio and the data link (MAC) behavior. Subsequent transmissions on the radio level are assumed to be independent and identically distributed.

For the radio part, the packet reception ratio (PRR) is a commonly used metric to quantify radio communication.

Definition 9. Packet reception ratio $p_r(i,j)$ denotes the probability that a packet transmitted at node i is successfully received at node j.

According to [26] this is a fairly good model to approximate reality. The experiment with a real setup that we did, and on which we report in Section 8, confirms that conclusion. The point-to-point PRRs can be obtained through direct calibration measurements or using an analytic model (e.g., in the form of a WSN design tool as described in [14], or models such as those of [5,27]).

Definition 10. The *MAC success rate* $p_{mac}(i,j)$ models MAC properties. It is defined as the probability that a packet transmitted by the data-link layer of a node i is received successfully at the data-link layer of a node j, assuming the physical radio communication is successful.

Proposition 1. In the gMAC protocol the MAC success rate is $p_{mac} = 1/N_L$

Proof. The MAC protocol is successful in the case that: 1) when node i transmits a packet, node j has its receiver on (*matching*) and 2) none of the other nodes within the interference range of node j transmit a packet at the same time (*collision*). In the case study, the gMAC protocol provides collision-free communication. Thus, here MAC success rate depends only on the schedule matching between nodes i and j. In the gMAC protocol, a node randomly chooses one of N_L non-overlapping listening segments, so that the MAC success rate is given by: $p_{mac} = 1/N_L$.□

The radio and MAC probabilities are per-transmission independent, therefore the weights $p(i,j)$ in our WSN model become: $p(i,j) = p_{mac}(i,j) \cdot p_r(i,j)$. In our case study, this results in $p(i,j) = p_r(i,j)/N_L$.

5.1.2 Computation aspects of GWSN

The computation aspects of WSN are related to the cache-specific processes: item selection and cache update. The *item selection* procedure is the heart of the gossiping mechanism. It selects N_S data items from the cache. In our case study, the local measurement and $N_S - 1$ randomly selected other data items are chosen. Except during an initial start-up period, caches will always be full and contain one data item from each of the sources in K (the freshest received version).

Proposition 2. After the initial period is over, the probability that a data item k is selected from the cache of node i, in our case study, is:

$$p_s(i,k) = \begin{cases} \frac{N_S - 1}{|K| - 1} & \text{if } k \neq i \\ 1 & \text{if } k = i \end{cases} \quad (8)$$

Proof. Straightforward. If $k = i$ the item is guaranteed to be selected in this protocol. If $k \neq i$ then the remaining $N_S - 1$ places for items to be communicated are uniformly randomly selected from the remaining $|K| - 1$ items in the filled cached.□

Note that the model captures the behavior in steady-state and there may be some discrepancies during the transient period at the startup, until caches are fully filled.

The application requires only the freshest data, so each time a new version is received the old one is removed from the cache. The cache update process is deterministic, but dependent on the data item versions present in the cache. A stochastic abstraction describing cache content from the perspective of an individual data item cannot be obtained easily [10]. Instead, we propose an approach that first models item dissemination without considering suppression of data items by newer versions and incorporate this effect later on when we integrate the dissemination of individual items (Section 6.3).

5.2 Monte Carlo sampling of SVGM

We evaluate the characteristic properties of the stochastic graphs using Monte Carlo sampling. Monte Carlo sampling of the SVGM provides a concrete, non-stochastic, weighted graph $G_S\big(V, E, D(k,v)\big)$, where edge weights $D(k,v): E \to \mathbb{N}$ correspond to the sampled, concrete propagation delays of data item k version v (see Figure 5). This graph represents the concrete dissemination of a single version (5 in the figure) of a single data item (2).

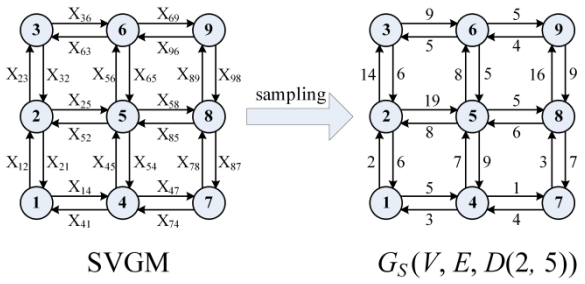

Figure 5. Stochastic Variable Graph Model (left, with X_{ij} denoting $X_{k,v}(i,j)$) and weighted graph $G_S\big(V, E, D(2,5)\big)$ (right, edge weights are sampled propagation delays).

The time (number of rounds) required for node i to deliver a data item to node j depends on two factors:

1) The number of attempts (transmissions) needed to deliver a data item successfully. This number depends on the probability of successful item transmission, and is independent among different neighbor nodes of the source node.

2) The time between attempts to transmit the same data item. This time depends on the item selection procedure which determines what is going to be transmitted in each frame. The time between transmissions is the same for all neighbors of the source node due to the use of broadcast communication.

Thus, the propagation delay is modeled as the stochastic process, $X_{k,v}(i,j)$, which consists of two constituent stochastic processes: item transmission and item selection. Based on the previous analysis of the propagation delay, we have that:

1) The process of item selection at node i is common for all stochastic variables $X_{k,v}(i,j)$, for j such that $(i,j) \in E$. The time between transmission attempts for specific data items is the same for all outgoing edges of vertex i. As a result, stochastic variables $X_{k,v}(i,j)$ and $X_{k,v}(i,n)$ are correlated. For the example given in Figure 6, the times between transmissions of some data item from node i are the same for neighbor nodes j, l, and n.

2) The processes of item selection on different nodes are assumed to be independent, as are the processes of the item transmissions between different pairs of nodes. Based on these assumptions, it is not hard to show that stochastic variables $X_{k,v}(i,j)$ and $X_{k,v}(m,n)$ are independent if $i \neq m$.

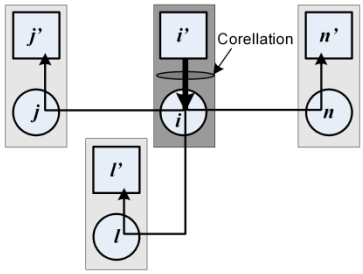

Figure 6. Spatial correlation.

We denote the number of transmissions of data item k, version v in node i until a packet is delivered successfully to node j as $N_{Tx}(i,j,k,v)$. Furthermore, we denote the time between the $(m\text{-}1)$-th and the m-th transmission of data item k from node i as $T_{Tx}(m,i,k,v)$ (where for $m = 1$, T_{Tx} returns the time from arrival until first transmission). Recall the successful transmission probability p and the item selection probability p_s introduced in Section 5.1. The success per round has a Bernoulli distribution and the number of Bernoulli trials needed to get one success is described by a geometric distribution. Then the stochastic processes of item transmission (the number of trials before an item is successfully transmitted) and the time between transmissions (the number of trials before the item is selected) are both described with geometric distributions:

$$\Pr(N_{Tx}(i,j,k,v) = l) = \big(1 - p(i,j)\big)^{l-1} p(i,j) \quad (9)$$

$$\Pr(T_{Tx}(m,i,k,v) = l) = \big(1 - p_s(i,k)\big)^{l-1} p_s(i,k) \quad (10)$$

Note that the distributions are independent of the version v.

Based on the previous analysis, a general strategy for random sampling of the stochastic variable for link delay is defined (Monte Carlo sampling). First, for each of the neighbors j of node i, the number of required transmissions to successfully deliver data item from node i to node j, $N_{Tx}(i,j,k,v)$, is sampled from the distribution of Eqn. (9). The total number of transmissions needed to deliver item k, version v, over *all* outgoing links from node i, is determined as the maximum number required for any neighbor j:

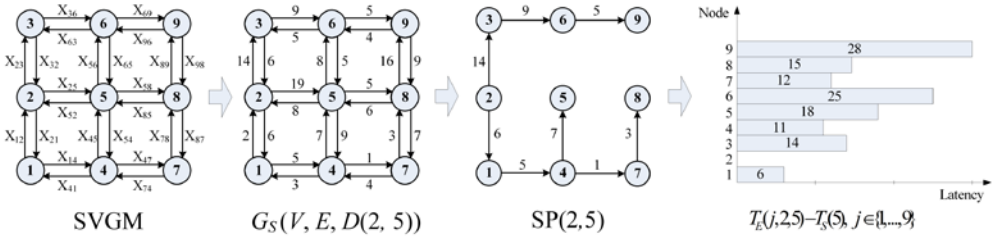

Figure 7. Calculating latency of a single version (5) of a single data item (2) to all other nodes (with X_{ij} denoting $X_{k,v}(i,j)$).

$$N_{max}(i,k,v) = \max_{j\ s.t.(i,j)\in E} N_{Tx}(i,j,k,v) \quad (11)$$

To model the distribution process of data item (k,v) to *all* neighbors of i, we therefore need concrete samples for $N_{max}(i,k,v)$ lengths of inter-transmission intervals The intervals between transmissions $T_{Tx}(m,i,k,v)$, $1 \le m \le N_{max}(i,k,v)$ are sampled from the distribution of Eqn. (10).

