Dense Multi-Channel Sniffing in Large IoT Networks

Jelena Crnogorac, Jovan Crnogorac, Malisa Vucinic, Enis Kočan, Thomas Watteyne

Dense Multi-Channel Sniffing in Large IoT Networks

JELENA CRNOGORAC [1], (Graduate Student Member, IEEE), JOVAN CRNOGORAC [1], MALIŠA VUČINIĆ [2], (Member, IEEE), ENIS KOČAN [1], (Member, IEEE), AND THOMAS WATTEYNE [2], (Senior Member, IEEE)

[1]Department of Telecommunications, University of Montenegro, 81000 Podgorica, Montenegro
[2]Inria, 75012 Paris, France

Corresponding author: Jelena Crnogorac (jelena.kovac@ucg.ac.me)

In this article we deal with the issue of network traffic monitoring in large multi-channel wireless IoT networks. Assuming known link conditions on all radio channels, i.e. connectivity matrix defined through Packet Delivery Ratio on all frequencies and all links between nodes, we propose two methods for defining the number and the positions of sniffer devices, with the goal to maximize the capture of network traffic. Method I is based on probabilistic theory and assumes brute-force search over the connectivity matrix for defining the optimal positions of a given number of sniffers, or, for a given percentage of the traffic to be captured as the input parameter, this method determines number of sniffers and their locations. Due to the computational complexity of brute-force search of the connectivity matrix, we complement Method I and propose Method II. Method II is based on graph theory and uses the minimal Packet Delivery Ratio on each link as the input parameter for defining the number and position of sniffers. We input traffic traces from an experimental testbed into the network to examine and compare both methods. Results show that the Method I outperforms Method II in the percentage of captured network traffic, for a given number of deployed sniffers. However, Method II complements Method I in scenarios where there are a large number of sniffers, due to lower computational complexity.

Sniffer, multi-channel IoT network, network monitoring.

I. INTRODUCTION

A sniffer is a device used to capture network traffic. At its core, it consists of a radio receiver allowing it to listen to a specific frequency. Some sniffers have multiple receivers, which allows for multi-channel sniffing: the sniffer can listen to several frequencies, i.e. radio channels, simultaneously and report the captured traffic. This is particularly important for radio technologies such as time-slotted channel hopping (TSCH) [1], a technique in which packet exchanges happen on multiple frequencies to combat multipath fading and external interference.

We witness the expansion of wireless sensor networks in different areas, like industry, agriculture, environmental monitoring, smart grids, smart cities, military. For a reliable operation of large wireless sensor networks, and network debugging, it is of great importance to be able to perform efficient network traffic monitoring. Besides, in the process of developing and testing novel protocols, it is necessary to obtain an insight into the exchanged network traffic. For this purpose, a sniffer is an irreplaceable tool.

Networks that operate in a star topology can typically be covered with a single sniffer. This is not the case for larger networks, that are based on a multi-hop or mesh topology. Having a single sniffer to cover the area of a multi-hop wireless network is not feasible due to the radio propagation peculiarities: propagation loss, limited sensitivity of the sniffer's receiver, low transmit power of the nodes, blocking objects in surrounding environment in some cases and internal interference in the network. In addition, propagation conditions vary significantly across different frequencies:

A good link between two nodes on one frequency may be a poor communication link on another frequency [2].

We hence need to add more sniffers to be able to capture the frames exchanged in the network. This article focuses on answering the following questions: *How many sniffers should one add? Where should those sniffers be placed?* The stochastic behavior of the radio channel makes answering these questions hard.

In this article, we propose and analyze different techniques for sniffer placement with the goal of capturing mesh network traffic. We develop two methods, simply called Method 1 and Method 2. Method 1 searches the network connectivity matrix, looking for the combinations of sniffers that maximizes the coverage percentage of the network, based on known packet delivery ratio values in the network. Method 2, based on graph theory, takes a heuristic approach and determines the position and the number of sniffers that satisfy the dominating set of the network graph. The algorithms are adapted for different use cases and can serve the network administrators to either cover the network with a parameterizeable loss in coverage with a minimal number of sniffers, or find the best locations for a given number of sniffers.

The remainder of this article is organized as follows. Section II surveys the related work on multi-channel sniffing. Section III gives background and presents the common evaluation setup. Section IV describes and evaluates Method I. Section V describes and evaluates Method II. Section VI compares both methods using real-world traces combined with simulation. Finally, Section VII concludes this article.

II. RELATED WORK

Different techniques have been proposed for sniffing.

Shin *et al.* [3] develop a distributed online algorithm for the optimal sniffer-channel assignment for passive monitoring in multi-channel wireless networks, using single-channel sniffers. They intend to maximize the total amount of traffic captured by sniffers, by properly assigning frequencies to sniffers. Every node has a weight factor that is higher if the node transmits a larger volume of data. The algorithm proposes a decision based on the total weight factor, taking into account the fact that every node must be covered with at least one sniffer. The authors evaluate the performance in two types of networks: random and scale-free networks. In random networks, sniffers are randomly deployed; in scale-free networks, nodes with the highest degrees are chosen as sniffers.

Xu *et al.* [4] investigate the sniffer channel assignment problem for passive monitoring in multi-channel wireless networks. Passive monitoring utilizes sniffers to overhear the exchanged packets between nodes; sniffers are independent of the communication system. The authors propose the redundant sniffer deployment strategy in unreliable monitoring conditions. By unreliable monitoring conditions they assume channel fading effects and the imperfect monitoring by sniffers. The goal is to maximize the probability of capturing data packets. They develop a learning policy to estimate the traffic load on each frequency, while making sniffer-channel assignment decisions. Based on the load estimate, the resulting algorithm decides how many sniffers monitor each frequency, assuming single-channel sniffers.

Zeng *et al.* [5] study network delays in WSNs, as an important quality metric for real-time control applications. They develop two algorithms for sniffer placement inside the architecture: Max-flow Sniffer Placement and Max-degree Sniffer Placement. The former is an approximation algorithm that utilizes max-flow formulation. The latter relies on the assumption that a candidate sniffer with a larger degree – a sniffer that has a greater number of network connections with other nodes – can monitor more nodes and hence may reduce the number of sniffers needed.

