Ad Hoc Networks 91 (2019) 101870

Contents lists available at ScienceDirect

Ad Hoc Networks

journal homepage: www.elsevier.com/locate/adhoc

Design of network coding based reliable sensor networks

E. AL-Hawri*, N. Correia, A. Barradas

Center for Electronic, Optoelectronic and Telecommunications (CEOT), Faculty of Science and Technology - University of Algarve, Faro, 8005-139, Portugal

ARTICLE INFO

Article history:
Received 3 August 2018
Revised 2 April 2019
Accepted 17 April 2019
Available online 18 April 2019

Keywords:
Wireless sensor network
Network coding
Overlay network
CoAP
RELOAD
Node placement

ABSTRACT

Network coding can be very useful in achieving a balance between energy efficiency and end-to-end packet error in sensor networks, particularly if only a subset of the nodes act as encoding nodes. In this article, a mathematical model and a heuristic algorithm are proposed to plan for the best placement of encoding nodes while ensuring reliability. These approaches are also able to address scenarios where sensor networks, using different gateways, are federated. In this case a distributed storage system is required to ensure the recovery of packets when related coded packets arrive to the different gateways. The experimental results show that the adequate number and placement of encoding nodes can be effectively determined, significantly enhancing the performance of constrained sensor networks using network coding.

1. Introduction

In a near future, sensing devices are expected to integrate many applications (e.g., health care, environmental) and in many cases these will have to communicate using wireless sensor networks (WSN) [1]. Such constrained networks usually have energy efficiency and end-to-end packet error rate concerns, which are competing goals (e.g., a packet may be sent through multiple paths to reduce packet error rate but this increases energy consumption). Network coding can be very useful in such scenarios, allowing a balance between these two goals to be achieved in an elegant way [2].

Although network coding has been applied to wired and wireless mesh networks, many developments in these networks cannot be applied to sensor networks. More specifically, network coding has been used mainly for throughput increase, or efficient use of bandwidth, in one-to-many traffic flow scenarios where the same content is distributed to multiple clients. Sensor networks, contrarily to wireless mesh networks, have the following characteristics [2,3]:

- There are multiple sources sending data notifications to gateway/sink nodes, meaning that the traffic flow is mainly many-to-one;
- Energy efficiency is the main concern rather than bandwidth, since nodes typically produce small volumes of data;
- The cost of transceivers is a concern in large-scale deployments;
- In tree-based many-to-one approaches, widely used in WSNs, messages may not reach the sink although alternative viable paths exist (trees are built based on past network conditions, and alternative paths may not have been discovered yet).

For all these reasons, bandwidth efficiency is often sacrificed to achieve power and cost efficiency [3]. This is why the authors of [2] propose a network coding based collection protocol, called SenseCode, having as design goal the balance between low packet error rate and energy efficiency. Packet error rate is defined as the fraction of messages generated by the sources that are not successfully communicated to the destination, capturing the ability of the protocol to deliver the original data in the face of packet loss. A tree rooted at the gateway is used for packet forwarding toward the gateway. A node forwards to its parent linear combinations of its own packets, packets from its children and overheard packets. The tree is built having packet transmission minimization as goal (shortest path tree). SenseCode is simple and decentralized, having no need to maintain multiple routing structures for packets to flow through multiple paths.

The research work in [2] is important because, to the best of our knowledge, it is the first one presenting the design and implementation of a collection protocol that is suitable for sensor networks using network coding. However, it is assumed that all nodes participate as encoding nodes, which is not a very cost-effective solution. As stated in [4], network coding operations increase the

* Corresponding author.
E-mail addresses: esalhawri@ualg.pt (E. AL-Hawri), ncorreia@ualg.pt (N. Correia), abarra@ualg.pt (A. Barradas).

https://doi.org/10.1016/j.adhoc.2019.101870
1570-8705/

delay and node complexity and, typically, the best performance is not achieved when all nodes perform network coding.

In this article an efficient design of network coding based reliable sensor networks is addressed. More specifically, given a network scenario with certain critical communication channels/links (failure scenarios), the adequate amount and placement of network coding nodes is planned while ensuring that all data arrives to a gateway, even in case of packet loss at critical links. Each failure scenario includes one or more bad quality links that may go down simultaneously. As far as known, the reliability plus network coding node placement problem in many-to-one sensor networks has not been addressed. Previous work on node placement (e.g., [4]) focus on one-to-many scenarios and the goal is to increase network throughput, which is not suitable for sensor networks where reliability is a concern. Also, intermediate nodes just act as helper nodes assisting in content delivery. Here, on the contrary, a many-to-one scenario is assumed and the number of encoding nodes is reduced while ensuring reliability. Any node can produce data to be delivered. This approach is suitable for sensor networks where energy efficiency and reliability are more critical.

The proposed mathematical formulation is generic, allowing for the use of one or multiple gateways while considering any possible failure scenarios. The case of multiple gateways is relevant when sensor networks are federated, requiring a distributed storage system (e.g., P2P overlay) for the storage of data and coded packets. Such architecture is detailed in Section 3. In summary, the contributions of this work are the following:

- A mathematical model is developed to determine the optimal number of encoding nodes, and their location, under certain failure scenarios. Besides reducing cost and improving reliability, sensor networks end up having a higher performance in terms of delay because the overall number of network coding operations decreases;
- A heuristic algorithm is proposed for large-scale network scenarios;
- An architecture for the federation of network coding based WSNs is proposed. This ensures the recovery of lost packets when the other non-lost packets (coded or original packets) are forwarded towards different gateways.

Besides having failure scenarios into account, one should privilege places where encoding nodes would generate more innovative coded packets (nodes with high degree of connectivity). An innovative coded packet is a packet that is linearly independent from the previously received ones, considering a specific generation of packets. These packets bring, therefore, new information that is useful in the decoding process. The adopted approach ensures this issue.

The remainder of this article is organized as follows. Section 2 discusses related research work. Section 3 details the assumed architecture, which can incorporate scenarios of multiple gateways, along with the principles of network coding. In Section 4, the mathematical formulation for the design of network coding based reliable sensor networks is presented, along with a heuristic approach. Section 5 includes a performance analysis of the mathematical model and heuristic, and Section 6 concludes the article.

2. Related work

2.1. Relay node placement

Regarding node placement in WSNs, [5–8] are relevant works. In [5], the authors propose two methods for relay node placement that give k-connectivity to sensor nodes. One of the methods uses a genetic algorithm, while the other stands on a greedy approach. With a given number of potential positions, and number of targets (sensor nodes), both algorithms try to select the location for relay nodes so that the targets become k-connected. In [7], an algorithm is proposed for placing the minimum number of relay nodes, working as cluster heads in a two-tier WSN. Full coverage and connectivity of the WSN, and communication cost minimization, are goals to be achieved. The algorithm is based on spiral sequence which is created for stationary sensor nodes to minimize the total number of relay nodes. In [6], cluster heads are chosen so that the lifespan of the sensor network is extended. In [8], the authors introduce a constrained relay node placement problem. The goal is to minimize the number of relay nodes, which can be located in pre-determined candidate locations.

2.2. Network coding in wireless networks

Network coding was initially introduced in [9], and has caught the attention of many researchers. Most of these works try to introduce a good mechanism of coding, focusing on specific parameters like lifetime, delay, bandwidth, and/or energy to improve the whole network performance.

In [10], a network model is proposed where nodes are connected by disjoint routes. The packets are divided into generations, and every r packets are grouped into one of these generations. These are then coded using an encoding vector generated randomly over a finite field. At the destination, the coded packets are combined and represented as a system of linear equations. This approach is suitable for adhoc networks, where the network topology, packet loss, and node/link failures are difficult to predict, and no specific gateway/sink nodes is assumed. In Cope [11], the coding layer is inserted between MAC and IP layers. Cope uses three techniques to provide efficient network coding: opportunistic listening, opportunistic coding, and learning neighbor state. The authors in [12] propose a ARQ MAC protocol for control packets to coordinate a set of relay nodes, acting as helpers in bidirectional communication. These helper nodes apply network coding technique (XOR method) to enhance the network performance. In all these works any node can be the source or destination, meaning that the approaches are more adequate for general wireless networks.