Finally, the sampled delay between nodes i and j in the SVGM for item k and version v is calculated as the sum of the first $N_{Tx}(i,j,k,v)$ inter-transmission intervals, namely the number of attempts to transmit the item until success for neighbor j:

$$D_{ij}(k,v) = \sum_{m=1}^{N_{Tx}(i,j,k,v)} T_{Tx}(m,i,k,v) \quad (12)$$

5.3 Shortest-path abstraction of dissemination

Once we have the sampled graph representing the node to node dissemination of a data item, we can use it to determine point-to-point latencies. The analysis is based on the following characteristic of the sampled graph.

Proposition 3. The latencies of a data item (k,v) to each of the nodes can be calculated as the shortest paths from node k in the sampled graph.

Proof. Gossiping disseminates data items over multiple links simultaneously, resulting in stochastic distribution paths. The edge weights of the sampled graph are the concrete propagation delays of the links for a specific data item. The dissemination along a single link starts as soon as the data item arrives at the source of the link and takes an amount of time according to the edge weight. Hence, the time to propagate along a certain path is obtained from the sum of the edge weights. The shortest path between nodes k and j then corresponds to the first arrival of data item generated at node k to node j.□

The evaluation of latencies using the sampled graph is illustrated in the example in Figure 7.

In order to get statistically significant metric estimation, the propagation of sufficiently many data item versions has to be simulated. For each data item version, a new weighted graph is sampled from the SVGM. The point-to-point latency for delivery of data items from node k to node j as defined in Eqn. (4) can be estimated as:

$$\hat{L}_j(k) = \frac{1}{N_{smp}} \sum_{v=1}^{N_{smp}} T_E(j,k,v) - T_S(k,v) \quad (13)$$

where N_{smp} denotes the total number of simulated versions.

6. MONTE CARLO PARTIAL SIMULATION

6.1 Overview of the method

The shortest-path analysis described in the previous section captures the propagation of an individual data item (independently from propagation of other data items in the system). Such analysis does not take into account potential loss of data items due to suppression by newer versions. The interaction between different versions of a data item can be analyzed through a combination of a discrete-event simulation and the Monte Carlo shortest path analysis. The shortest-path analysis remains the core of such an approach, but the discrete-event mechanism allows us to combine results corresponding to different versions of a data item, i.e., different instances of sampled graphs. An event is defined as the first arrival of some data item to some node. A graph sampled from an SVGM describes event times assuming no interaction between data items. The algorithm to calculate event times for a single data item in isolation is introduced in Section 6.2. An efficient method for the simultaneous analysis of multiple item versions is presented in Section 6.3.

The duration of a MC simulation is typically smaller than the simulation time needed for a plain discrete-event (DE) simulation due to two main reasons:

1) MC simulation is partial. Propagation patterns for only one data item are simulated at a time, while the propagation of all data items is simultaneously simulated in DE simulation.

2) The number of events that are simulated is considerably smaller in comparison with standard DE simulation. In plain DE simulation, an event corresponds to individual packet transmissions, while in the MC simulation an event denotes first-time successful reception of an item. Thus, fewer events need to be simulated.

6.2 MC simulation for a single data item version

The problem of calculating the latency of a data item in a sampled graph can be mathematically described with a set of fixed-point equations, and solved using Dijkstra's shortest-path algorithm.

Assume that system time t_{mc} denotes the time passed from the beginning of simulation (measured in communication rounds). Event time $T_E(j,k,v)$ is defined as the time when node j receives a data item (k,v) for the first time. This event time corresponds to the arrival time as defined in Eqn. (1).

We maintain two lists according to the node status with respect to the specific data item: the list of the nodes that have received data item (k,v), called the *infected* nodes:

$$IN(t_{mc}, k, v) = \{i \in V | T_E(i, k, v) < t_{mc}\} \quad (14)$$

and the list of nodes that have not yet received the data item, the *susceptible* nodes:

$$SU(t_{mc}, k, v) = V \backslash IN(t_{mc}, k, v) \quad (15)$$

The naming of the lists comes from the fact that gossiping behavior mimics the spread of a virus. An event results in a change in the infected list, i.e., a node that has received a data item version for the first time is added to the list of the infected nodes and removed from the list of the susceptible nodes.

Furthermore, we denote the inter-event time $T_{IE}(i, j, k, v)$ as the time necessary for node i that has received data item (k, v) to deliver it to node j. Sampling the SVGM provides a weighted graph $G_S\big(V, E, D(k, v)\big)$, where the inter-event times are the edge labels in that graph:

$$T_{IE}(i, j, k, v) = D_{ij}(k, v) \quad (16)$$

The inter-event time $T_{IE}(i, j, k, v)$ is the result of a time-invariant process and thus it does not depend on the moment when the item arrives at node i.

For the source node, the event time of the arrival of item (k, v) is equal to its sampling time $T_S(k, v)$. In case destination node j is not the source node ($j \neq k$), a data item can be delivered to node j only via one of its neighbors. If $T_E(i, k, v)$ denotes the time when a node i, neighbor of node j, received the data item (k, v) for the first time, then the time when node i delivers the same data item for the first time to node j is equal to: $T_E(i, k, v) + T_{IE}(i, j, k, v)$. This sum is called an *event-time estimate* for node j via node i. The time of the first arrival of data item (k, v) to node j is equal to the minimum of the event-time estimates via each of the neighbor nodes. The set of event times can be described with the following fixed-point equations:

$$\begin{aligned} T_E(j, k, v) &= T_S(k, v) && j = k \\ T_E(j, k, v) &= \min_{i \text{ s.t. } (i,j) \in E} T_E(i, k, v) + T_{IE}(i, j, k, v) && j \neq k \end{aligned} \quad (17)$$

It is straightforward to show that the equations permit a unique solution [28]. The set of fixed-point equations is solved dynamically. As shown in [28], the solution for this set of equations is the same as the result of Dijkstra's shortest path algorithm applied on graph $G_S\big(V, E, D(k, v)\big)$ with an additional shift according to the starting sampling time value (Figure 7). Thus, Dijkstra's algorithm [29,30] can be used as a method for event list calculation.

$T_E = Event\_list(G_S, k, v)$
1. Initialize: $T_E(i, k, v) = \infty$, $T_E(k, k, v) = T_S(k, v)$
2. $IN = \emptyset$
3. $SU = V[G_S]$
4. **while** $SU \neq \emptyset$ **do**
5. $p \leftarrow \text{argmin}_{i \in SU} T_E(i, k, v)$
6. $IN \leftarrow IN \cup \{p\}$; $SU \leftarrow SU \backslash \{p\}$
7. **for each** $q \in V$ **such that** $(p, q) \in E$
8. $T_E(q, k, v) = \min\big(T_E(q, k, v), T_E(p, k, v) + T_{IE}(p, q, k, v)\big)$
9. **end for**
10. **end while**

Figure 8. MC pseudo-code.

The pseudo-code for MC event list calculation based on Dijkstra's algorithm is given in Figure 8. Initially, the event time of the source node is set to its sampling time, while all other event times are set to infinity. Each time when an event is executed, the MC algorithm selects the vertex $p \in SU$ with the minimum tentative event time, adds p to the infected list and removes it from the susceptible list, making its event time definitive, and finally updates tentative event times of remaining nodes (lines 7-8.). The event times are tentative, since its computation considers only the currently infected nodes. Each time when a new node is infected, computed event times might change and have to be updated (line 8.). The minimum tentative event time is the time of the next event since this cannot change anymore.

Computationally, the most demanding task in this procedure is to obtain the minimum of the tentative event times (from the set of susceptible nodes, line 5.). Each time new tentative event times are calculated (lines 7-8.), the values have to be sorted. In order to reduce the procedure's complexity, in our implementation, the set of tentative event times is organized as a min-priority binary heap [30]. In that case, the complexity of MC event list calculation is $\mathcal{O}(|E| \, log|V|)$, where $|E|$ denotes the total number of edges and $|V|$ the number of vertices in the graph G_S.

6.3 MC simulation for multiple data item versions from the same source

The MC simulation method described in the previous section does not take into account the interaction between different versions of a data item. The more often items are sampled, the more refined is the sampling of the physical phenomena being monitored, but the higher is the chance that some items are overtaken by newer versions and therefore are considered lost. In this section, we show how the set of fixed-point equations describing event times can be extended with a set of deadline constraints so that relevant interactions which lead to lost items are captured.