Luo *et al.* [6] propose a deep reinforcement learning approach for a spatial sniffer-channel assignment problem in wireless networks without advanced information about users' activities. Spatial sniffer-channel assignment problem considers assigning each wireless sniffer a proper operating channel and location to detect data packets in the target network. A sniffer is considered mobile and can only monitor the traffic data on a single channel. Based on the traffic measurements collected from all sniffers, the central entity estimates the spatial distribution of data traffic and specifies the channel and location for each sniffer. The central entity may also enable the sleep mode of a sniffer to save energy consumption.

The mentioned works assume that the network traffic is not equally distributed on all channels, which is the case in TSCH networks we consider. The authors dominantly assume single-channel sniffers. Contrary to them, we assume multi-channel sniffers, which are commercially available. For example, the BeamLogic device[1] receives on all 16 radio channels in the 2.4 GHz band simultaneously. We consider that each frequency is equally loaded, as it is, for example, in TSCH networks. We further assume sniffers can be placed at the same location of existing nodes in the network. In this article, we propose two methods for determining the position and number of sniffers in the multi-hop multi-channel networks: a brute-force approach based on probabilistic theory and a heuristic approach based on graph theory. For both approaches, the input parameter is the connectivity matrix on all frequencies. The resulting solution can be used in different ways. One example is, for a given number of sniffers, determine their positions. Another is, given target percentage of received packets, determine the number of sniffers.

III. BACKGROUND AND EVALUATION SETUP

WSN protocols and standards must be extensively tested and validated before being applied to real-world deployments. Testbeds represent a necessary tool for experimental testing. They offer the possibility of loading new firmware onto the nodes, debugging their behavior, measuring energy consumption, storing logs, etc. One representative of an open access

[1] http://www.beamlogic.com/

testbeds is FIT IoT-LAB [7], a large-scale IoT testbed composed of 2728 low-power wireless nodes, deployed in 6 sites across France. FIT IoT-LAB offers access to IoT devices that users can program with arbitrary firmware, launch repeatable experiments, and gather results. In the phase of testing network performance on experimental testbeds, it is common practice to use sniffers to capture network traffic. To find the optimal number of sniffers and their locations in the network it is necessary to have an insight into the network connectivity.

Mercator[2] is a tool that collects connectivity traces in a deployment site, a testbed or real-world deployment. It consists of two parts: firmware that is fleshed on all nodes and software running on a computer. Nodes exchange packets on all channels while the computer collects data on the quality of links in the observed network. The computer calculates PDR values based on the ratio of sent and successfully transmitted packets. An output file of Mercator contains information about the quality of links between all nodes on all frequencies used in the network [8]. The collected connectivity trace represents the starting point for our research: the connectivity matrix. Each cell in the connectivity matrix represents a link between two nodes with a corresponding PDR on a certain frequency.

We will evaluate our proposed solution within the 6TiSCH Simulator [9], a discrete-event simulator written in Python that implements the 6TiSCH protocol stack, a time-slotted channel hopping (TSCH) radio technology standardized by the Internet Engineering Task Force. It focuses on modeling the network from the MAC layer. The protocol messages in the simulator carry only semantically-relevant parameters: Exchanged messages are not byte-accurate. 6TiSCH Simulator can use connectivity traces collected in experimental testbed platforms or real-world deployments [8]. This allows the replay of previously collected real-world connectivity traces instead of randomly generated connectivity matrices with the radio propagation model. In this way, it is possible to accurately represent physical layer phenomena in the propagation of radio waves, such as multipath fading, which is one of the major drawbacks of the simulation-based analysis.

We extended the 6TiSCH Simulator by introducing a new type of device: *a sniffer*. The sniffer type of device does not send packets and listens to network traffic on all 16 frequencies in the 2.4 GHz band simultaneously. Table 1 lists the simulation parameters. We use the standard box plot to plot results, each point being simulated 100 times. We will evaluate our solution on random mesh and trace-based topologies. The trace-based topology was recorded with Mercator tool on the experimental testbed platform FIT IoT-lab Grenoble, which uses 384 M3 open nodes and 288 A8 nodes. For our research, we use the connectivity matrix recorded in a part of the network with 50 nodes [8]. We simulated the network operation 100 times, each simulation lasting 5000 TSCH slotframes.

[2]https://github.com/openwsn-berkeley/mercator

TABLE 1. Simulation parameters.

Parameter	Value
number of runs	100
number of nodes	50
number of slotframes	5000
scheduling function	MSF
topology	Random, K7

IV. METHOD I: SNIFFER PLACEMENT BASED ON BRUTE-FORCE CONNECTIVITY MATRIX SEARCH

Method I uses the connectivity matrix of the network, which contains Packet Delivery Ratio (PDR) values among all nodes over all frequencies. The goal is to determine the number and position of sniffers. We assume that PDR values are stable over time, i.e. they do not change during the process of sniffer selection. This assumption is true when there is no external interference in the network, and when the nodes and the environment are stationary. Sniffers can listen for packets simultaneously on all frequencies, and can only be positioned at existing node locations. After the algorithm suggests positions for sniffers, the goal is to put the sniffer on the chosen node's location. This way, a sniffer has approximately the same quality of radio links as the node at its location. Because of the proximity between the sniffer and the node at the same location, communication between them is limited only by internal interference and is considered ideal, i.e. with a PDR of 1. We assume that the nodes in the network generate approximately the same amount of traffic, as is the case when all nodes are of the same kind. We further assume that an equal amount of traffic is generated at each frequency, as in TSCH networks. If these assumptions are not satisfied, the analysis can be extended with a weighted average approach, similar to Shin *et al.* [3].

First, we analyze the network with a single sniffer. For every node in the network – every potential location for the sniffer – the algorithm calculates the probability of packet reception averaged over the number of nodes and frequencies, as expressed in Eq (1).

$$P(s) = \frac{1}{N}\frac{1}{F}\sum_{i=1}^{N}\sum_{f=1}^{F}(P_{i,j,f}) \tag{1}$$

Eq. (1) models the probability that the single sniffer receives a frame transmitted by an arbitrary node in the network, on an arbitrary frequency. In (1), N is the number of nodes in the network, F is the number of frequencies the nodes can communicate on. $P_{i,j,f}$ is the PDR value of the link between node i and node j on frequency f. Eq. (1) represents the sum of the probabilities of successful receptions between the observed and other nodes in the network, on all frequencies, averaged over the number of frequencies and nodes. This probability is derived from the connectivity matrix, and represents a measure of the quality of a given location for placing the sniffer node. The algorithm positions the sniffer at the location of the node with the highest computed probability.