The Adaptive Network Coding (AdapCode) in [13] uses network coding to provide reliable data transmission in WSNs, while keeping the amount of traffic minimized, in single source scenarios. That is, the other nodes just help in spreading the messages they receive. A set of r packets gives rise to one coded packet, by applying linear combination. At any node n, the coded packet ω is generated using $\omega = \sum_{i=0}^{r} \alpha_{n,i} k_i$. When the coded packet ω is ready, the node sends it with the coefficients (α_1, α_2,..., α_{r-1}) included in the packet's header. A node not able to decode the received packets will not resend them, in order to avoid collisions. Although focusing on WSNs, the assumption of having a single source is limitative.

In SenseCode, [2], network coding for many-to-one communication is addressed. That is, there are multiple sources sending data packets towards a gateway/sink using tree-based routing. Each node in SenseCode may deal with three kinds of messages: messages received from its children, messages overheard from its neighbors, and its own messages. To perform network coding, the node generates linear combinations of these messages. Applying this technique ensures that the sink will be able to recover packets that have been dropped. The approach in [14] is also many-to-one but simple XOR operations are applied, avoiding complex encoding techniques. Tree-based routing is not assumed, and any node can perform XOR operations with overheard packets. The SR-Code in [15] also assumes XOR operations, but a 2-tier network is adopted. In [16,17] a clustering approach is proposed that takes into account channel conditions and inter-node distance to decide adequate coding, decoding or control usage. The attractiveness of the

framework resides in the capacity of cluster heads to avoid the decoding process if it would not be necessary, and this decision is based on the channel state and distance. The mentioned works do not try to determine the adequate number of encoding nodes, and their placement, while ensuring energy efficiency and reliability, which is addressed here in this article. Our model is also adequate for multi-gateway scenarios, which is not considered by these authors.

2.3. Federation of WSNs

In federated WSNs, sensor nodes in different geographically distributed networks can use each other services. However, one main challenge related to this federation is how to announce resources for clients to be able to discover them. One possibility is to use REsource LOcation And Discovery (RELOAD) protocol that allows proxy nodes of constrained sensor networks to form a peer-to-peer (P2P) network where resources would be announced. For this reason, a Constraint Application Protocol (CoAP) usage for RELOAD has been recently proposed in [18]. Federated WSNs have been discussed in [19–22]. In [21], a machine-to-machine communication architecture is proposed for the federation of distributed WSNs. In this architecture, heterogeneous WSNs can be connected to each other, and to the Internet, using RELOAD and CoAP protocols. The work in [19] introduces P2P overlay sensor networks without any need for proxies or infrastructure support. Each group of nodes is organized as a ring, each containing at least one master node. Master nodes are sensor nodes with more powerful resources, which makes them work as a master for limited nodes.

The integration of Kad network (P2P network implementing the Kademlia P2P overlay protocol, [23]), with CoAP protocol is proposed in [22]. In this method, called CoHaRT, the CoAP protocol is located at the top of HaRTKad protocol stack. In [20], a method for the management of resource-limited device groups is proposed.

2.4. Placement of network coding nodes

Regarding the placement of network coding nodes, addressed in this article, [4] and [24] are two relevant works. Authors in [4] take care of such problem in the context of streaming overlays. The article discusses how to decrease the delay of packet delivery while selecting just a few number of nodes to be encoding nodes. Initially, the expected number of duplicate packets is calculated for each node in the network, helping to select the nodes performing the encoding process. Two algorithms are proposed for two different cases: *i*) When a central node knows the whole network information; *ii*) when nodes have just a local information knowledge. The results for both algorithms show maximum profit while minimizing the number of encoding nodes. The approach in [24], called Centrality based Network Coding Node Selection (CNCNS), minimizes the number of encoding nodes, leading to network throughput increase. Nodes at the center of dense neighborhoods are chosen to become encoding nodes, as network throughput is more likely to improve because such nodes can generate more innovative packets.

The problem addressed in this article is different from the previous ones due to the following:

- Network coding is applied for reliability, similarly to SenseCode (in [2]), but here the number of nodes performing encoding is minimized, avoiding unnecessary overhead to some nodes. The optimal location of encoding nodes is also determined.
- The adopted approach takes into account many independent failure scenarios, each leading to one or more inoperable links, which is more realistic than considering that all links can equally fail.
- The possibility for WSNs to be federated is considered. This has the following advantages: *i*) sensor nodes can perform encoding while leaving the decoding processing burden to the gateway or interested users, saving energy; *ii*) a distributed storage system to store original and coded data, together with encoding vectors, allows coded packets from encoding nodes at border zones (with neighbours sending their packets to other gateways) to be useful for the decoding process, improving efficiency.

As far as known, federated network coding based reliable sensor networks has not been addressed by previous authors.

3. Architecture

3.1. Network coding

Instead of simply relaying the received data packets, an encoding node generates new packets by combining received packets with its own packets [25]. This approach can be used to improve the throughput, efficiency, scalability, resilience to attacks and eavesdropping in networks.

With random linear encoding, r packets are combined in the form $\sum_{i=1}^{r} \alpha_i k_i$, where α_i is a coefficient generated over finite field $\mathbb{F}_{2^s}$, of size s, and k_i is a specific packet. Fig. 1 shows an example of this kind of encoding. When a source node S needs to send its packets, it sends two coded packets that are a linear combination of k_1 and k_2, using coefficients α_i. The relay nodes R_1 and R_2 receive these coded packets and forward them. When encoding node E receives the coded packets from R_1 and R_2, it encodes them and sends the resulting recoded packet to relay node R_3, which forwards it to gateways G_1 and G_2. Both gateways are able to decode these packets, retrieving the original ones, as long as they receive enough independent coded packets. Regarding the decoding process, two possibilities exist:

- Packets carry network encoding coefficients, allowing encoding nodes to decode packets first and then make further linear combinations. That is, the encoding vector must be appended to the packet. This may constitute a significant fraction of the payload if the original packet is short, which is typical in WSNs [2].
- Network encoding coefficients are known at the gateways or stored at a P2P distributed storage system, as discussed in

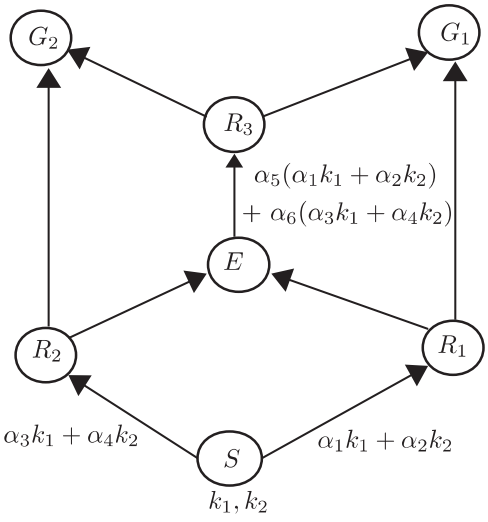


Fig. 1. Random linear coding. S is the source of packets, E is an encoding node, R_i are relay nodes, and G_i are the gateways.