The propagation of N_{smp} versions of data item k can be described with the stochastic ensemble of weighted graphs sampled from SVGM for all N_{smp} versions:

$$\boldsymbol{G}_k = \big\{G_S\big(V, E, D(k, v)\big) \big| 1 \leq v \leq N_{smp}\big\} \quad (18)$$

According to the GWSN description in Section 3.2, dissemination of a data item (k, v) is independent of the dissemination of previous versions of the same data item; older versions never replace newer versions. As a result, the propagation properties of a data item (k, v), computed from the sampled graph $G_S\big(V, E, D(k, v)\big)$, do not have any influence on the event times of the data item $(k, v + 1)$, but might influence the event times of the data item $(k, v - 1)$. The idea is to analyze the stochastic ensemble of Eqn. (18) in reverse temporal order. In this case, whenever it is necessary, MC simulation can 'look into the future' and check whether a specific event is suppressed by some other event corresponding to the propagation of some later version of the same data item.

As before, the time of the first arrival of data item (k, v) to node j is equal to the minimum over corresponding event-time estimates of the set of neighbor nodes. This is described with the same set of fixed-point equations

(Eqn. (17)). However, the interactions between versions impose additional constraints that event times have to satisfy. These constraints can be considered as 'deadlines' determined by the propagation of next versions of the data item. Data item (k, v) is successfully delivered to node j only if two deadline conditions are met:

1) Version v can be successfully *received* at time $T_E(j,k,v)$, only if none of the later data item versions has already arrived to node j before event time $T_E(j,k,v)$:

$$\text{for all } n \in \mathbb{N}, T_E(j,k,v) < T_E(j,k,v+n) \quad (19)$$

2) Version v can be successfully *transmitted* from infected neighbor node i at time $T_E(j,k,v)$, only if prior to that no newer version of the item has already arrived to node i (because a new version causes the old one to be removed):

$$\text{for all } n \in \mathbb{N}, (i,j) \in E, T_E(j,k,v) < T_E(j,k,v+n) \quad (20)$$

Reversed version order analysis provides that the previous conditions can be easily evaluated. For each node j, the latest delivery time (deadline) for data item (k, v), $T_L(j,k,v)$ is defined as the earliest delivery time of any later data item version up to N_{smp}, the number of versions simulated:

$$T_L(j,k,v) = \min\{T_E(j,k,n) \,|\, v < n < N_{smp}\} \quad (21)$$

The list of latest delivery times can be updated easily with $\mathcal{O}(1)$ complexity. Each time a graph instance $G_S(V, E, D(k,v))$ is analyzed, the latest delivery times are updated with new information. Based on Eqn. (21), it easily follows that:

$$T_L(j,k,v-1) = \min(T_E(j,k,v), T_L(j,k,v)) \quad (22)$$

Considering constraints (19) and (20), the set of fixed-point equations describing event times can be written as:

$$\begin{aligned} T_E(j,k,v) &= T_S(k,v) && j = k \\ T_E(j,k,v) &= \min_{i\ s.t.(i,j)\in E} T_E(i,k,v) + T^*_{IE}(i,j,k,v) && j \neq k \end{aligned} \quad (23)$$

where the inter-event times are modified to include deadlines imposed by propagation of the later data item versions:

$$T^*_{IE}(i,j,k,v) = \begin{cases} T_{IE}(i,j,k,v) & C_1 \text{ and } C_2 \\ \infty & \text{otherwise} \end{cases} \quad (24)$$

where conditions C_1 and C_2 are defined as follows.

C_1: $T_E(i,k,v) + T_{IE}(i,j,k,v) < T_L(j,k,v)$

C_2: $T_E(i,k,v) + T_{IE}(i,j,k,v) < T_L(i,k,v)$

In case one or both of the deadline constraints C_1 and C_2 cannot be met, the inter-event time $T^*_{IE}(i,j,k,v)$ is defined to be infinity, i.e., the data item cannot be delivered to node j via node i. As a consequence, in the case that there is no neighbor of node j for which both conditions are met, data item (k, v) is never delivered to node j and the corresponding event time $T_E(j,k,v)$ is infinite.

The solution for the set of fixed-point equations of Eqn. (23) is calculated by incorporating deadline constraints into the MC algorithm of Figure 8. The modified pseudo-code which performs dynamic event-list calculation taking into account interactions between items is given in Figure 9.

$[T_E, T_L] = Event\_list(G_S, k, v, T_L)$
1. Initialize: $T_E(i,k,v) = \infty$, $T_E(k,k,v) = T_S(k,v)$
2. $IN = \emptyset$
3. $SU = V[G_S]$
4. **while** $SU \neq \emptyset$ **do**
5. $p \leftarrow \text{argmin}_{i \in SU}\, T_E(i,k,v)$
6. $IN \leftarrow IN \cup \{p\}$; $SU \leftarrow SU \backslash \{p\}$
7. **for each** $q \in V$ **such that** $(p,q) \in E$ **do**
8. **if** C_1 **and** C_2 **then**
9. $T^*_{IE}(p,q,k,v) = T_{IE}(p,q,k,v)$
10. **else**
11. $T^*_{IE}(p,q,k,v) = \infty$
12. **end if**
13. $T_E(q,k,v) = \min(T_E(q,k,v), T_E(p,k,v) + T^*_{IE}(p,q,k,v))$
14. $T_L(q,k,v-1) = \min(T_E(q,k,v), T_L(q,k,v))$
15. **end for**
16. **end while**

Figure 9. Modified MC pseudo-code.

The simulation process consists of N_{smp} executions of the modified MC algorithm (one for each graph in the stochastic ensemble). Propagation of data item versions is evaluated in reverse time order. Initially, the latest delivery time (deadline) list is set to be:

$$T_L(j,k,v) = \infty \text{ for all } j \quad (25)$$

The modified MC algorithm uses this list to evaluate deadline constraints. After the algorithm is finished, the list is updated with the most recent results. The updated list is then used for the simulation of the preceding (in simulated time) data item version. The process of simulation is shown in Figure 10.

$[\hat{\mathrm{L}}(k), \hat{\mathrm{R}}(k)] = Metrics(G_S, k)$
1. Initialize: $T_L(j,k,v) = \infty$ for all $j \in V[G_S]$
2. **for** $v = N_{smp}$ **downto** 1 **do**
3. $[T_E, T_L] = Event\_list(G_S, k, v, T_L)$
4. **end for**
5. Compute: $\hat{\mathrm{L}}(k)$, $\hat{\mathrm{R}}(k)$ /* Eqn. (5) */

Figure 10. MC simulation.

The simulation starts at some time in the future, and the initialization procedure assumes that there are no more data item versions generated after that moment. This means that data item (k, N_{smp}), which is evaluated first (and generated last) is successfully delivered to all nodes in the network. The metrics are calculated based on their definitions given in Section 4.

Evaluating conditions C_1 and C_2 does not increase computational complexity of the MC algorithm, because updating the deadline list can be done with $\mathcal{O}(1)$ complexity. Thus, algorithmic complexity of the modified MC algorithm remains the same: $\mathcal{O}(|E|\, log|V|)$. Note that this is the complexity per item version, while the total simulation complexity is $\mathcal{O}(N_{smp}|E|\, log|V|)$.

In order to obtain statistically significant latency estimation, propagation of a sufficiently large number of item versions has to be simulated. The metrics are obtained by averaging event times over a probabilistic ensemble of sampled graphs (Eqn. (18)). [31] shows that average shortest path distances in probabilistic graphs almost surely converge. Statistical significance of the obtained results is analyzed in Section 8.

7. STRATIFICATION-BASED PERFORMANCE ESTIMATION

MC partial simulation provides a fast way to evaluate propagation properties of a data item (from a single source). The performance of the complete system can be obtained by performing one MC simulation for each data item key (i.e., for each source node) in the network, in which case the evaluation duration is proportional to the number of keys. However, a good estimation of the network properties can be obtained by analyzing only a subset $S \subseteq K$ of data item keys and extrapolating the results. Obviously, the accuracy of such estimation depends on the right selection of the subset S.

In statistics, stratification is the process of dividing members of a population into homogeneous subgroups, *strata*, before sampling. We use a similar approach in our estimation method. A subset of data item keys is selected according to the location of their source node. We observe that data items originating at nodes with similar topological properties exhibit similar propagation patterns. We show that the so-called *closeness centrality metric* provides an accurate and efficient prediction of both latency and reliability in GWSN.

7.1 Stratification

The stratification goal is to select a subset S of data item keys, and to estimate network metric $\widehat{M}$ based on only the elements of the subset. The general idea of the proposed stratification method for performance evaluation is illustrated in Figure 11. For each data item, we calculate features, which correlate with the performance of interest of that data item. In the case of gossiping, performance is correlated to the node position in the graph and centrality measures can be used as the stratification feature (see Sections 7.2 and 8.4). Based on the calculated features, the subset S of data items is selected. A good subset should contain data item keys covering the full range of features, and mimic the feature distribution of the complete set. For each of the subset elements i, a MC partial simulation is performed and the point-to-multipoint metric $M(i)$ is calculated (Eqn. (5)). Finally, the network metric $\widehat{M}$ is estimated based on the calculated point-to-multipoint metrics $M(i)$, $i \in S$. An example of the subset selection and estimation function is given in Section 7.3, while the effect of the subset size is analyzed in Section 8.4.