In a network with multiple sniffers, we first determine all possible combinations of sniffer positions based on the positions of the nodes in the network, then perform a brute-force search to determine the node combination with the highest average probability of packet reception. Eq. (2) represents the number of combinations of sniffers' positions, S is the number of sniffers. It is equal to the binomial coefficient of the total number of nodes in the network and the number of available sniffers.

$$Number\ of\ combinations = \binom{N}{S} = \frac{N!}{S!(N-S)!} \tag{2}$$

Eq. (2) defines the number of combinations without repetition: only one sniffer can be placed at a given node location. For a network with multiple sniffers, the probability of packet reception is expressed in (3). Eq. 3 represents the probability that a packet sent from any node on any frequency is received by at least one of the S sniffers.

$$P(s) = \frac{1}{N}\frac{1}{F}\sum_{i=1}^{N}\sum_{f=1}^{F}(1 - \prod_{j=1}^{S}(1 - Pi,j,f)) \tag{3}$$

Method I operates in two modes. In the first mode, given a number of sniffers, the algorithm computes the locations for those sniffers that yield the maximal probability of packet reception. In the second mode, given a target percentage of the received frames, the algorithm computes a minimal number of sniffers and their locations.

Alg. 1 details the first mode of Method I. It considers all possible combinations given a number of sniffers, and determines the one that maximizes the percentage of received traffic. It loops through all defined combinations and initializes the average probability to 0. Then, it loops through all network nodes, all frequencies, and all the nodes from the current combination and calculates the probability of packet reception error between the node from the combination and the current node on the considered frequency. On line 8, it converts the probability of error into the probability of successful reception. After that, it calculates the probability of packet reception averaged over all the frequencies (line 10) and the probability that packet from arbitrary node in the network will be successfully received by at least one sniffer from the selected combination (line 12). The algorithm outputs *sniffers_locations*, the locations of the sniffers which yields the maximal probability of packet reception.

Alg. 2 details the second mode of Method I. The input parameter, besides the connectivity matrix, is the target probability of successful packet reception. There is no limitation on the number of proposed sniffers, but the goal is to find the solution with a minimal number of sniffers that yields at least the target probability. The algorithm passes through a list of integers from 1 to N, where N is the number of nodes in the network. After that, as in Alg. 1, the algorithm calculates probabilities of successful packet reception. It iterates until it finds a combination that yields at least the target probability. It outputs the number and position of sniffers. Looping starts from the lowest number of sniffers (one sniffer) then increases until it finds a solution that meets the target probability; this approach results in a minimal number of sniffers satisfying the target probability of successful packet reception.

Algorithm 1 Method I, First Mode

```
   Input: Connectivity matrices G_1 ... G_16,
          number_of_sniffers
   Output: sniffers_locations, average_pdr
1  all_combs= combinations(nodes, number_of_sniffers);
     // Loop through all combinations for defined
     number of sniffers
2  for combination in all_combs do
3      pdr_sum_comb = 0;
       // Loop through all nodes in the network
4      for node in nodes do
           // Loop through all frequencies
5          for frequency in frequencies do
               // Loop through all sniffers from
                  defined combination
6              for sniffer in combination do
                   // Probability of reception error
7                  p_err = p_err*(1-
                   get_pdr(node,sniffer,frequency));

               // Probability of successful
                  reception
8              p_success = 1 - p_err;
9              pdr_sum_sens = pdr_sum_sens + p_success;
10         avg_pdr_sens = pdr_sum_sens/16;
11         pdr_sum_comb = pdr_sum_comb +
            avg_pdr_sens;
12     avg_pdr_comb = pdr_sum_comb /
       number_of_nodes;
13     if avg_pdr_comb > max_avg_pdr_comb then
14         best_comb = combination;
15         max_avg_pdr_comb = avg_pdr_comb;
16     sniffers_locations = best_comb;
17     average_pdr = max_avg_pdr_comb;
```

A. EVALUATION

Fig. 1 shows the percentage of the received packets in the network, for the given number of sniffers as an input parameter. In such a case, the method proposes locations of the sniffers in order to maximize the received amount of network traffic. With two sniffers in the FIT/IoT-LAB Grenoble topology, the percentage of received packets ranges between a minimum of 43% and a maximum of 60%, with a median value for the hundred conducted experiments of 51%. With five sniffers, the percentage of received traffic ranges between a minimum of 80% and a maximum of 83%, with a median value of 82%. With two sniffers in Random topology, it is possible

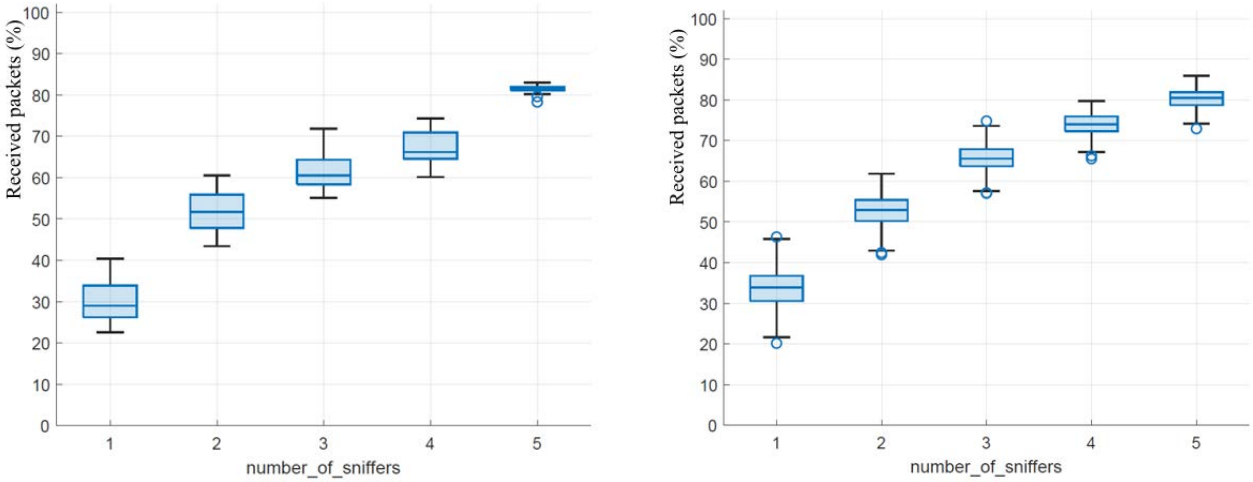


FIGURE 1. Percentage of received packets for different values of parameter *number_of_sniffers* for a) FIT/IoT-LAB Grenoble topology and b) Random topology.

to receive about 54% of network packets, and deploying 5 sniffers would result in 81% of network traffic reception.