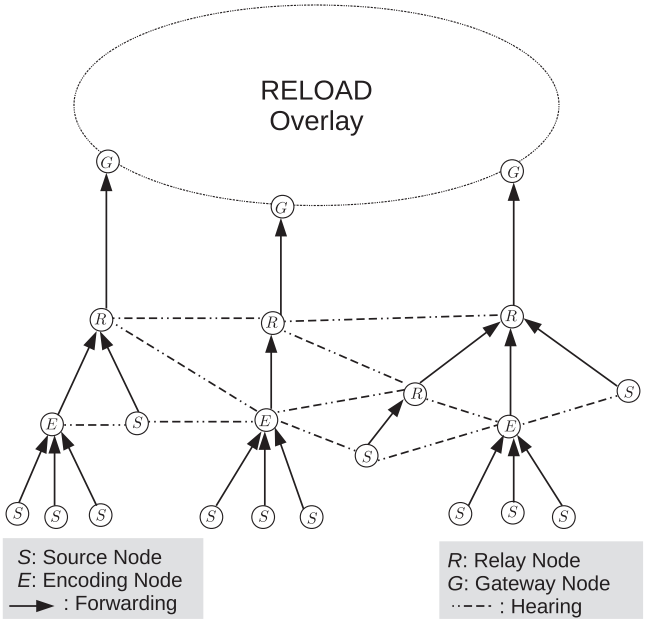


Fig. 2. RELOAD/CoAP based architecture.

the following section and assumed in this article. Recoding can be performed by inner network encoding nodes, without a previous decoding, because recursive decoding can always be performed at the end. Although this might be suitable for more static configurations, a node-to-gateway registration protocol (for encoding vector set up) may allow its implementation in more dynamic environments. Scalability is not an issue if a P2P distributed storage system is used because: *i*) In P2P overlays, the load per peer/gateway is independent from the total number of participating peers; *ii*) in WSNs, the number of nodes being served by a gateway is usually kept low in order not to increase the number of hops (towards the gateway) and congestion near the gateway.

3.2. RELOAD overlay

For WSNs to be federated, a RELOAD/CoAP based architecture can be used. Fig. 2 shows such architecture allowing multiple gateways to share a distributed storage system. CoAP is a Web application transfer protocol that was developed for constrained systems to provide RESTful services [26]. Although devices are heterogeneous regarding their radio layer, CoAP is expected to become a common application layer protocol. Recently, a CoAP Usage for RELOAD, a base protocol that provides a generic self-organizing P2P overlay network service, was proposed in [18]. The use of pluggable application layers, called Usages, allows RELOAD to fit any purpose [27]. Another example of RELOAD usage is the one defined for the Session Initiation Protocol (SIP) in [28]. In a RELOAD/CoAP architecture the gateway nodes form a distributed P2P overlay network to announce resources at its WSN, allowing clients to discover them. More specifically, the overlay can be used: as a lookup service, to store existing resources, and as cache for data.

In network coding based sensor networks, encoding nodes may end up listening to packets that will not be forwarded to the same gateway, if multiple gateways exist. In this case the recovery of lost packets may only be possible if the gateways share a storage system. For this reason the use of a RELOAD/CoAP based architecture is proposed. In such architecture, nodes can have different roles:

- *Source*: Senses the field under observation and sends its data packets to either encoding or relay nodes.
- *Encoder*: Receives data packets from sources and relay neighbors, encodes the received packets with its own packets, and sends the coded packets towards the gateway(s). Nodes of this kind can also send their original data packets, besides coded packets.
- *Relay*: Forwards the received packets (either coded or original) towards the gateway(s).
- *Gateway*: Receives the packets (either coded or original) from nodes at the wireless section, and stores them in the P2P overlay. Such data can then be fetched by interested users.

4. Encoding node location problem

4.1. Problem definition

Network coding induces delay and computational overhead, which will increase with the number of encoding nodes. For this reason the minimum number of encoding nodes, and their location, should be determined. This problem is defined as follows:

Definition 1 (Encoding Node Location (ENL) Problem)**.** Given a constrained sensor network graph $G(\mathcal{N}, \mathcal{L})$, where $\mathcal{N}$ are the nodes and $\mathcal{L}$ are the wireless communication channels (links), and given a set of independent failure scenarios $\mathcal{F}$, where failure scenario $f \in \mathcal{F}$ implies that one or more links are down, determine which nodes of $\mathcal{N}$ should perform network coding operations, together with packet flowing towards a gateway, so that any lost packet can be recovered through decoding operations.

In other words, failure scenarios reflect critical communication channels. Taking these into account will significantly reduce the end-to-end packet error rate (fraction of messages generated by the sources that are not successfully communicated to the destination).

4.2. Mathematical formulation

Having the previous definition in mind, subset $\mathcal{G} \subset \mathcal{N}$ is used to denote the gateway/sink nodes, meaning that $\mathcal{N} \backslash \mathcal{G}$ ends up including just inner network nodes. A link $l \in \mathcal{L}$ between nodes n_i and n_j basically means that n_i and n_j are within the range (coverage area) of each other, i.e., they can hear/reach each other.

The set of failure scenarios is denoted by $\mathcal{F}$, where failure scenario $f \in \mathcal{F}$ includes one or more links that may be inoperable at the same time. Failure scenarios are independent. That is, they do not occur simultaneously. The links included in a failure scenario f are denoted by $\mathcal{L}^f$.

As devices can have different capabilities (multiple classes of devices are envisaged by [29]) only a subset of $\mathcal{N}$ is expected to be able to act as encoding node (high processing capability required). Such set of encoding capable nodes is denoted by $\mathcal{N}^e$, $\mathcal{N}^e \subseteq \mathcal{N}$. An encoding node can be seen as providing reliability to the nodes it can hear.

Path trees are assumed for data to flow, each tree being rooted at a gateway, as in Fig. 2. The number of hops from data sources towards gateways is limited to H. The variables of the problem are the following:

ϑ^{n_i} One if node $n_i \in \mathcal{N}^e$ is chosen to act as encoding node; zero otherwise.

δ_l^s One if source node $s \in \mathcal{N}\backslash\mathcal{G}$ uses link $l \in \mathcal{L}$ to send its data towards a gateway; zero otherwise.

γ_l One if link $l \in \mathcal{L}$ is used by a path tree; zero otherwise.

$\sigma_{l,f}^{l'}$ One if link $l' \in \mathcal{L}$ is used for protection (data flowing in it will be linearly combined with other data at encoding nodes) of link $l \in \mathcal{L}^f$; zero otherwise.

β_s^g One if $g \in \mathcal{G}$ is the gateway for source $s \in \mathcal{N}\backslash\mathcal{G}$; zero otherwise.

$\eta_{l,f}^n$ One if node $n \in \mathcal{N}$ is the last/destination node of the protection path for link $l \in \mathcal{L}^f$; zero otherwise.

Regarding protection paths, their first node will be the source of the link they are protecting (failing link), their last/destination node will be a node with an operating uplink (according to the failure scenario under consideration), and both intermediate and final nodes must be encoding nodes in order to ensure that data reaches the operating uplink. Note that protection paths are not used for rerouting, but for overhearing.

4.2.1. Objective function

$$\text{Minimize} \sum_{\{n_i \in \mathcal{N}^e\}} \vartheta^{n_i}. \tag{1}$$

This objective function minimizes the total number of nodes performing encoding. Such goal naturally leads to the selection of nodes having high reachability/overhearing degree, while considering their usefulness as reliability providers (contributing with encoding at any protection path step).

4.2.2. Trees for data flow

For data to flow between constrained nodes and gateways, using tree-based routing, the following constraints must be fulfilled:

$$\sum_{\{l \in \mathcal{L}: src(l)=n\}} \delta_l^s - \sum_{\{l \in \mathcal{L}: dst(l)=n\}} \delta_l^s = \begin{cases} 1, & if\ n = s \\ -\beta_s^n, & if\ n \in \mathcal{G} \\ 0, & otherwise \end{cases}, \forall s \in \mathcal{N}\backslash\mathcal{G}, \forall n \in \mathcal{N}. \tag{2}$$

$$\sum_{g \in \mathcal{G}} \beta_s^g = 1, \forall s \in \mathcal{N}\backslash\mathcal{G}. \tag{3}$$

where $s \in \mathcal{N}\backslash\mathcal{G}$ is a data source, $src(l)$ is the source node of link l and $dst(l)$ is its destination node. Since β_s^n denotes if n is serving as gateway for data source s, or not, these two sets of constraints ensure, for a specific s, the so-called flow conservation law towards a single gateway.

$$\gamma_l \geq \delta_l^s, \forall l \in \mathcal{L}, \forall s \in \mathcal{N}\backslash\mathcal{G}. \tag{4}$$

$$\sum_{\{s \in \mathcal{N}\backslash\mathcal{G}\}} \sum_{\{l' \in \mathcal{L}\backslash\{l\}: src(l')=src(l)\}} \delta_{l'}^s \leq (1 - \gamma_l) \times \Delta, \forall l \in \mathcal{L}. \tag{5}$$

These constraints ensure that trees are built for routing. More specifically, if a link is performing data forwarding, which is determined by constraints (4), then constraints (5) ensure that no other link with the same source can be used (node has a single parent node). The Δ represents a big value and can be set to $\Delta = |\mathcal{N}| \times |\mathcal{L}|$.

$$\sum_{\{l \in \mathcal{L}\}} \delta_l^s \leq H, \forall s \in \mathcal{N}\backslash\mathcal{G}. \tag{6}$$

These constraints limit the number of hops for any flow (tree depth is limited).