The computational complexity of the stratification method depends on the complexity of the feature calculation, the complexity of the individual data-item performance evaluation, the complexity of the estimation function, and the size $|S|$ of the selected subset. Generally, the complexity of the feature calculation and the estimation is smaller than the complexity of the metric calculation. The MC simulation method proposed in Section 6 provides an efficient method for evaluating performance of individual data items. Neglecting the feature calculation and recalling that the complexity of MC is $\mathcal{O}\big(N_{smp}|E|\log|V|\big)$, the overall complexity of the network metric estimation is $\mathcal{O}\big(|S|N_{smp}|E|\log|V|\big)$. Hence, the size of the subset S provides a trade-off between the prediction accuracy and the simulation time.

Figure 11. Performance evaluation based on stratification.

7.2 Centrality measures

Centrality measures are commonly used to analyze properties of social and biological networks [32,33,34]. The network is represented as a graph, where the edges in the graph represent certain types of relations between nodes. For example, in social networks, individuals that connect two otherwise not connected groups are important because of their 'bridge' position, and centrality measures are metrics assigned to vertices or edges of the graph to capture their relative importance.

Centrality measures can capture different importance metrics in various topologies. Some of the commonly used centrality measures are: *degree*, *betweenness*, *closeness*, and *eigenvector centrality* [35]. In our work, we use closeness centrality which is known to reward nodes with better ability to access/spread gossiped information over the network [36]. It indicates how near a given node is to other nodes in the network (directly or indirectly). Our experiments show that this information has a strong correlation with GWSN performance metrics (latency, reliability, see Section 8.4).

The centrality measure is calculated based on a weighted graph $G_C(V, E, e)$ obtained from the SVGM. As before, V and E are the sets of vertices and edges of the WSN. The edge weight function $e: E \to R^+$ denotes the *expected time* necessary for node i that has received a certain data item (k, v) to deliver that data item to neighbor node j. It is calculated as an expected value of the stochastic variable corresponding to that edge. In Section 5.2., it was shown that the stochastic variables comprise two independent stochastic processes that are described with geometric distributions (Eqn. (9) and (10)). In that case, the edge weight function e for edge (i, j) can be written as:

$$e(i,j) = \frac{1}{p(i,j) \cdot p_s(i,j,k,v)} \tag{26}$$

The feature describing propagation properties of a certain data item is calculated as the closeness centrality measure of its source node. It is defined as the inverse of the sum of the lengths of the shortest paths to all other nodes in the network. Closeness centrality of some node i is defined as:

$$C_C(i) = \left(\sum_{j \in V \setminus \{i\}} d(i,j) \right)^{-1} \tag{27}$$

where $d(i, j)$ is the shortest path distance between nodes i and j in graph $G_C(V, E, e)$

Centrality-based stratification can be applied successfully in all scenarios where the computation properties are homogeneous over the set of data item keys (e.g., temperature and humidity measurements in agricultural areas [37] where all data items have the same importance). The assumption of homogeneous computation properties over the set of data item keys provides that edge weights in a graph $G_C(V, E, e)$ are data invariant, i.e., $p_s(i, j, k, v) = p_s(i, j, k', v')$, for all i, j, k, k', v, v'. Note that topological and communication properties need not be homogeneous.

The stratification approach can be adapted to certain scenarios where computational properties are heterogeneous. In that case, the stratification procedure requires an additional step where data items are classified according to the computational specifics. The classification is trivial in a situation where an application defines a few data item classes (e.g., priority) and each data item is statically assigned to one of those classes. However, when the classes are dynamic, it may not be possible to efficiently implement a stratification procedure. This kind of analysis is left for future work.

7.3 Performance estimation

The goal of centrality-based stratification is to determine a subset of data items such that properties of the complete set can be estimated accurately based only on the metrics calculated from the subset. According to the calculated features, the complete range of observed centrality values is divided into I smaller intervals (the strata), where N_i is number of data items in stratum i. From each of these strata i, we select $n(i)$ data item keys, so that $|S| = \sum_{i=1}^{I} n(i)$. Then, the network performance is estimated as the weighted average:

$$\widehat{M} = \sum_{i \in S} K_i \cdot M(i) \tag{28}$$

where K_i is a coefficient describing the importance (weight) of the sampled metric $M(i)$, often the fraction of nodes in that stratum i.

The importance coefficient of a data item key is proportional to the size of the stratum from which it is selected (i.e., the fraction of all nodes in stratum i, $N_i/|V|$) and inversely proportional to the number of data items selected from that stratum $n(i)$. Thus, we have that:

$$\widehat{M} = \sum_{i \in S} \frac{N_i}{n(i) \cdot |V|} \cdot M(i) \tag{29}$$

We use equal-size strata, with one sample per stratum. We select a node with the median centrality value within the stratum. The results show that the proposed stratification method in combination with MC partial simulation provides adequate accuracy in short analysis time frames (Section 8.4).

8. EXPERIMENTAL EVALUATION

We performed an extensive set of experiments to analyze the proposed performance evaluation techniques. We use a real deployment and simulations performed at different levels of abstraction. The simulation length was chosen such that sufficient statistical significance is achieved. In Section 8.1, we provide details of the experimental setup. The accuracy of the stochastic modeling is analyzed in Section 8.2, by comparing the MC method using the SVGM model to both low-level simulation and a real deployment. The speed of the MC method is assessed in Section 8.3, where the results of DE and MC simulation are compared. Finally, in Section 8.4 we analyze the accuracy and efficiency of the stratification method.

8.1 Experimental setup

8.1.1 Accuracy assessment of the MC partial simulation using low-level simulation

As our base of reference to study the accuracy of the modeling abstractions, we use a low-level simulation in MiXiM [21], providing full system simulation including packet processing, detailed protocol implementations (gossiping and MAC) and a path-loss channel model to determine success of radio communication. The speed of MiXiM simulation does not allow efficient exploration of large design spaces. However, the detailed models allow accurate assessment of the GWSN protocol stack and represent a good reference.

8.1.2 Accuracy assessment of the MC partial simulation using a real deployment

Our SVGM model uses the average PRR to model radio behavior. A Bernoulli process with average PRR is often used to model the radio layer [26]; however, this abstracts from the dynamics in the radio channel. The path-loss channel model used in MiXiM simulations is also a Bernoulli process with average PPR, where the PRRs are computed based on distance between nodes. In order to evaluate the applicability of our MC method to realistic scenarios we evaluate it by comparison to a real deployment.

8.1.3 Speed assessment of MC partial simulation

We evaluate the speed gain of the MC partial gossiping simulation against a full gossiping simulation implemented as a standard discrete-event (DE) simulation. Such a DE simulation assumes message transmissions as basic events and uses a functional implementation of the gossiping procedures – cache update and item selection – to accurately capture data item dissemination. To isolate the speed gain of the MC approach, we use the same stochastic abstractions for data link and radio layers in both the DE and MC simulations.

8.1.4 Accuracy assessment of the stratification method

The stratification method simulates propagation properties of a small subset of data items and estimates the full system behavior based on those results. In order to estimate the accuracy of this approach, we compare the results of a stratification-based method with results of full system simulation.

8.1.5 Grid-like topologies used in simulations

Simulation experiments are performed for different topologies and configurations. We consider rectangular topologies consisting of different combinations of rectangular parts. Such topologies correspond to some realistic scenarios, e.g., when monitoring buildings or out-door fields. Within the area, nodes are positioned in a grid-like structure with an average spacing R. The spacing R was chosen according to the transmission range of nodes; in case of the minimal transmission power, two nodes at distance R can still communicate. We refer to the structure as grid-like, because each node is allowed to drift randomly from the grid position. The maximal drift value

is chosen to be $0.4R$. The drift is selected uniformly and independently for each axis. A network is heterogeneous and consists of three types of nodes with different transmission power settings (-6dB,-12dB,-18dB). Such power settings roughly correspond to the transmission ranges R, $R\sqrt{2}$ (connecting nodes on the grid diagonals) and $2R$, respectively (using a path loss model with exponent 4). An example of a heterogeneous grid-like topology is given in Figure 12. Edges in the topology graph indicate whether nodes are in communication range.

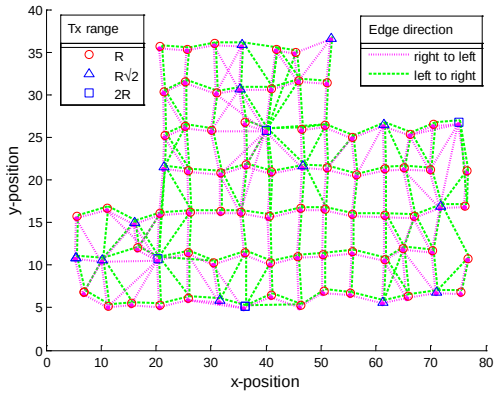

Figure 12. Grid-like network deployment.