Fig. 2 shows percentage of received packets as a function of the input parameter *target_pdr*, for the FIT/IoT-LAB Grenoble topology and the Random topology. The dashed red line passes through the points where the required percentage of received packets (10%, 15%, 20%, 95%) are located. As parameter *target_pdr* increases, the percentage of received packets also increases. In both considered network topologies, similar results on traffic reception are obtained. In randomly generated topologies, the percentage of received packets constantly increases with *target_pdr*, while in FIT/IoT-LAB Grenoble we can notice deviations from the linear characteristic. Since trace-based simulation uses only one topology, we can see the impact of individual characteristics of the FIT/IoT-LAB Grenoble network in the graph. These fluctuations are settled for randomly generated topologies because the simulator generates a new topology for each of the 100 simulations.

V. METHOD II: SNIFFER PLACEMENT BASED ON DOMINATING SET

Method I relies on a brute-force search over all combinations of potential sniffer locations. In an example network of 50 nodes and 10 sniffers, there are over ten billion possible sniffer deployment combinations. Number of combinations increases exponentially with the number of sniffers, which poses a limitation for a higher number of sniffers: It becomes computationally infeasible to determine sniffer locations. For example, the computations necessary to find the location for 6 sniffers with Method I lasted several days on Intel Xeon 5218 machine running at 2.4 GHz. We propose Method II[3] to overcome the computational infeasibility of Method I for a large number of sniffers.

Method II relies on the graph theory to determine the number of sniffers and suitable locations for their placement. Similarly to Method I, we assume that the connectivity matrix of the network is already known, and its values are stable in time. For potential sniffer locations, we consider only the positions of the existing nodes in the network. Once a sniffer is placed on a selected node location, the assumption is that the sniffer has the same quality of links with other nodes, as the node at whose position the sniffer was deployed.

We model the network on each frequency as a graph whose vertices are nodes and edges are the radio links between them. We obtain the edge weights as the PDR values from the connectivity matrix on a given frequency. Edges whose weights are less than the input parameter *sniffer_link_pdr* are discarded.

Definition 1: A dominating set for a graph G = (V, E) is a subset D of V such that every vertex not in D is adjacent to at least one member of D. Dominating set D of graph G is minimal if there does not exist a subset of D that is a dominating set for a graph G.

In one graph, there can be several minimum dominating sets (DS). Fig. 3 shows an example graph with six nodes and possible dominating sets marked in blue.

Our algorithm selects potential locations for sniffers on one frequency by determining the minimal dominating set for the created auxiliary graph. The algorithm then repeats the process on all frequencies, and the union of the proposed locations represents the output parameter of the first part of Method II. The created list, called sniffer candidates, is a dominating set over all the auxiliary graphs on different frequencies. We depict this in Algorithm 3.

In scenarios with significant differences in connectivity matrices on different frequencies, the union of minimal dominating sets over the different frequencies results in a large

[3]The description of Method II we present in this article is an extended version of Kovač *et al.* [10]. This article extends the description of the original algorithm and presents novel results obtained on FIT IoT-LAB Grenoble topology.

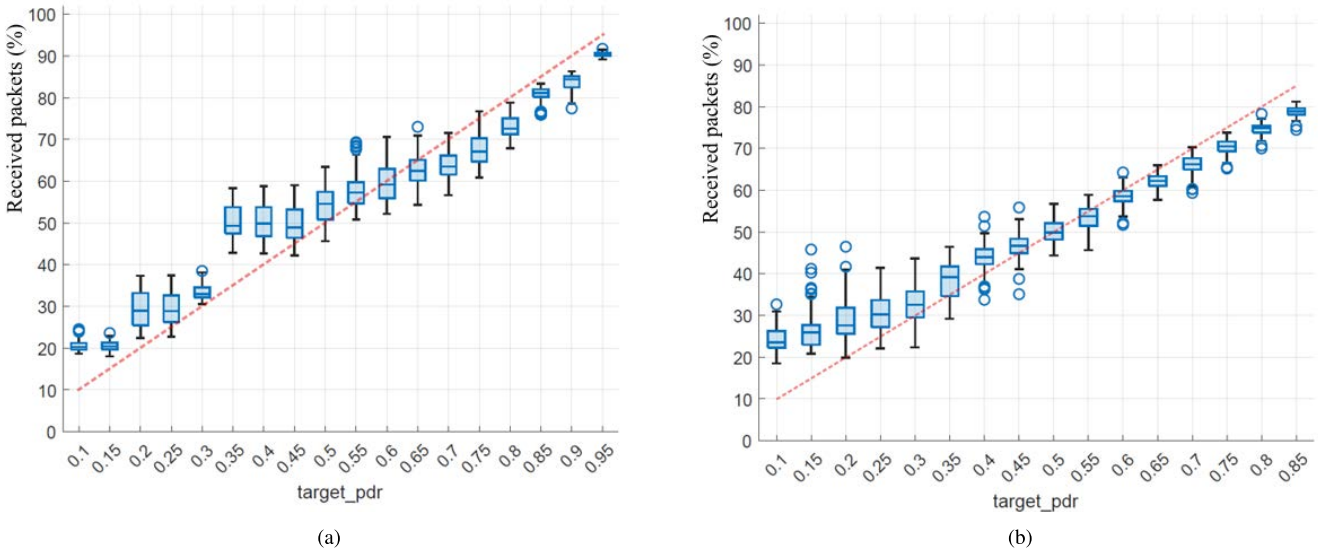


FIGURE 2. Percentage of received packets for different values of parameter *target_pdr* for a) FIT/IoT-LAB Grenoble topology and b) Random topology.