4.2.3. Applying network coding upon failure scenario

$$\sum_{\{l' \in \mathcal{L}\backslash\mathcal{L}^f: src(l')=n\}} \sigma_{l,f}^{l'} - \sum_{\{l' \in \mathcal{L}\backslash\mathcal{L}^f: dst(l')=n\}} \sigma_{l,f}^{l'} = \begin{cases} -1 + \sum_{\{l' \in \mathcal{L}\backslash\mathcal{L}^f: src(l')=n\}} \gamma_{l'}, & if\ n = src(l) \\ -\eta_{l,f}^n, & otherwise \end{cases}, \forall f \in \mathcal{F}, \forall l \in \mathcal{L}^f, \forall n \in \mathcal{N}. \tag{7}$$

For a specific failure scenario, these constraints force data flow through overhearing nodes forming the protection path. More specifically, for each $l \in \mathcal{L}^f$ an alternative path (including overhearing/encoding nodes) must be ensured from the source of l until a node with an active link tree is reached. Note that neighbor nodes may have their uplinks down, if they fail too (simultaneous failure of more than one link is possible). This means that overhearing through one or multiple nodes may be required. An alternative path is built only if there is no active uplink for the source of the failing link, determined by $-1 + \sum_{\{l' \in \mathcal{L}\backslash\mathcal{L}^f: src(l')=n\}} \gamma_{l'}$. The $\eta_{l,f}^n$ variables are determined by the following set of constraints.

$$\sum_{\{n \in \mathcal{N}\}} \eta_{l,f}^n = 1 - \sum_{\{l' \in \mathcal{L}\backslash\mathcal{L}^f: src(l')=src(l)\}} \gamma_{l'}, \quad \forall f \in \mathcal{F}, \forall l \in \mathcal{L}^f \tag{8}$$

That is, these constraints are necessary to determine if a destination node, for the protection path of link $l \in \mathcal{L}^f$, will exist. In this case $\sum_{\{n \in \mathcal{N}\}} \eta_{l,f}^n$ will be one, meaning that a node n must be the final destination for the path protection being determined.

$$\eta_{l,f}^n \leq \sum_{\{l' \in \mathcal{L}\backslash\mathcal{L}^f: src(l')=n\}} \gamma_{l'}, \forall f \in \mathcal{F}, \forall l \in \mathcal{L}^f, \quad \forall n \in \mathcal{N}\backslash\mathcal{G} \tag{9}$$

$$\eta_{l,f}^n \leq 1 - \gamma_{l'}, \forall f \in \mathcal{F}, \forall l \in \mathcal{L}^f, \forall n \in \mathcal{N}, \quad l' : src(l') = n \wedge dst(l') = src(l) \tag{10}$$

Constraints (9) and (10) determine if n can be the final destination node for the protection path of link $l \in \mathcal{L}^f$, $\eta_{l,f}^n = 1$, or if it can only serve as an intermediate node, $\eta_{l,f}^n = 0$.

$$\vartheta^{n_i} \geq \sigma_{l,f}^{l'}, \forall n_i \in \mathcal{N}^e, \forall f \in \mathcal{F}, \forall l \in \mathcal{L}^f, \quad \forall l' \in l\backslash\mathcal{L}^f : d(l') = n_i. \tag{11}$$

These constraints set the ϑ^{n_i} variables, which indicate whether a node is acting as encoding node or not, according to the overhearing needs.

4.2.4. Binary assignments

$$\vartheta^{n_i}, \delta_l^s, \gamma_l, \beta_s^n, \sigma_{l,f}^{l'}, \eta_{l,f}^n \in \{0, 1\}. \tag{12}$$

The trees being built will adapt to failure scenarios and chosen encoding nodes.

4.3. Hardness of the problem

Theorem 1. *The ENL problem is NP-hard.*

Proof. When considering a single failure scenario, constraints (7)–(10) come down to the minimum Steiner tree problem, which happens to be NP-hard [30]. The minimum Steiner tree problem can be defined as follows: given an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with non-negative edge costs, find the tree of minimum cost that contains a given subset $\mathcal{T} \subseteq \mathcal{V}$ as terminal nodes. The nodes $\mathcal{V}\backslash\mathcal{T}$ are called Steiner nodes and can be used to build the Steiner tree.

For the just mentioned ENL subproblem to fit the minimum Steiner tree problem, an extra virtual node v is inserted into the network topology. That is, $\mathcal{V} = \mathcal{N} \cup \{v\}$, where $\mathcal{N}$ are the WSN nodes. Extra links from v to each node in $\mathcal{N} \backslash \mathcal{S}$ are included, where $\mathcal{S} = \cup_{l \in \mathcal{L}^f} src(l)$ includes the sources of failed links for failure scenario $f \in \mathcal{F}$, and failed links in $\mathcal{L}^f$ are excluded. That is, $\mathcal{E} = \mathcal{L} \backslash \mathcal{L}^f \cup \{l = (v, n) : n \in \mathcal{N} \backslash \mathcal{S}\}$. Assuming $\mathcal{S}$ to be the terminal nodes, solving the Steiner tree problem on such modified graph will provide the smallest possible set of encoding nodes, $\mathcal{N}^e$. In case of multiple failures, such ENL subproblem can be seen as an $|F|$-dimensional minimum Steiner tree problem. Thus, the ENL problem is NP-hard as well. □

4.4. Upper bound and heuristic algorithm

As stated in [4], finding the most effective subset of network encoding nodes is an NP-hard problem. For this reason the following heuristic is also proposed.

4.4.1. Upper bound

Let us assume $\mathcal{T}^{\mathrm{U}}$ as a universe set of feasible path tree partitions of nodes. That is, $\mathcal{T}^{\mathrm{U}} = \{\mathcal{T}^1, \mathcal{T}^2, \ldots, \mathcal{T}^{|\mathcal{T}^{\mathrm{U}}|}\}$ and $\mathcal{T}^i = \{\mathcal{T}^i_{g_1}, \mathcal{T}^i_{g_2}, \ldots, \mathcal{T}^i_{g_{|\mathcal{G}|}}\}$, $\forall i \in \{1, \ldots, |\mathcal{T}^{\mathrm{U}}|\}$. That is, $\mathcal{T}^i$ includes a tree for each gateway/root and $\mathcal{T}^i_{g_i}$ is used to denote the tree having g_i as root, covering nodes $\mathcal{N}^i_{g_i}$ and including links $\mathcal{L}^i_{g_i}$. Each partition $\mathcal{T}^i$ covers, therefore, all nodes. The impact of a path tree partition of nodes is described by the following cost function $f : \mathcal{T}^{\mathrm{U}} \rightarrow \mathbb{N}$:

$$f(\mathcal{T}^i) = \sum_{g \in \mathcal{G}} \sum_{n \in \mathcal{N}^i_g} Hops(n, g) \tag{13}$$

where $Hops(n, g)$ is the number of hops from node n towards g. The best path tree partition of nodes is, therefore, given by:

$$\mathcal{T}^* = \arg \min_{\mathcal{T}^i \in \mathcal{T}^{\mathrm{U}}} \{f(\mathcal{T}^i)\} \tag{14}$$