8.1.6 Statistical significance of simulation results

The statistical significance of all simulation results is estimated using established techniques for estimation of long-run sample averages [38,39]. Each simulation consists of a certain number N_{sr} of subruns, where each subrun includes the propagation of N_{si} versions of an item and thus the total number of simulated item versions is $N_{smp} = N_{sr} \cdot N_{si}$. In each subrun z the metrics of interest $\widehat{M}(z)$, latency and reliability in our case, are calculated. After each subrun, confidence intervals of 95% and relative margins of error (half the width of the confidence interval) are calculated.

The long-run sample averages method assumes that the individual samples are independent. These samples are taken from separate subruns, which should be long enough to make their results sufficiently independent. Following a general rule of thumb, we adopted a subrun size such that samples from $N_{si} = 100$ data items are collected. Because only the propagation of the last few data items from the subrun depends on the propagation of the first few data items from the next subrun (a new data item can overtake an old one), this subrun size results in only a small dependence between two subsequent subruns.

The number of subruns has to be such that sufficient statistical significance is achieved. According to [39], a confidence interval of α requires the number of subruns $N_{sr} > (\alpha + 1)/(1 - \alpha)$. According to this guideline and taking $\alpha = 0.95$, we choose our number of subruns $N_{sr} = 39$

The total number of generated item versions $N_{smp} = 3900$ provides a sufficient statistical significance, i.e., relative margins of error on the 95% confidence interval that are well below the desired value of 5%.

8.1.7 Monte Carlo sampling – approximation of weak links

The Monte Carlo sampling of SVGM described in Section 5.2 needs to sample the intervals between packet transmissions. The required number of random samples describing the intervals between transmissions of a data item from some node corresponds to the maximal number of transmissions required to achieve success over all links originating at that node (Eqn. (11)). Note that the required number of transmissions can be arbitrarily large. Especially for weak links it can be rather high. In practice, provided that there is a sufficient number of good links, the effect of weak links on the system performance is negligible. In order to reduce simulation time, we could either remove weak links from the topology, or approximate their role in the topology based on a small number of random samples. We choose the approximation approach and impose an upper-bound on the number of random samples that are generated to describe the intervals between transmissions. When for some link more samples are required (Eqn. (12)), we consider the given link as a weak link. The weight of a weak link is approximated by reusing random samples, i.e., the missing random samples in Eqn. (12) are taken from a set of already generated samples. Our experimental results have shown that, in simulated topologies, the number of random samples describing the intervals between transmissions can be bounded to 10, without noticeable accuracy loss (0.1% for the statistical significance specified in Section 8.1.6).

8.1.8 Simulation platform

All our simulation experiments were run on a standard laptop with the CPU at 2GHz and with 2GB of RAM.

8.2 Accuracy of the stochastic abstractions in SVGM

In Section 8.2.1, we assess the accuracy of our stochastic models used in the MC method by comparison with low-level simulation. The radio behavior in a real deployment and the accuracy loss caused by using average PRR as a radio model are analyzed in Section 8.2.2.

8.2.1 Comparison with low-level simulation

The MiXiM simulation that we use as a reference simulates the full protocol stack, including packet processing, MAC protocol and radio channel (using a path-loss channel model to determine success of communication). The simulation speed of the MiXiM framework allows protocol design and system analysis; however, it is not sufficiently fast for DSE, not allowing for large sets of simulations within a reasonable time. Considering the simulation times, our experiments were limited to relatively small grid topologies, with $|V| \in \{25,36,49,64\}$ nodes. The sampling period is chosen to be $T_{smp} = \lfloor |V|/2 \rfloor$, which is a value that results in significant interaction between data item versions.

The differences between results of MC simulation and MiXiM simulation are shown as absolute relative errors in Figure 13. Each point in the graph represents the difference between the outcomes of one MC simulation run and one MiXiM simulation run for the given topology. The MC simulation results are close to the MiXiM results, showing at most 1.2% difference.

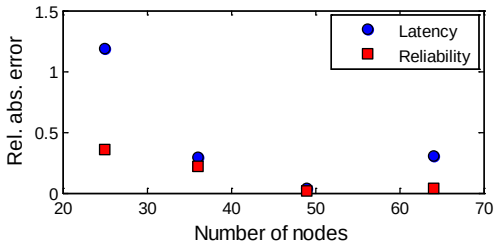

Figure 13. Accuracy of prediction, MC vs. MiXiM.

Table I compares the running times of the two simulation approaches. MiXiM simulations have simulation times in the order of minutes to hours, whereas our MC simulations do not take more than half a second. Strongly reduced simulation times are the result of the applied stochastic abstractions instead of the detailed functional models.

Table I. Running times for MiXiM and MC simulation (per subrun).

Sim/Grid	**5x5**	**6x6**	**7x7**	**8x8**
MiXiM [*min*]	26	52	94	153
MC [*s*]	0.03	0.05	0.08	0.16

From the experimental results we observe a significant speed gain from the stochastic abstractions, with a relatively small loss in accuracy. This is a good indication that stochastic modeling is the right approach for exploration of large design spaces in GWSN.

8.2.2 Comparison with a real deployment

All models we use, including the detailed MiXiM models, model the quality of the communication links in a WSN using, among other things, packet reception ratio, describing the *spatially and temporally independent* probability of successful communication over the radio channel. In this way, the average link quality is captured, while abstracting from temporal fluctuations and potential spatial correlations. In this subsection, by comparison of model results and real deployment measurements, we assess the loss in accuracy as a consequence of the chosen abstractions.

Figure 14. WSN deployment in a home indoor environment (2nd floor) [courtesy: Roessingh Research and Development]

The experiment uses an actual deployment of our case study of Section 3. The experiment is performed by deploying 57 nodes inside a fully furnished 2-floor apartment (see Figure 14), during a 10-day period and in presence of a human being performing daily activities. The nodes were mostly static, although some of the nodes exhibited limited mobility (e.g., nodes attached to doors, chairs, and home appliances). The experiment was designed to analyze the effects of short- and long-term fluctuations in the radio channel.

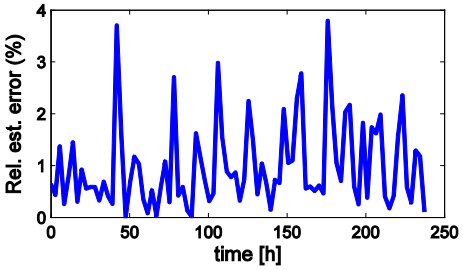

Figure 15. Error of an SVGM based performance evaluation (home indoor environment)

The effects of short-term radio fluctuations on the overall system-model accuracy are analyzed by comparing the results from the real deployment to the results obtained by the MC method using a radio model which is calibrated with the deployment measurements. While the real deployment exhibits short term fluctuations, the model uses a constant model with the same behavior on average. The experiment is divided in 1 hour time intervals, a period in which, in our experiment, the environment does not change significantly. For each 1 hour interval the average PRRs are calculated and the radio model in the SVGM is calibrated. The short-term fluctuations are most prominent in links that are not very strong and not very weak either. From the result given in Figure 15, showing the estimation accuracy for each of the time intervals, it can be observed that using the average PRR as a radio model, thus abstracting from short-term temporal fluctuations in the radio properties, does not cost much in terms of accuracy (with the error being less than 4% in all simulated time intervals, with a mean absolute error of 1%).

Ideally, we would like to perform measurements for a small period, then calibrate our simulator with the parameters extracted from the measured data and then use the calibrated simulator to estimate performance for arbitrary periods in the future. However, even in networks with static nodes, the environment changes (time of day, temperature and humidity, the level of human activity, movement of furniture, presence of various sources of electromagnetic interference, etc) leading to long-term fluctuations in radio properties. The effects of long-term changes in the environment are analyzed by comparing the metrics obtained from a time interval of data and metrics obtained from the model calibrated with the PRRs corresponding to an interval of data measured at a different time of day. Our experimental results show that the maximal estimation error increases from 4% to 12% in that case (calculated over all pairs of one hour intervals). Analysis of the largest deviations shows that these are caused by movement of furniture, causing structural changes in the network topology.

An illustration of long-term fluctuations in the radio channel is given in Figure 16. The PRRs are averaged over hourly intervals and the measurements during a 24h period are shown in the figure. The results are plotted for all links originating from a single node. We show them as

box-plot diagrams. For each box, the central line represents the median and the edges of the box are the 25 and 75 percentiles. The dashed line denotes the range including all data points not considered outliers (99.3% coverage assuming the data is normally distributed); data points outside this range are plotted individually and denoted with crosses. About two thirds of the links exhibit only small fluctuations (they are either good or bad links). However, the links in the gray area (almost one third of the links) change significantly over time. The reason is that these links are more sensitive to environmental changes, such as the presence of a person or movement of furniture. The changes in these links do not affect network operation, since, in our deployment, nodes still have a sufficient number of stable links; however the performance metrics change.