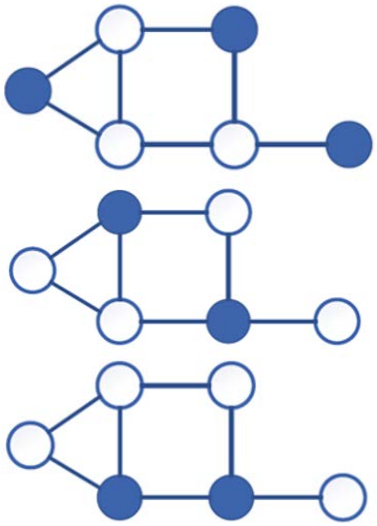


FIGURE 3. Illustration of the dominating sets for an example graph.

number of proposed sniffers. The union of minimal dominating sets over the different frequencies is thus not optimal as it introduces redundancy on almost all, or all given frequencies. Method II further takes a heuristic approach and attempts to minimize the number of sniffers in an opportunistic fashion. We define the input parameter, *removal_load*, as a decimal number from 0 to 1. When the value is set to 0, the algorithm simply returns *sniffer_candidates* from Algorithm 3. When the value is set to 1, the algorithm attempts to minimize the number of sniffers while keeping the dominant set on all frequencies. The values between 0 and 1 relate to the percentage of sniffers (0 to 100%) the algorithm attempts to remove from the initial sniffer candidate list.

In Algorithm 4, we first calculate the target number of sniffers to remove based on the input parameter *removal_load*. After that, we sort the list of sniffer candidates in ascending order by the sum of the PDR values on all links and all frequencies. The higher this quality measure, the more packets a sniffer placed at a given location will capture. In that way, the algorithm removes the sniffer with the lowest quality first and checks whether the new graph is still DS for all auxiliary graphs, i.e. for all frequencies. The considered sniffer cannot be removed if the new proposal is not DS on any of the created graphs. Reaching the desired number of sniffers or passing to the end of the list interrupts the process. Method II proposes a list of sniffer locations and calculates the estimated average probability of packet reception. The estimated average probability of packet reception is calculated as in Method I. First, we calculate the probabilities of successful receptions between nodes from the proposed sniffer locations and all nodes in the network on each frequency. After that, we calculate the average probability of packet reception on all 16 frequencies and the average probability that packets from all nodes in the network will be successfully received by at least one sniffer from the combination selected by Algorithm 4.

A. EVALUATION

We evaluate Method II on trace-based topologies obtained on FIT IoT-LAB Grenoble testbed. For result over random topologies, the reader is referred to our previous work [10]. We study the impact of the input parameters *sniffer_link_pdr* and *removal_load* on the number of sniffers and percentage of received packets.

In the following, we define a ‘‘uniquely received packet’’ as the packet that was received by exactly one sniffer. We use the term ‘‘multiple received packet’’ to denote a packet that was received by more than one sniffer. The distribution of uniquely vs. multiple received packets in a network is

Algorithm 2 Method I, Second Mode

Input: Connectivity matrices $G_1 \ldots G_{16}$, target_pdr
Output: sniffers_locations, average_pdr

```
for number_of_sniffers in 1. N do
    all_combs = combinations(nodes,
      number_of_sniffers); // Loop through all
      combinations for defined number of
      sniffers
    for combination in all_combs do
        pdr_sum_comb = 0;
        // Loop through all nodes in the network
        for node in nodes do
            // Loop through all frequencies
            for frequency in frequencies do
                // Loop through all sniffers from
                   defined combination
                for sniffer in combination do
                    // Probability of reception
                       error
                    p_err = p_err*(1-
                      get_pdr(node,sniffer,frequency));
                // Probability of successful
                   reception
                p_success = 1 - p_err;
                pdr_sum_sens = pdr_sum_sens +
                  p_success;
            avg_pdr_sens = pdr_sum_sens/16;
            pdr_sum_comb = pdr_sum_comb +
              avg_pdr_sens;
        avg_pdr_comb = pdr_sum_comb /
          number_of_nodes;
        if avg_pdr_comb ≥ target_pdr then
            target_comb = combination;
            target_avg_pdr_comb = avg_pdr;
            break
sniffers_locations = target_comb;
average_pdr = target_avg_pdr_comb;
```

Algorithm 3 Part I: Selection of the Sniffers That Covers the Deployment

Input: Connectivity matrices $C_1 \ldots C_{16}$, sniffer_link_pdr
Output: sniffer_candidates

```
sniffer_candidates = empty_set;
for frequency in 1. 16 do
    // Graph G_s contains links greater than
       sniffer_link_pdr
    G_s = make_graph(C_frequency, sniffer_link_pdr);
    singleChLoc = min_dominating_set(G_s);
    sniffer_candidates = sniffer_candidates ∪
      singleChLoc;
```

Algorithm 4 Part 2: Reducing the Number of Sniffers From Initial sniffer_candidates

Input: Auxiliary Connectivity Graphs $G_{a1} \ldots G_{a16}$, sniffer_candidates, removal_load
Output: reduced_candidates

```
// Calculate the target number of sniffers
target_sniff_num =
length(sniffer_candidates)*(1-removal_load);
// Graph G_s contains nodes from the parameter
   sniffer_candidates
G_s = make_graph(sniffer_candidates);
sniffer_candidates = order_by(sniffer_candidates);
for sniffer in sniffer_candidates do
    // Remove node sniffer from graph G_s
    G_s.remove_node(sniffer);
    // Check if the new graph is still
       dominating set
    if not is_dominating_set(G_s, G_a1 ... G_a16) then
        // Put back node in the graph
        G_s.add_node(sniffer);
    // Abort the removal process if we reach the
       target num
    if length(G_s) ≤ target_sniff_num then
        break
reduced_candidates = nodes from G_s;
```