and $\mathcal{L}^*$ will be its links. Let us assume now that the overall set of links that need to be protected is given by:

$$\mathcal{L}^{\mathcal{F}} = \bigcup_{f \in \mathcal{F}} \mathcal{L}^f \tag{15}$$

meaning that duplicate links are merged. If the set of alternative paths, where overhearing/encoding nodes would be applied, associated with failing link $l \in \mathcal{L}^{\mathcal{F}}$ is defined by $\mathcal{P}_l = \{p = (src(l), \ldots, n_i) : \exists (n_i, n_j) \in \mathcal{L}^* \backslash \mathcal{L}^{\mathcal{F}}\}$, then the following cost function $g : \mathcal{P}_{l_1} \times \mathcal{P}_{l_2} \times \ldots \times \mathcal{P}_{l_{|\mathcal{L}^{\mathcal{F}}|}} \rightarrow \mathbb{N}$ can be defined:

$$g(\mathbf{p} = [p_{l_1}, p_{l_2}, \ldots, p_{l_{|\mathcal{L}^{\mathcal{F}}|}}]) = \left| \bigcup_{i \in \{1, \ldots, |\mathcal{L}^{\mathcal{F}}|\}} p_{l_i} \right| \tag{16}$$

Thus, the optimal solution would be:

$$S^* = argmin_{\mathbf{p}} \{g(\mathbf{p})\}. \tag{17}$$

Since this is hard to find, an upper bound can be obtained by $S^{\mathrm{UB}} = g(\mathbf{p}^{\min})$, where $\mathbf{p}^{\min} = [p^{\min}_{l_1}, p^{\min}_{l_2}, \ldots, p^{\min}_{l_{|\mathcal{L}^{\mathcal{F}}|}}]$ is made of shortest paths only. That is, $p^{\min}_{l_i}$ is the path starting at $src(l_i)$ that takes less hops to reach a node connected to an uplink in $\mathcal{L}^* \backslash \mathcal{L}^{\mathcal{F}}$.

4.4.2. Algorithm

Based on the previous upper bound, a scalable heuristic algorithm is proposed to determine which nodes should be encoding nodes. This is shown in Algorithm 1. Assuming the best known implementation for the single source shortest path problem, having complexity $O(|\mathcal{L}| + |\mathcal{N}| \log \log |\mathcal{N}|)$ (see [31]), Algorithm 1 will have complexity $O(|\mathcal{G}| + |\mathcal{L}| + |\mathcal{N}| \log \log |\mathcal{N}|)$. More specifically, although Lines 17–22 include a call to the Dijkstra algorithm,

```
1
2  Input: 𝒩, 𝒢, ℒ, ℱ, ℒ^f, ∀f ∈ ℱ;
3  Output: 𝒯*, 𝒩^e;
4
5  /* add virtual node and connect it to gateways */;
6  𝒩^e = ∅;
7  𝒩 = 𝒩 ∪ {v^X};
8  for g ∈ 𝒢 do
9      ℒ ∪ (v^X, g);
10     ℒ ∪ (g, v^X);
11 end
12 /* find path trees rooted at gateways */;
13 (𝒯*, ℒ*)=Dijsktra(source = v^X, destinations = 𝒩\𝒢);
14 Compute ℒ^ℱ;
15 p^min = [];
16 /* find alternative paths */;
17 for l ∈ ℒ^ℱ do
18     destinations = {n_i ∈ 𝒩\𝒢 : ∃(n_i, n_j) ∈ ℒ*\ℒ^ℱ};
19     (𝒯^temp, ℒ^temp)=Dijsktra(source = src(l), destinations);
20     p_l=ExtractShortestPath(𝒯^temp, ℒ^temp);
21     p^min ← p_l;
22 end
23 /* determine encoding nodes */;
24 for l ∈ ⋃_{l∈ℒ^ℱ} p^min[p_l] do
25     𝒩^e ← dst(l);
26 end
27
```

Algorithm 1: Determining path trees and encoding nodes.

Table 1
Topology information.

# Nodes	# Links	Connectivity degree
20	50	Sparse
20	80	Dense
40	120	Sparse
40	150	Dense

meaning that such algorithm could run $|\mathcal{L}|$ times (at most), it is possible to make a single call to the Dijkstra algorithm if a virtual node, connected to every $src(l) \in \mathcal{L}^{\mathcal{F}}$, is added to the topology. This possibility is omitted for the sake of clarity of the algorithm. Therefore, Line 13 plus Lines 17–22 result into complexity $O(|\mathcal{L}| + |\mathcal{N}| \log \log |\mathcal{N}|)$ because coefficients are ignored in the Big-O notation. Lines 14 and 20 are assumed to bring no extra complexity because such information can be ensured during the execution of the Dijkstra algorithm, if adequate data structures are used. Lines 8–11 and Lines 24–26 result into $|\mathcal{G}|$ and $|\mathcal{L}|$ steps (at most), respectively. Therefore, the overall complexity of Algorithm 1 will be $O(|\mathcal{G}| + |\mathcal{L}| + |\mathcal{N}| \log \log |\mathcal{N}|)$, after ignoring coefficients. From the foregoing complexity, Algorithm 1 can be considered scalable since components exhibit a growth lower than quadratic [32].