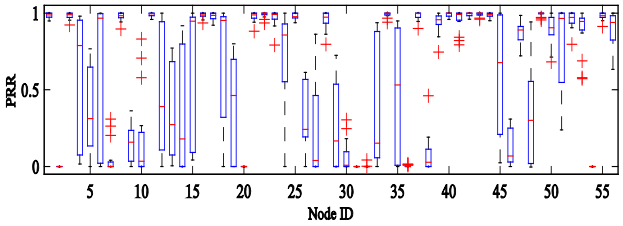

Figure 16. Long-term fluctuations in the radio channel

In conclusion, the experiment with the real deployment shows that the use of PRRs to abstract from radio behavior is accurate as long as there are no (or only very limited) structural changes in the network or its environment. For structural changes in environment or topology, we need either a periodic recalibration of PRRs, or a model that can capture those changes (e.g., Markov models as described in [6]).

8.3 MC simulation

The MC and DE simulations we perform use the same abstractions to model the communication properties of the system (data link and radio layer), but the evaluation of those abstractions differs. The simulated events in the case of DE simulation are related to the packet transmissions, and are evaluated in each communication round. On the other hand, the events in MC simulation represent the reception of a new version of a data item by a node for the first time. In this case the abstractions are used to sample the time between events (according to Eqn. (12)) and compute tentative event times (Section 6). Besides that, MC simulation does not simulate the gossiping protocol in a discrete-event fashion, but rather uses a combination of Monte Carlo simulation and shortest-path analysis to capture the gossiping behavior. Both DE and MC simulations capture the same stochastic processes and in the limit the metrics converge almost surely to the same value. In this section, first, we experimentally verify this proposition, then we compare the speed of the MC and DE simulations.

8.3.1 Accuracy

Experiments are performed for five different network sizes $|V| \in \{83,155,251,371,515\}$. The number of generated random topologies is 240 per network size (1200 in total). The radio model is a probabilistic unit disk radio model [5], with probabilities selected from uniform distribution $U(0.1,1)$. Experiments are performed for two sampling periods, both chosen to provide significant interaction between data items, $T_{smp} = \lfloor |V|/2 \rfloor$ and $T_{smp} = |V|$.

MC simulation is performed for each data item separately and results are compared with the DE simulation. Estimation deviations are calculated as an average value of deviations of all point-to-multipoint metrics, for each of the network sizes. The box-plots presented in Figure 17 show the obtained results. The estimation deviation is below 1% in all simulated cases, while the deviation median and mean are close to 0%. This indicates that we have a random error with a small variation and almost no bias.

Although in the limit both simulations provide the same result, the underlying modeling and simulation processes are not entirely the same. For example, MC simulation does not suffer from the transient period at the simulation startup when selection probabilities are not yet stable because caches are not yet fully filled, but it does suffer from some transient behavior at the end, because it assumes that the last sampled data items arrive at every node. Transient periods affect only a few data items (first and last), and the effect is negligible in our experiments due to the large number of simulated data items (3900 for each source node).

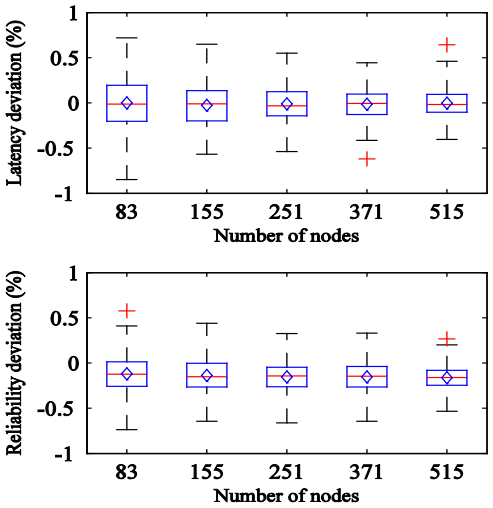

Figure 17. Deviation between MC and DE simulation: (top) latency; (bottom) reliability.

8.3.2 Speed

In order to illustrate the speed gain of the MC simulation, we compare the running times of MC and DE simulations (Table II). As expected, MC simulation provides short average times necessary to estimate propagation of a single data item (only 24.4ms for a subrun of 100 data items in the network of 515 nodes). DE simulation provides no benefits in a situation when dissemination properties of data items from a single source are evaluated; it simultaneously simulates propagation of data items from all sources. For comparison, in Table II, besides running times of DE and MC simulations, we present the "total" MC simulation time as the time required to perform simulation for data items from all source nodes ($|V| \cdot T_{MC}$).

Table II. Running times (in seconds) for DE and MC simulations.

Sim/Size	T_{smp}	83	155	251	371	515
T_{DE} [s]	$\|V\|/2$	0.46	2.33	8.14	22.3	45.6
	$\|V\|$	0.87	4.24	14.7	39.3	87.1
$\|V\|\cdot T_{MC}$ [s]	$\|V\|/2$	0.21	0.72	1.71	3.95	7.32
	$\|V\|$	0.31	1.11	2.99	6.47	12.6
T_{MC} [s]	$\|V\|/2$	0.003	0.005	0.007	0.011	0.014
	$\|V\|$	0.005	0.007	0.012	0.017	0.024

The results show that even in the case when all data items are simulated, MC simulation achieves shorter running times than DE simulation (2-3 times for 83 nodes and about 6-7 times for 515 nodes). This may look counter-intuitive considering the fact that events in DE simulation inherently contribute to the analysis for multiple data items, while in the MC simulation for each data item a new stochastic ensemble is selected. There are several factors that contribute to the observed behavior: DE simulation simulates data items from all sources in parallel and uses a functional implementation of the gossiping protocol (i.e., the cache update and the item selection procedures) while the MC simulation is a partial system simulation that extracts dissemination properties of a single data item from the shortest-path analysis, thus avoiding the complexity of the protocol implementation; events in DE simulation are at a finer granularity than in MC simulation and although in MC simulation all data items are simulated separately the total number of events turns out to be similar.

The total number of events and thus the simulation time depends on the data item sampling period. This dependency is analyzed through the experiments performed for the two values of the sampling period ($T_{smp} = \lfloor |V|/2 \rfloor$ and $T_{smp} = |V|$). Based on the results presented in Table II we can observe the following:

1) The running time of DE simulation is almost directly proportional to the sampling period. This is the expected result since the number of events in DE simulation is proportional to the number of communication rounds. For a fixed number of simulated data item versions N_{smp}, the number of simulated rounds is directly proportional to the sampling period, $N_{smp} \cdot T_{smp}$.

2) The running time of the MC simulation also depends on the sampling period. However, this dependency is more complex. An event in the MC algorithm denotes a situation when a node receives a data item for the first time, and thus the total simulation time is proportional to the number of data-item arrivals. A smaller sampling period implies a more significant interaction between data item versions and thus lowers the number of nodes that receive that specific version of the data item (i.e., reliability); consequently, it lowers the number of events and the simulation time. On the other hand, for all sampling periods greater than a certain value, system reliability is close to one and the number of events does not change.

The sampling period used in the experiments provides a reliability of about 40% ($T_{smp} = \lfloor |V|/2 \rfloor$) and about 65% ($T_{smp} = |V|$). It is not difficult to conclude that the speed advantage of MC simulation will further increase with an increase in the sampling period (more drastically once the reliability becomes close to one).

8.4 Stratification-based performance analysis

The performance of the stratification approach is analyzed through the same set of 1200 experiments used in the previous subsection. Recall that topologies are irregular and heterogeneous, two aspects that might be challenging for stratification.

Our stratification method is based on closeness centrality. We use equal-size strata, with one subset sample per stratum. From each stratum, a node with the median value within the stratum is selected as its representative. From the selected nodes, network metrics are estimated (Eqn. (29)). Four different subset sizes (number of strata) are analyzed, containing 5, 10, 15, and 20 data items (nodes).

The time required for the centrality-based stratification is directly proportional to the number of selected data items. For each data item in the subset, one instance of MC simulation has to be performed. Compared to the MC running time, the time required to compute centrality metrics is negligible. The running times per data item of MC simulation have been given in Table II. The accuracy of the centrality-based stratification is evaluated by comparison with the results of the full system simulation. It is presented using box-plots of estimation errors for network latency and network reliability in Figure 18. Results are sorted horizontally according to both network size and subset size as follows. The first four results are for network size 83, and subset sizes 5, 10, 15, and 20; the last four results are for network size 515, and subset sizes 5, 10, 15, and 20 (so 'a1' denotes the combination of network size 83 and subset size 5, while 'e4' denotes network size 515 and subset size 20).