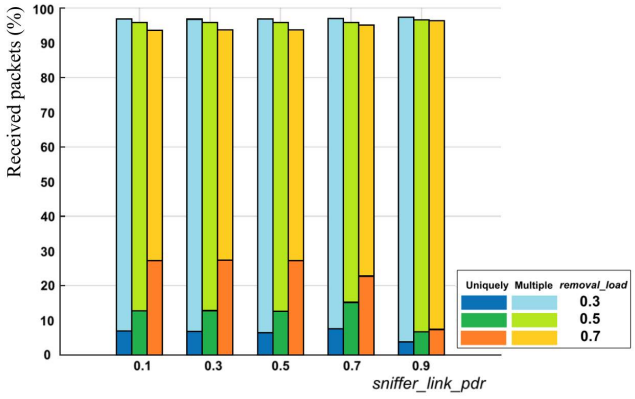
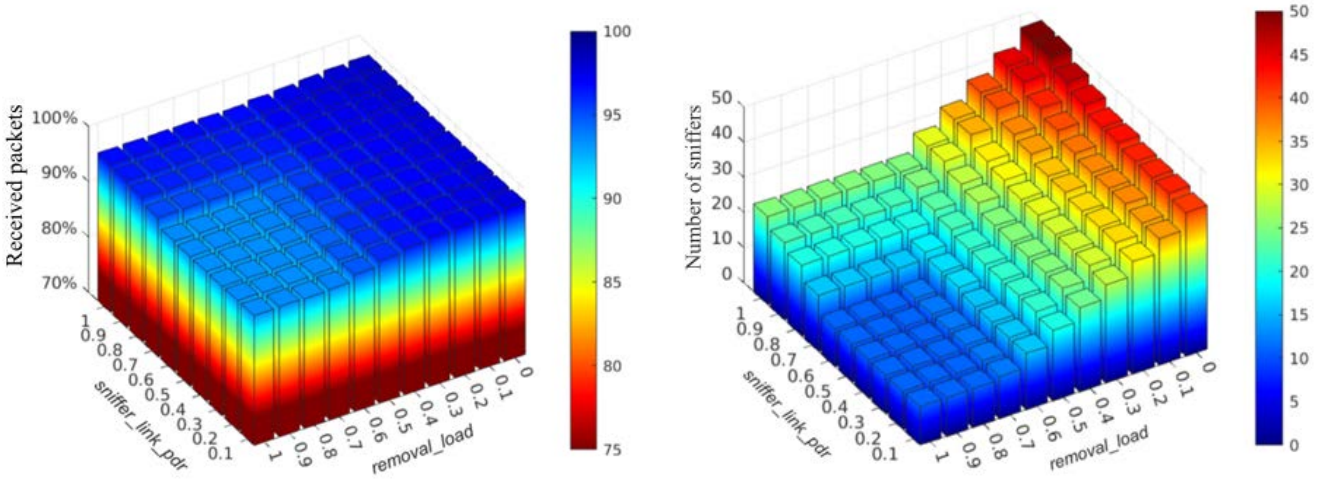


FIGURE 4. Percentage of uniquely and multiple received packets for different values of input parameters in FIT/IoT-LAB Grenoble topology.

important because it gives better insight into the parts of the network where more than one sniffer receives the same traffic. Fig. 4 shows the percentage of uniquely and multiple received packets. Increasing the *snifer_link_pdr* parameter, the percentage of received packets increases, but so does the number of multiple receptions. For high percentages of network traffic reception (above 95%), over 90% of received packets are multiple received. As a sidenote, we remark that this can make the analysis of the recorded traffic difficult because it is necessary to filter multiple received packets [11].



(a) (b)

FIGURE 5. **Percentage of received packets (a) and number of sniffers (b) as a function of `sniffer_link_pdr` and `removal_load` parameters.**

Fig. 5 shows the percentage of received packets and the number of sniffers for the different values of input parameters in trace-based topology. When the parameter *removal_load* is 0, the required number of sniffers is greater than 40 for all values of the *sniffer_link_pdr* parameter. Our opportunistic removal approach takes advantage of the correlation between the different frequencies: With the increase of the *removal_load* parameter, the number of sniffers decreases. The minimum number of sniffers in this topology is 11, for the *removal_load* parameter equal to 1 and *sniffer_link_pdr* equal to 0.1.

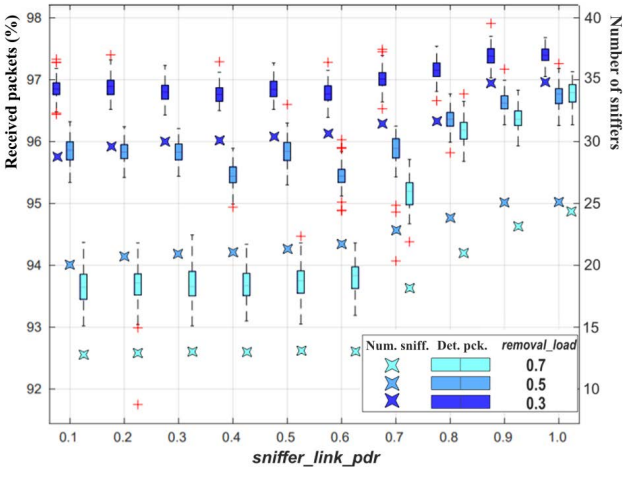


FIGURE 6. **Percentage of received packets and the average number of sniffers for different values of input parameters for trace-based topology.**

Fig. 6 shows the relation between the percentage of received packets and the average number of sniffers. We can see that ten or more sniffers are required for capturing more than 90% of network traffic, even for *removal_load* parameter equal to 0.7. The percentage of traffic reception ranged between a minimum of 93% and a maximum of 94.5%, with a median value for the hundred conducted experiments of 93.5%, with 13 sniffers deployed, input parameters *sniffer_link_pdr* between 0.1 and 0.7 and *removal_load* 0.7. With 20 sniffers and input parameters *sniffer_link_pdr* 0.1, *removal_load* 0.5, the percentage of traffic reception ranged between a minimum of 95.5% and a maximum of 96.5%, with a median value of 95.8%.