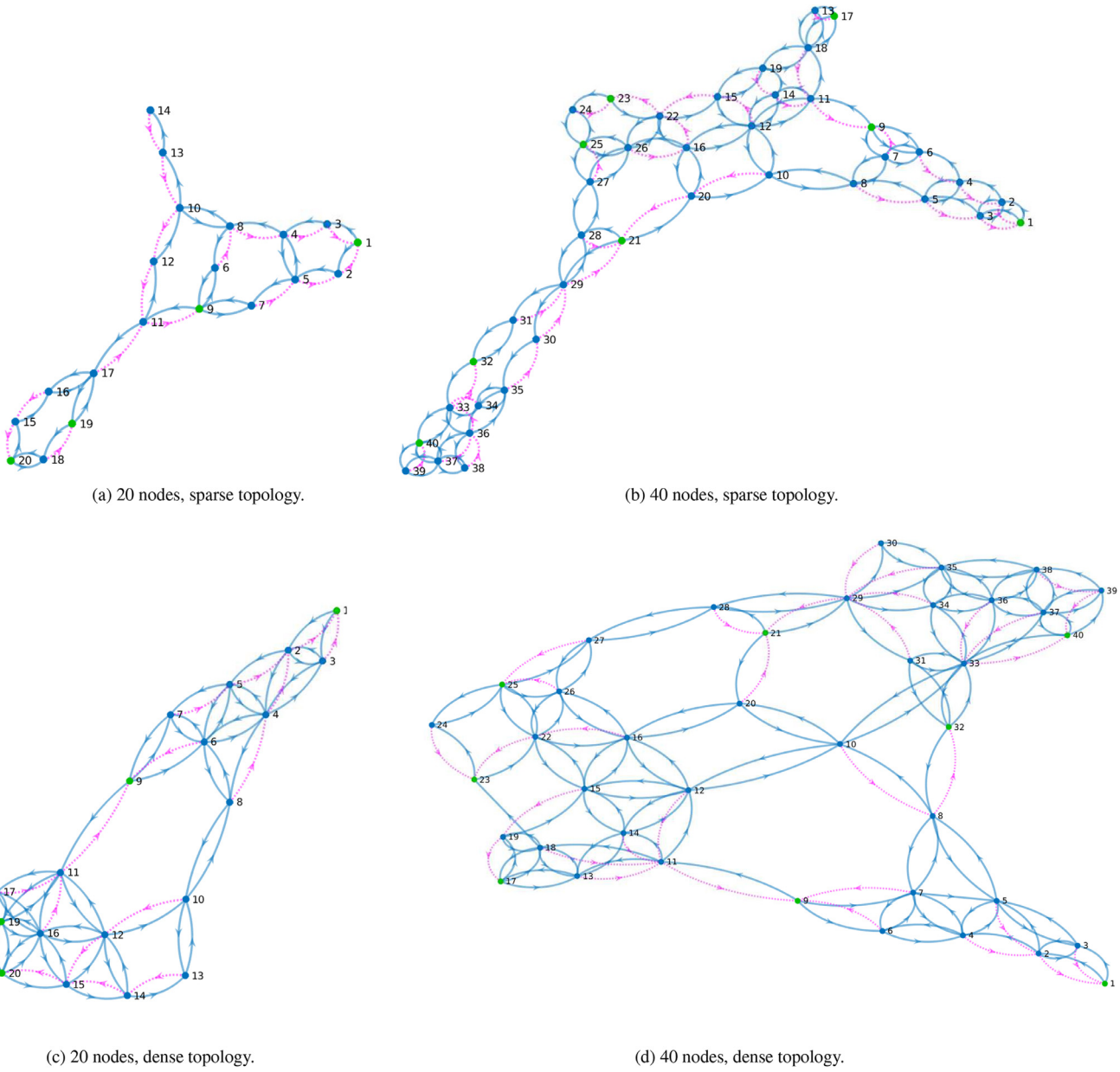
5. Performance evaluation

5.1. Scenario setup

Two different topology sizes were used for the analysis of results. These topologies were randomly generated using the algorithm in [33]. For each topology, two different connectivity degrees (dense and sparse) were considered. Such topology information is summarized in Table 1.

CPLEX[1] was used to find the results of the optimization model, while the heuristic and the random linear network coding simulation (for comparison of the amount of generated packets by our approach and [2]) were built in Matlab.[2]

[1] IBM ILOG CPLEX Optimizer version 12.8.
[2] MathWorks, Inc.



(a) 20 nodes, sparse topology.

(b) 40 nodes, sparse topology.

(c) 20 nodes, dense topology.

(d) 40 nodes, dense topology.

Fig. 3. Tested network topologies. For illustration, maroon links highlight a particular tree (from blue wireless links) while green nodes highlight a particular set of gateways.

5.2. Analysis of results

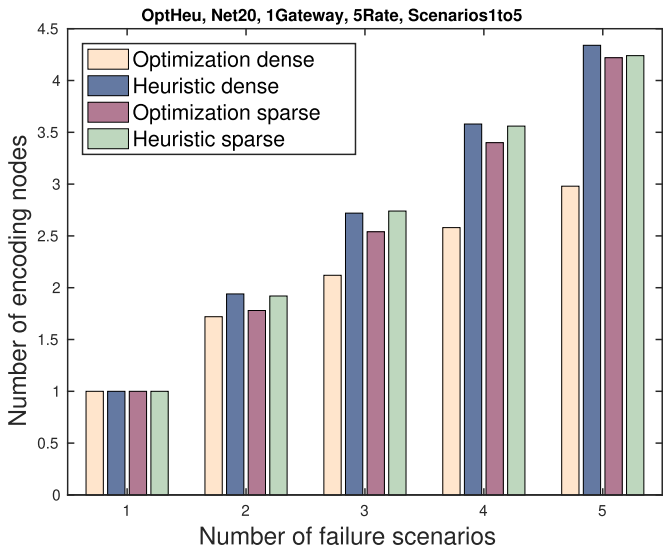
To analyze the performance of both the mathematical and heuristic approaches, different tests were performed for the just mentioned topologies, illustrated in Fig. 3. These figures include maroon links to highlight a particular tree for data flow (for illustration), while green nodes highlight a particular set of gateways. Different failure scenarios were randomly generated, based on specific link failure rates. Failure scenarios do not occur at the same time, but each failure scenario can lead to the failure of one or more links at the same time. More specifically, tests were done considering:

- Number of failure scenarios: 1–5.
- Link failure rate of each failure scenario: 5% or 10%.
- Network density: Dense or sparse.
- Number of gateways: 1 or 4 for small topologies, and 1 or 8 for big topologies.

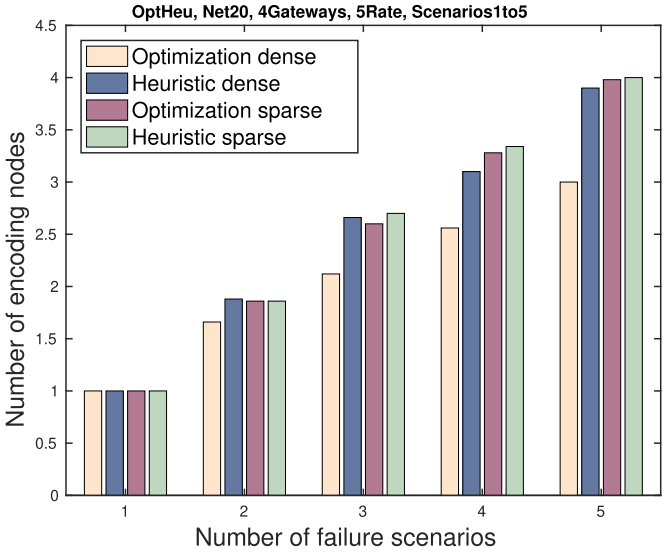
Table 2
Example offailure scenarios.

Nodes	Links	#fScens	fScens
20	50	4	[40][6][45][18]
20	80	3	[28][17][46]
20	50	5	[23][45 16][31][47 1][26 30]
20	80	2	[78 3][7 68]
40	120	2	[44 114][11 93]
40	150	3	[9 40][151 36][148 73]
40	120	2	[31 41 4 52][62 10 93]
40	150	1	[36 74 28 106]

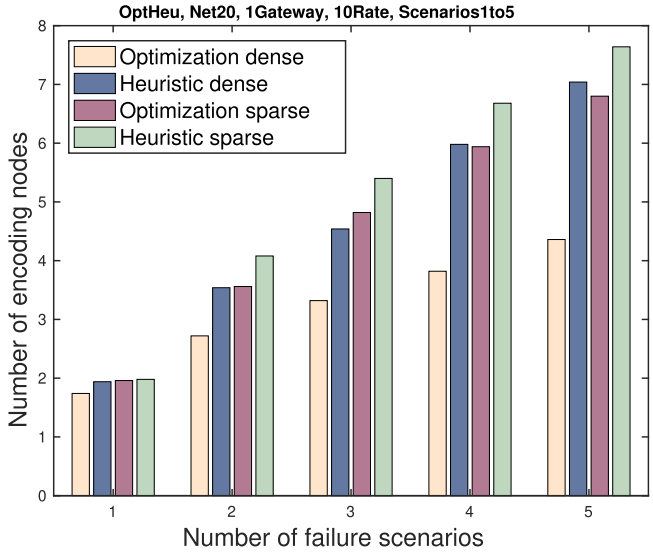
Table 2 provides examples of failure scenarios, where $\#fScens$ is the number of failure scenarios and *fScens* are the failure scenarios (each having one or more link numbers). Each test included 50 runs, and failing links change randomly at every run. The average of such 50 runs is used for the plotting of results.



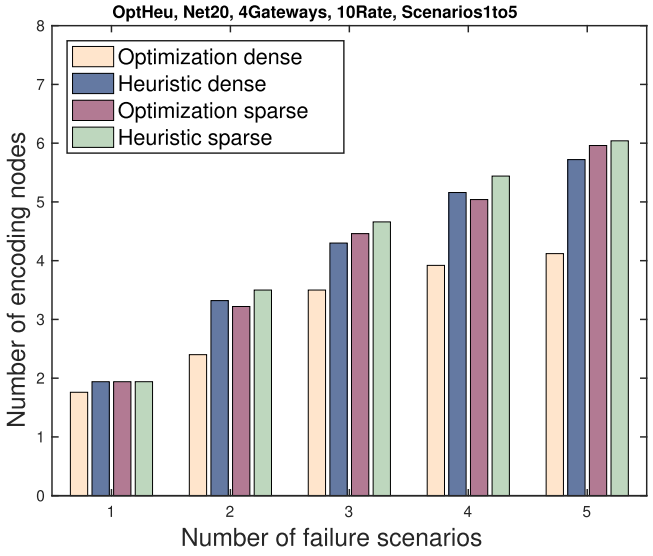
(a) 1 gateway, failure rate of 5%.