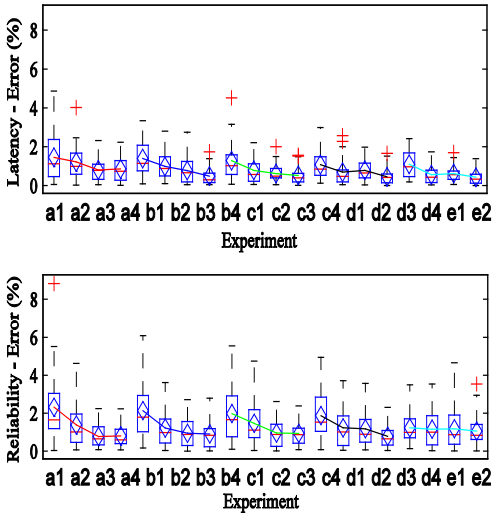

Figure 18. Accuracy of stratification.

As an expected trade-off, there is a general trend that the estimation error of the stratification approach decreases with an increase in the number of elements in the selected subset. Figure 18 shows that even with only 5 samples (a selected subset of 5 data items), the estimation errors are below 5% for latency and below 6% for reliability in all simulated topologies (the mean error is 1.5% for latency and 2.5% for reliability).

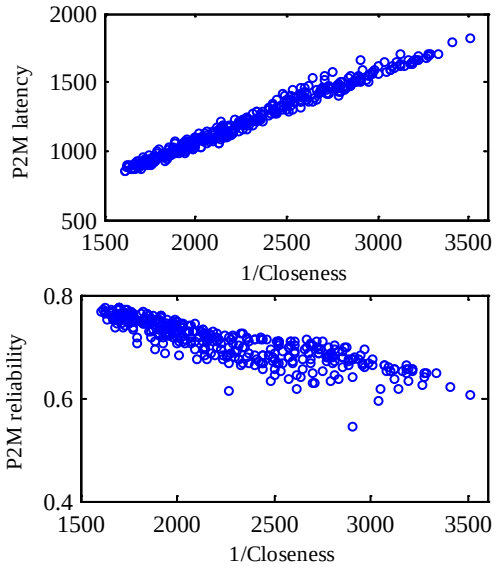

Figure 19. Correlation between closeness and point-to-multipoint (P2M) metrics (random topology with 371 nodes).

Figure 19 illustrates correlation between the closeness centrality measure and performance metrics latency and reliability. It confirms our assumption about the good correlation between centrality values and the metrics. Closeness centrality correlates slightly better with point-to-multipoint latency than with reliability, which is the main reason for the observed difference in the estimation accuracy.

It is interesting to observe that an increase in network size does not increase the estimation error, and the same number of strata provides the same (or even a better) accuracy with an increase in the network size (Figure 18). It may seem counter-intuitive that estimation based on a smaller amount of information leads to the same or an improved accuracy. This can be explained however by the tendency of gossiping to even out the point-to-multipoint metrics for nodes that share similar topological properties. This effect is stronger with an increase in network size when more nodes have similar topological properties, and it compensates for the decrease in the relative amount of available information in the stratification method.

Finally, the quality of the centrality-based stratification strategy is evaluated by comparing it to a random stratification strategy. The number of randomly selected data items has been chosen the same as for centrality-based stratification, i.e., 5, 10, 15, and 20. For each simulated topology and each subset size, the random stratification procedure is performed 100 times. The results are given in Figure 20. If we compare results presented in Figures 18 and 20 it can be seen that the overall accuracy of centrality-based stratification is noticeably better than the accuracy of random stratification. The gain obtained by centrality-based stratification is larger for latency than for reliability. This is not unexpected since correlation between latency and centrality is stronger than correlation between reliability and centrality.

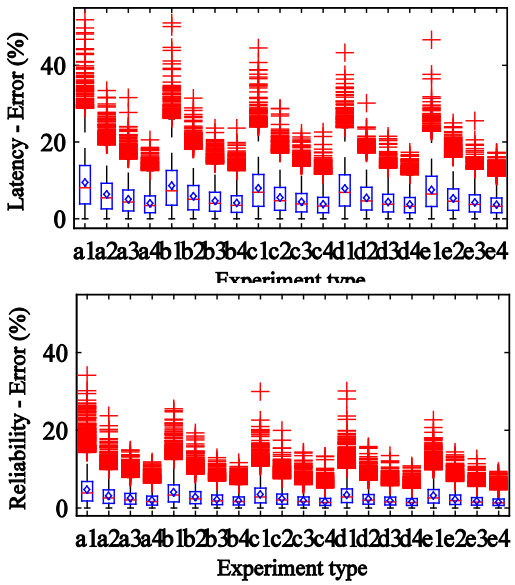

Figure 20. Accuracy of estimation based on randomly selected data items.

We may conclude that centrality-based stratification in combination with MC simulations provides a fast and accurate way to analyze system-level properties of GWSN and is sufficiently fast to support the exploration of large design spaces.

9. CONCLUSION

This paper presents an integral modeling approach for gossip-based WSN (GWSN) that allows fast analysis of system-level properties and is suitable for DSE (Design-Space Exploration). Two metrics are evaluated: latency and reliability. In case other metrics, such as for example energy consumption, vary due to the stochastic nature of a WSN, we expect our method to be extensible to such metrics, capturing for example network lifetime as well. An experiment with a real deployment shows that the chosen models may be simple, yet capture relevant behavior well. The stochastic abstraction of the radio model requires calibration, either directly via a prototype deployment or indirectly via an analytic radio model. Considering results with respect to the accuracy of analytic radio models reported in [14,5,27], the latter appears to be a promising approach.

The core of our method is a Monte Carlo (MC) simulation technique to evaluate the propagation of an individual data item in the network; this provides significant speedup in case a complete system simulation is not required. Instead of simulating protocols and interactions between different data items, the MC approach extracts dissemination properties of a single data item from a shortest-path abstraction. The short running times of MC simulation make it the preferred solution in all situations when individual point-to-(multi)point metrics have to be obtained in a fast but accurate way. Examples are when one is interested in the performance of only a subset of nodes or data items, such as the latency for high-priority data samples, or in worst-case performance aspects such as the longest delay in the network. Although MC is targeted to partial system simulation, it turns out to also be faster than standard discrete-event simulation for full-system simulation.

Building on the MC simulation technique, we proposed a centrality-based stratification method that estimates the complete network performance through the simulation of only a representative subset of data items. Experimental results show that a small set of source nodes selected based on closeness centrality provides good performance estimation. The accuracy depends on the number of selected source nodes but increase in accuracy is not linear; the increase levels off with the number of selected nodes.

The applicability of the simulation techniques proposed in this paper depends on the possibility to describe random effects concisely and accurately by stochastic variables. The modular approach proposed in this paper makes the models easily tunable for changes in system parts, including changes in the application running on top of the gossiping. In future work, we plan to explore the possibility for capturing dynamic aspects such as structural time-dependent link quality variations, interference, moving obstacles, and node mobility. Our approach may be compatible, for example, with the approach to capture link variations through Markov models as described in [6]. In parallel, we plan to develop DSE techniques such as those of [14,4] based on our models.

List of main symbols

T_{smp}	*Sampling period*
(k, v)	*Data item with key* k *and version* v
$C_i(k, t)$	*Entry of the cache of node i corresponding to data item* k *at time* t
$T_E(i, k, v)$	*Arrival time of data item* (k, v) *at node* i
$D_i(k, v)$	*Delivery time of data item* (k, v) *at node* i
$U_i(k, v)$	*Delivery success of item* (k, v) *at node* i
$L_i(k)$	*Point-to-point latency between source* k *and destination* i
$L^D(k)$	*Point-to-multipoint latency between source* k *and set* D *of destinations*
$L^{K,D}$	*Network latency between set* K *of sources and set* D *of destinations*
$R_i(k)$	*Point-to-point reliability between source* k *and destination* i
$R^D(k)$	*Point-to-multipoint reliability between source* k *and set* D *of destinations*
$R^{K,D}$	*Network reliability between set* K *of sources and set* D *of destinations*
$G_{SV}(V, E, X)$	*Stochastic-variable graph with nodes* V *of the gossiping network, edges* E *of links in the network and stochastic edge weights* X
p_r	*Packet reception ratio*
p_{mac}	*MAC success rate*
p_s	*Selection probability*
$X_{k,v}(i, j)$	*Stochastic propagation delay of data item*(k, v) *over link* (i, j)
$N_{Tx}(i, j, k, v)$	*Number of transmissions of data item* (k, v) *from node* i *to node* j *until first success*
$T_{Tx}(m, i, k, v)$	*Length of the* m*-th intertransmission interval for data item* (k, v) *at node* i
N_{max}	*Number of transmissions until the data item is delivered to all neighbors*
N_S	*Number of data items that are gossiped by each node in one round*
G_S	*Sampled graph obtained from a stochastic variable graph*
$T_{IE}(i, j, k, v)$	*Inter-event time, the time between arrival of item* (k, v) *to node* i *and successful delivery to node* j
$T_S(k, v)$	*Sampling time of item* (k, v)
$\boldsymbol{G}_k$	*Stochastic ensemble of weighted graphs sampled from the stochastic variable graph for source* k *for all versions*
$T_L(i, k, v)$	*Latest delivery time (deadline before being overtaken by later version) for data item* (k, v) *at node* i
$C_C(i)$	*Closeness centrality of node* i

ACKNOWLEDGEMENTS

The authors would like to thank the anonymous reviewers for their constructive comments that significantly contributed to improved presentation of the paper. This work was supported in part by the Dutch innovation program Point-One, through project ALwEN, grant PNE07007.