VI. DISCUSSION

In the following, we compare the two methods by plotting the percentage of received packets in the network as a function of the average number of sniffers. Fig. 7 shows the result for FIT/IoT-LAB Grenoble (a) and randomly-generated topologies (b). Fig. 7 a) shows that using 1 to 5 sniffers, enables capturing of less than 80% of the network traffic. Higher number of sniffers, more than 10, allows us to capture above 90% of the traffic. Fig. 7 a) also shows than no overlapping zone for the two methods exists. On the one hand, it was computationally infeasible to calculate sniffer locations for more than 6 sniffers using Method I. On the other hand, the output of Method II using the minimal dominating set resulted in a higher number of sniffers (more than 10) due to the characteristics of the radio connectivity traces obtained in a real-life testbed.

Fig. 7 b) shows the same result obtained in a randomly-generated topology. We note that the topology has a slight effect on the percentage of received packets. Similarly to Fig. 7 a), we can note that by using 1 to 5 sniffers in a network, we can receive up to 80% of the traffic. Fig. 7 b) also shows a slight advantage of Method I over Method II in the marked zone, where, with the same number of sniffers, Method I achieves a higher percentage of received packets.

The advantage of Method I in the percentage of received packets in comparison with Method II is expected. Method I

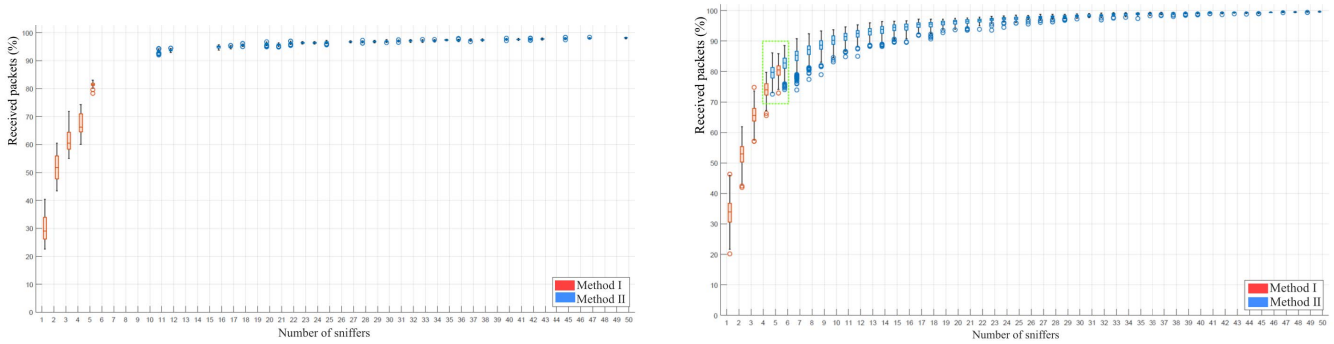


FIGURE 7. Relation between the average number of sniffers and the average percentage of received packets for Method I and II for a) FIT/IoT-LAB Grenoble topology and b) Random topology.

finds the optimal combination of sniffer locations for a given number of sniffers at the cost of higher computational load. However, using Method I becomes computationally infeasible for a higher number of sniffers. We overcome this limitation by the introduction of Method II, which complements Method I for a higher number of sniffers.

VII. CONCLUSION

Deployment expansion of wireless IoT networks in different areas, brings to focus the issue of network traffic monitoring, not only for the purpose of developing and testing of novel protocols and wireless technologies, but also for reliable operation of the existing IoT networks. This issue is particularly challenging in large multi-channel wireless IoT networks, which motivated us to propose and examine two different methods for determining number and positions of sniffers in such IoT networks. The Method I, based on probabilistic theory, implements brute-force search over the known connectivity matrix and enables defining locations of the sniffers in large multi-channel IoT networks, for the given number of sniffers, as well as defining number and positions of the sniffers in case where the lowest PDR value is used as the input parameter. We examine the performance of the pro-posed method using the 6TiSCH Simulator, with randomly generated topologies, as well as using the captured traces of the network traffic on experimental FIT IoT-LAB testbed in Grenoble, with 50 nodes. The obtained results have approved Method I as a reliable solution, which enables capturing of more than 50% of network traffic with only 2 sniffers and more than 80% of the traffic with 5 sniffers in multi-channel, multi-hop IoT network. However, computational complexity in brute-force-based Method I puts the limits on practical implementations of this solution for more than 5 sniffers in the considered IoT network with 50 nodes. Indeed, com-putations to obtain the optimal combination of sniffers in the network, with 6 sniffers using Method I, took on the order of days on a machine running at 2.4 GHz. The similar computation for determining the output of Method II using a larger number of sniffers took less than an hour. Thus, the approach based on graph-theory (Method II) complements the proposed Method I solution. Method II usually results in a large number of the proposed sniffers to be implemented, due to the union of all dominating sets for each frequency. Thus, we further improve the proposed Method II, with the option to remove some of the originally proposed sniffer positions, while still keeping dominating sets (DS) on all frequencies. Criteria for selection of the sniffer that can be removed is the sum of PDR values on all the links from the considered sniffer, to its neighboring nodes, on all frequencies. Implementing this criteria enables Method II to reach performance only several percent lower than Method I, in terms of volume of received network traffic.

Further research work may include modelling coexistence of different communication technologies, which introduce interference on certain frequencies.

REFERENCES

[1] X. Vilajosana, T. Watteyne, T. Chang, M. Vučinić, S. Duquennoy, and P. Thubert, "IETF 6TiSCH: A tutorial," *IEEE Commun. Surveys Tuts.*, vol. 22, no. 1, pp. 595–615, 1st Quart., 2020.

[2] K. Brun-Laguna, P. Minet, T. Watteyne, and P. H. Gomes, "Moving beyond testbeds? Lessons (We) learned about connectivity," *IEEE Pervasive Comput.*, vol. 17, no. 4, pp. 15–27, Oct. 2018.

[3] D.-H. Shin, S. Bagchi, and C.-C. Wang, "Toward optimal distributed monitoring of multi-channel wireless networks," *IEEE Trans. Mobile Comput.*, vol. 15, no. 7, pp. 1826–1838, Jul. 2016.

[4] J. Xu, S. Gong, Y. Zou, W. Liu, K. Zeng, and D. Niyato, "Redundant sniffer deployment for multi-channel wireless network forensics with unreliable conditions," *IEEE Trans. Cognit. Commun. Netw.*, vol. 6, no. 1, pp. 394–407, Mar. 2020.