(b) 4 gateways, failure rate of 5%.



(c) 1 gateway, failure rate of 10%.



(d) 4 gateways, failure rate of 10%.

Fig. 4. Number of encoding nodes for the 20-node networks: sparse and dense topology tests.

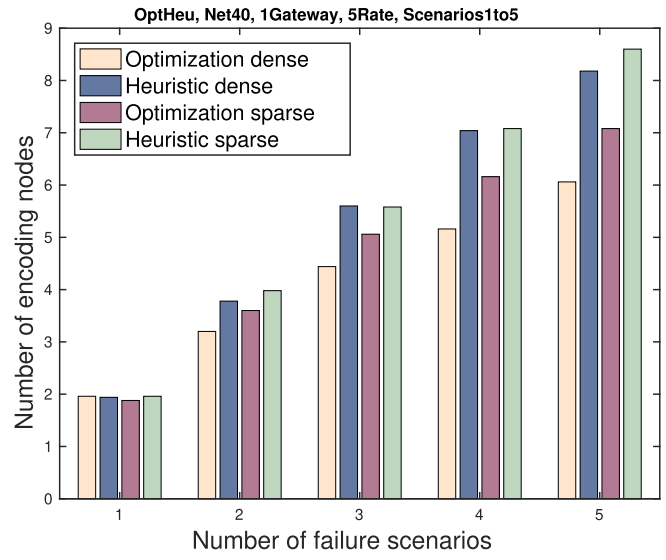
Note that the mathematical model is making a global optimization, which means that CPLEX will decide for trees that avoid failing links. This way the number of encoding nodes is minimized (objective function). On the contrary, the heuristic algorithm includes two steps: *i*) Building the trees; and *ii*) finding alternative paths (requiring encoding nodes) for failing links. Therefore, to fairly evaluate the performance of the heuristic, the mathematical model was also solved in two steps: *i*) Building the trees, using just constraints (2)–(6) and having as goal the minimization of the total number of hops; *ii*) finding alternative paths, using constraints (7)–(10) and setting the tree related variables according to the output from the first step. The link failure rate of each failure scenario, 5% or 10%, is then applied to tree links.

Regarding a possible comparison between the results obtained by the mathematical model, or heuristic, and SenseCode proposed in [2], it is important to highlight that the last assumes that all nodes are encoding nodes, and for this reason Figs. 4 and 5 do not include SenseCode. Section 5.2.3 discusses SenseCode regarding the amount of generated packets.

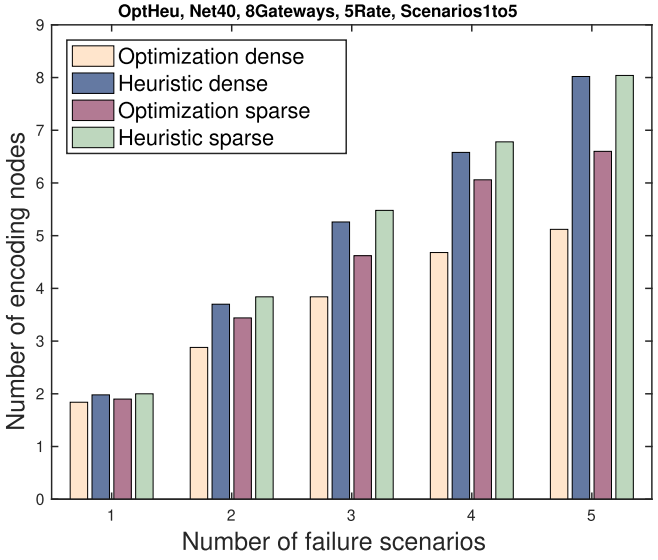
5.2.1. Number of encoding nodes: Small topology tests

Tests were performed for the sparse and dense 20-node topologies shown in Fig. 3. Plots 4(a) and 4(b) are results obtained for 5% of failure rate, 1 and 4 gateways, respectively, while plots 4(c) and 4(d) are results obtained for 10% of failure rate, 1 and 4 gateways, respectively. Results from the mathematical model (optimal) and the heuristic are both plotted for a changing number of failure scenarios.

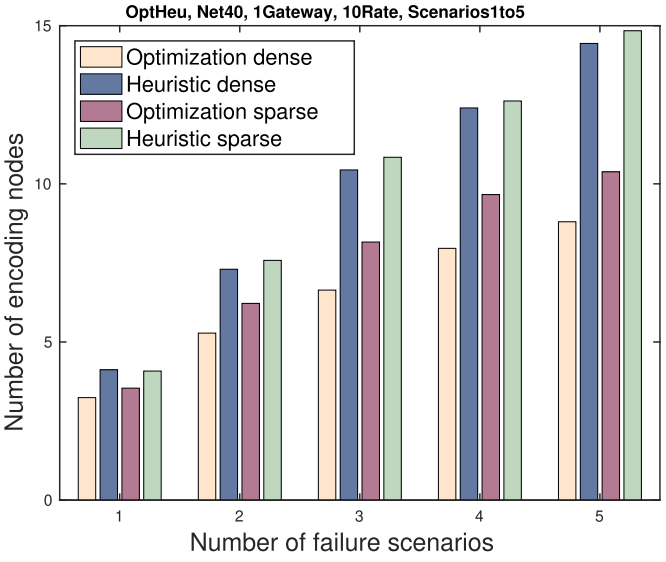
Regarding the sparse topology tests, it is possible to observe in plots 4(a) and 4(b) that there is a slight decrease in the number of required encoding nodes when the number of gateways change from 1 to 4. For plots 4(c) and 4(d) the benefit of using more gateways is much more significant, in particular for larger number of failure scenarios. In general, it is possible to state that the heuristic algorithm is able to get close to the optimal results obtained by the mathematical model, although this is less pronounced when there is a single gateway (less flexible scenario). Such behaviour is independent of the number of failure scenarios, which strengthens the scalability of the heuristic algorithm.



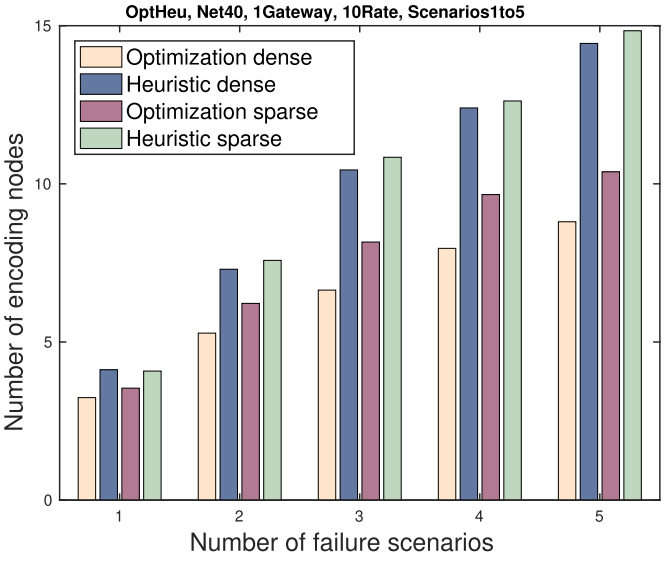
(a) 1 gateway, failure rate of 5%.



(b) 8 gateways, failure rate of 5%.



(c) 1 gateway, failure rate of 10%.



(d) 8 gateways, failure rate of 10%.

Fig. 5. Number of encoding nodes for the 40-node networks: sparse and dense topology tests.

Concerning dense topology tests, it is clear that the heuristic does not perform so well, although its performance improves when more gateways are used. Such behavior is similar for both failure rates of 5% and 10%. This means that multiple possible paths between nodes and gateways can make the algorithm diverge from the optimal. The number of encoding nodes is higher for a failure rate of 10%, similarly to the previous tests.