REFERENCES

[1] D. Gavidia, M. van Steen, A probabilistic replication and storage scheme for large wireless networks of small devices. In Proc. 5th IEEE Int'l Conf. Mobile and Ad Hoc Sensor Systems (MASS).

[2] R. Karp, C. Schindelhauer, S. Shenker, and B. Vocking, Randomized rumor spreading, In Proc. FOCS 2000, IEEE, 2000.

[3] K.P. Ferentinos,.T.A. Tsiligiridis, Adaptive design optimization of wireless sensor networks using genetic algorithms. In Journal of Computer networks, vol.51, pp. 1031-1051, 2007.

[4] M. Nabi, M.Blagojević, T. Basten, M. Geilen, T. Hendriks, Configuring Multi-Objective Evolutionary Algorithms for Design-Space Exploration of Wireless Sensor Networks. PM2HW2N 2009, ACM, 2009.

[5] A. Cerpa, J.L. Wong, L. Kuang, M. Potkonjak, D. Estrin, Statistical model of lossy links in wireless sensor networks. In Proc. of IPSN 2005, ACM/IEEE. 2005.

[6] A. Kamthe, M. Carreira-Perpinan, A. Cerpa, M&M: Multi-level Markov Model for wireless link simulations. In Proc. SenSys 2009, ACM, 2009.

[7] K.G. Langendoen, Medium Access Control in Wireless Sensor Networks. Chapter in Medium Access Control in Wireless Networks, Nova Science Publishers, 2008.

[8] K.G. Langendoen, A. Meier, Analyzing MAC Protocols for Low Data-Rate Applications. ACM Trans. on Sensor Networks 7(2):1-40, 2010.

[9] R. Bakhshi, L. Cloth, W. Fokkink, B. Haverkort, Mean-Field Analysis for the Evaluation of Gossip. In Proc.QEST 2009, IEEE Computer Society pp. 247-256, 2009.

[10] R. Bahkshi, D. Gavidia, W. Fokkink, M. van Steen, A Modeling Framework for Gossip-based Information Spread In Proc. QEST 2011, IEEE Computer Society, 2011.

[11] J. Polley, D. Blazakis, J. McGee, D. Rusk, J.S. Baras, Atemu: A fine-grained sensor network simulator. In Proc. SECON '04, pp.145-152, IEEE 2004.

[12] R. Lanotte, M. Merro, Semantic Analysis of Gossip Protocols for Wireless Sensor Networks. In Proc. CONCUR 2011, LNCS 6901, pp. 156-170, Springer, 2011.

[13] R. Bakhshi, D. Gavidia, W. Fokkink, M. van Steen. An Analytical Model of Information Dissemination for a Gossip-based Protocol. Computer Networks, vol. 53(13), August 2009.

[14] A. Guinard, A. McGibney, D. Pesch, A Wireless Sensor Network Design Tool to Support Building Energy Management. In Proc. 1st ACM Workshop On Embedded Sensing Systems For Energy-Efficiency In Buildings. ACM, 2009.

[15] M. Kwiatkowska, G. Norman, D. Parker. Analysis of a Gossip Protocol in PRISM. ACM SIGMETRICS Performance Evaluation Review, 36(3), pp. 17-22, 2008.

[16] A. Fehnker, P.Gao, Formal Verification and Simulation for

Performance Analysis for Probabilistic Broadcast Protocols. In Proc. ADHOC-NOW, pp.128-141, 2006.
[17] T. Hérault, R. Lassaigne, F. Magniette, S. Peyronnet. Approximate Probabilistic Model Checking. In Proc. 5th Int. Conf. on Verification, Model Checking and Abstract Interpretation (VMCAI'04), Springer, 2004.
[18] http://www.isi.edu/nsnam/ns/
[19] http://www.omnetpp.org/
[20] http://www.opnet.com
[21] A. Köpke, M. Swigulski, K. Wessel, D. Willkomm, P. Klein, T. Haneveld, T. Parker,, O. Visser, H. Lichte, S. Valentin, Simulating Wireless and Mobile Networks in OMNeT++ The MiXiM Vision. OMNeT++ Workshop, ICST, 2008.
[22] M. Blagojević, M.Nabi, T. Hendriks, T. Basten, M.C.W. Geilen, Fast simulation methods to predict wireless sensor network performance. In Proc. PE-WASUN 2009, ACM, pp. 41-48, 2009.
[23] P.A. Anemaet, Distributed G-MAC: A flexible MAC protocol for servicing gossip algorithms. Master's thesis, Delft University of Technology, The Netherlands, 2008.
[24] L. van Hoesel, P. Havinga, A lightweight medium access protocol (LMAC) for wireless sensor networks, in 1st Int. Workshop on Networked Sensing Systems, Tokyo, Japan, 2004
[25] F. van der Wateren, The art of developing WSN applications with MyriaNed. Tech. report, Chess Company. the Netherlands, 2008.
[26] A. Woo, D. Culler, Evaluation of Efficient Link Reliability Estimators for Low-Power Wireless Networks. Technical Report No. UCB/CSD-03-1270. University of California, Berkeley, 2003.
[27] M. Zuniga, B. Krishnamachari, Analyzing the transitional region in low power wireless links. In Proc. SECON '04, pp.517-526. IEEE, 2004.
[28] J. Misra, A walk over the shortest path: Dijkstra's Algorithm viewed as fixed-point computation, Information Processing Letters, Vol. 77 (2-4), pp. 197-200, 2001.
[29] E.W. Dijkstra, A note on two problems in connexion with graphs. Numerische Mathematik, pp. 269–271,1959.
[30] T. Cormen, C. Leiserson, R. Rivest, C. Stein, Introduction to Algorithms (2nd ed.). MIT Press, 2001.
[31] H. Frank, Shortest paths in probabilistic graphs. Operations Research, vol.17, pp. 583-599, 1969.
[32] P. Bonacich, Power and Centrality: A Family of Measures, The American Journal of Sociology, 92 (5), pp. 1170-1182, 1987.
[33] L. C. Freeman, Centrality in social networks: Conceptual clarification. Social Networks, 1(3), 215-239, 1979.
IEEE, 2008.
[34] H. Jeong, S.P. Mason, A.L. Barabasi, Z. Oltvai, Lethality and centrality in protein networks. Nature 411(6833), pp. 41-42, 2001.
[35] B. Junker B., F. Schreiber, Analysis of Biological Networks. Wiley Series in Bioinformatics, 2008.
[36] E. Yoneki, P. Hui, J. Crowcroft, Wireless epidemic spread in dynamic human networks. Bio-Inspired Computing and Communication, LNCS 5151, pp. 116–132, Springer, 2008.
[37] K.G. Langendoen, A. Baggio, O. Visser, Murphy loves potatoes: Experiences from a pilot sensor network deployment in precision agriculture. In: 14th Int. Workshop on Parallel and Distributed Real-Time Systems, WPDRTS 2006, .IEEE, 2006.
[38] D. Zwillinger, S. Kokoska, Standard probability and statistics tables and formulae. Chapman&Hall,1999.
[39] J.Y. Le Boudec. Performance Evaluation Lecture Notes (Methods, Practice and Theory for the Performance Evaluation of Computer and Communication Systems). EPFL, Switzerland, 2010.
[40] S. Bosch, M. Marin-Perianu, R.S. Marin-Perianu, P.J.M. Havinga, H.J. Hermens, Keep on Moving! Activity Monitoring and Stimulation Using Wireless Sensor Networks. In: Proc. 4th European conference, EuroSSC, 2009, pp. 11-23. 2009.
[41] A. Boukerche, A. Nakamura, A. Loureiro, Algorithms for Wireless Sensor Networks: Present and Future. Chapter in Algorithms and protocols for wireless sensor networks, John Wiley & Sons, 2008.
[42] R. Hoes, T. Basten, C.K. Tham, M. Geilen, H. Corporaal, Quality-of-service trade-off analysis for wireless sensor networks, Performance Evaluation, Vol. 66, Issues 3-5, pp.191-208, 2009.
[43] K. Wong. QoS Measurements and Experimental Validation of a Wireless Sensor Network Model. MSc graduation paper, Eindhoven University of Technology, The Netherlands, 2010