[5] W. Zeng, J. Cote, X. Chen, Y.-A. Kim, W. Wei, K. Suh, B. Wang, and Z. J. Shi, "Delay monitoring for wireless sensor networks: An architecture using air sniffers," *Ad Hoc Netw.*, vol. 13, pp. 549–559, Feb. 2014.

[6] X. Luo, M. Wang, C. Zeng, C. Li, J. Xu, and S. Gong, "Optimization-driven deep reinforcement learning for sniffer patrolling in wireless networks," in *Proc. Int. Wireless Commun. Mobile Comput. (IWCMC)*, May 2022, pp. 760–765.

[7] C. Adjih, E. Baccelli, E. Fleury, G. Harter, N. Mitton, T. Noel, R. Pissard-Gibollet, F. Saint-Marcel, G. Schreiner, J. Vandaele, and T. Watteyne, "FIT IoT-LAB: A large scale open experimental IoT testbed," in *Proc. IEEE 2nd World Forum Internet Things (WF-IoT)*, Dec. 2015, pp. 459–464.

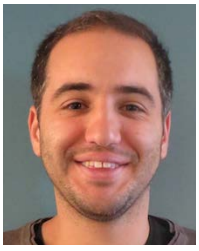
[8] Y. Tanaka, K. Brun-Laguna, and T. Watteyne, "Trace-based simulation for 6TiSCH," *Internet Technol. Lett.*, vol. 3, no. 4, p. e162, Jul. 2020.

[9] E. Municio, G. Daneels, M. Vučinić, S. Latré, J. Famaey, Y. Tanaka, K. Brun, K. Muraoka, X. Vilajosana, and T. Watteyne, "Simulating 6TiSCH networks," *Trans. Emerg. Telecommun. Technol.*, vol. 30, no. 3, p. e3494, Mar. 2019.

[10] J. Kovac, J. Crnogorac, E. Kocan, and M. Vucinic, "Sniffing multi-hop multi-channel wireless sensor networks," in *Proc. 28th Telecommun. Forum (TELFOR)*, Nov. 2020, pp. 1–4.
[11] J. Crnogorac, J. Kovac, E. Kocan, and M. Vucinic, "D-Argus: A distributed IEEE 802.15.4 sniffer," in *Proc. 27th Telecommun. Forum (TELFOR)*, Nov. 2019, pp. 1–4.

JELENA CRNOGORAC (Graduate Student Member, IEEE) received the M.Sc. degree in electronics, telecommunications, and computer science from the Faculty of Electrical Engineering, University of Montenegro, Podgorica, in 2022, where she is currently pursuing the Ph.D. degree. She has been working as a Teaching Assistant with the Faculty of Electrical Engineering, University of Montenegro, since November 2021. Her main research interests include wireless sensor networks and the Internet of Things (IoT) protocols and applications.

JOVAN CRNOGORAC received the B.Sc., Spec.Sci., and M.Sc. degrees in electronics, telecommunications, and computer science from the University of Montenegro, in 2017, 2018, and 2022, respectively. Since June 2022, he has been working as an IT Specialist with Crnogorski Telekom, a part of Deutsche Telekom Group—the leading telecommunications operator based in Podgorica, Montenegro. His main research interests include low-power wireless technologies and the IoT.



MALIŠA VUČINIĆ (Member, IEEE) received the Engineering degree from the University of Montenegro, in 2010, the joint master's degree (Hons.) from the Politecnico di Torino and the Grenoble Institute of Technology, in 2012, and the Ph.D. degree from Grenoble Alps University, in 2015. He is currently a Research Scientist with the AIO Team, Inria Research Institute, Paris, France. In parallel with the Ph.D. studies, he was a Research Engineer with STMicroelectronics, and a Visiting Scholar with the University of California at Berkeley, in 2015. He was the Co-Chair of the IETF LAKE Working Group which standardizes a lightweight authenticated key exchange protocol for the Internet-of-Things use cases. He led the security standardization work in IETF 6TiSCH, specifying the Constrained Join Protocol (CoJP) for 6TiSCH Networks. He is a Core Developer of the OpenWSN Project, the reference 6TiSCH Implementation, and a Co-Lead of the 6TiSCH Simulator. He is fluent in English, French, Italian, and Montenegrin.

ENIS KOČAN (Member, IEEE) received the M.Sc. and Ph.D. degrees in telecommunications from the University of Montenegro, in 2005 and 2011, respectively. Part of the Ph.D. Research Enis has conducted at the Aristotle University of Thessaloniki. He is currently an Associate Professor with the Faculty of Electrical Engineering, University of Montenegro. He has published more than 80 scientific papers in international journals and peer reviewed conferences. His research interests include digital communications over fading channels, with particular emphasis on solutions for beyond 5G networks, the IoT wireless communication solutions, and EMF exposure from wireless communication systems. He was a Recipient of the Best Paper Award at the International Conference on Wireless Personal Multimedia Communications (WPMC 2013), held in the frame of the Global Wireless Summit, in 2013.

THOMAS WATTEYNE (Senior Member, IEEE) is an insatiable enthusiast of low-power wireless technologies. He holds a Research Director position at Inria in Paris, in the AIO research team, where he leads a team that designs, models and builds networking solutions based on a variety of Internet-of-Things (IoT) standards. He is a Wireless System Architect at Analog Devices, the undisputed leader in supplying low power wireless mesh networking solutions for critical applications for industrial and beyond. Since 2013, he co-chairs the IETF 6TiSCH working group, which standardizes how to use IEEE802.15.4e TSCH in IPv6-enabled mesh networks, and is member of the IETF Internet-of-Things Directorate. In 2019, he co-founded Wattson Elements, the company that develops the award-winning Falco marina management solution. Thomas was a postdoctoral research lead in Prof. Kristofer Pister's team at the University of California, Berkeley. He founded and co-leads Berkeley's OpenWSN project, an open-source initiative to promote the use of fully standards-based protocol stacks for the IoT. Between 2005 and 2008, he was a Research Engineer at France Telecom, Orange Labs. He holds the Ph.D. in computer science (2008), the M.Sc. in networking (2005), and the M.Eng. in telecommunications (2005) from INSA Lyon, France. He is fluent in four languages.