5.2.2. Number of encoding nodes: Big topology tests

Similarly to the previous section, tests were performed for the sparse and dense 40-node topologies shown in Fig. 3. Plots 5(a) and 5(b) are results obtained for 5% of failure rate, but now for 1 and 8 gateways, respectively, while plots 5(c) and 5(d) are results obtained for 10% of failure rate, also 1 and 8 gateways, respectively. Results from the mathematical model (optimal) and the heuristic are both plotted for a changing number of failure scenarios.

From the sparse 40-node topology tests, it is possible to observe an increase in the number of encoding nodes. This is an expected result because the number of failure scenarios under consideration is greater. The solutions obtained are slightly worse than the ones obtained for the small topology, regarding heuristic to optimal behaviour. That is, for small topologies the heuristic is able to get closer to the optimal values obtained by the mathematical formulation, meaning that the heuristic may have scalability issues regarding the number of nodes. As network size increases, however, the benefit of using more gateways becomes much clear, specially for higher failure rates. The gap between the heuristic and optimal also becomes smaller, meaning that increasing the number of gateways, while considering multiple failure scenarios, may compensate the just mentioned scalability issue.

Concerning dense 40-node topology tests, results show that these were the scenarios presenting greater heuristic to optimal gaps, confirming that multiple possible paths, between nodes and gateways, can make the algorithm diverge from the optimal, although performance improves when more gateways are used.

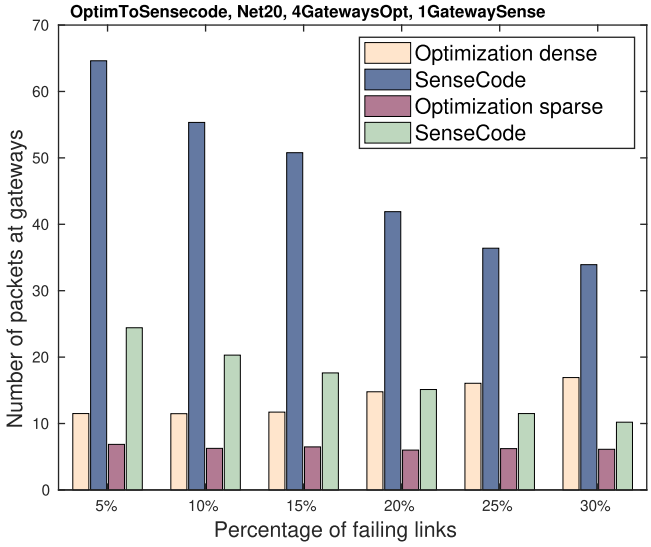


Fig. 6. Total number of packets at gateways (original and encoded) for 20-node networks.

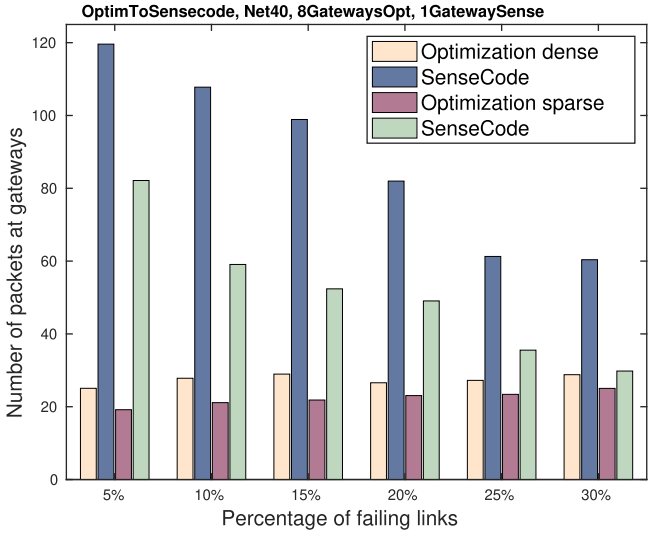


Fig. 7. Total number of packets at gateways (original and encoded) for 40-node networks.

5.2.3. Sensecode vs proposed approach

SenseCode assumes that all channels have a packet-erasure probability ϵ, but as long the sink receives N linearly independent combinations (N is the number of data sources), generated at intermediate relay nodes performing encoding, then recovery can be performed. The recovery depends on links failing to deliver packets and overhearing neighborhoods. Thus, in general, recovery ends up being more difficult when failing links are closer to the gateway. The proposed approach, on the contrary, considers specific critical links with very high packet-erasure probability (failure scenarios), while ensuring that a node in the neighborhood performs encoding for linear combinations to reach a gateway. The remaining links have zero packet-erasure probability. This way linear combinations are ensured to reach the gateway, allowing recovery to be performed. In other words, the proposed approach is suitable for network environments having predictable critical links, while SenseCode is suitable for network environments where packet loss location is not predictable.

It is possible to state that as long as critical/failing links are clearly identified (failure scenarios), the proposed approach has significant advantages because less packets are generated, due to less encoding nodes, allowing for longer network lifetimes and higher goodputs (rate of useful data) to be achieved. Figs. 6 and 7 show the total number of packets (original and encoded) at the gateways, in 20-node and 40-node networks, for both approaches. A single failure scenario, within which the percentage of failing links changes, is assumed for comparison with SenseCode to be possible. Also, similarly to [2], only the tree leaf nodes are data sources, assuming the trees generated by the optimization model discussed in Section 4.2. The percentage of failing links relates to tree links.

6. Conclusion

In this work, the design of network coding based reliable sensor networks is discussed. The aim is to determine a sufficient number of encoding nodes, and their location, taking into account failure scenarios. A mathematical model and a heuristic algorithm are developed to achieve such objective, considering either a single or multiple gateways. When multiple gateways are used, a shared distributed storage system becomes necessary because encoding nodes may listen to packets being forwarded to different gateways. Results show that both the mathematical model and heuristic algorithm can significantly reduce the number of required encoding nodes, when compared with a scenario where all network nodes are encoding nodes. Results also show that the heuristic is able to get closer to the optimal (obtained by the mathematical model) when small and sparse networks are considered. Large and dense networks make the heuristic diverge from the optimal, but this can be avoided if more gateways are considered. The adopted approach also generates a significantly smaller number of packets when compared to SenseCode proposed in [2].

Conflicts of interest

None.

Acknowledgments

This work was supported by FCT (Foundation for Science and Technology) from Portugal within CEOT (Center for Electronic, Optoelectronic and Telecommunications) and UID/MULTI /00631/2019 project. Eman AL-Hawri is supported by a grant from Thamar University - Yemen.

References

[1] N. Correia, G. Schutz, A. Mazayev, J. Martins, A. Barradas, An energy-Aware resource design model for constrained networks, Commun. Lett. 20 (8) (2016) 1631–1634.
[2] L. Keller, E. Atsan, K. Argyraki, C. Fragouli, Sensecode: network coding for reliable sensor networks, Trans. Sensor Netw. (TOSN) 9 (2) (2013) 25.
[3] W. Dargie, C. Poellabauer, Fundamentals of Wireless Sensor Networks: Theory and Practice, John Wiley & Sons, 2010.
[4] N. Cleju, N. Thomos, P. Frossard, Network Coding Node Placement for Delay Minimization in Streaming Overlays, in: International Conference on Communications (ICC), IEEE, 2010, pp. 1–5.
[5] S.K. Gupta, P. Kuila, P.K. Jana, Genetic Algorithm for k-Connected Relay Node Placement in Wireless Sensor Networks, in: International Conference on Computer and Communication Technologies (ICCCT), Springer, 2016, pp. 721–729.
[6] J.A. Martins, A. Mazayev, N. Correia, G. Schütz, A. Barradas, GACN: Self-Clustering genetic algorithm for constrained networks, Commun. Lett. 21 (3) (2017) 628–631.
[7] K. Nitesh, P.K. Jana, Relay Node Placement Algorithm in Wireless Sensor Network, in: International Advance Computing Conference (IACC), IEEE, 2014, pp. 220–225.
[8] S. Misra, S.D. Hong, G. Xue, J. Tang, Constrained relay node placement in wireless sensor networks: formulation and approximations, Trans. Netw. (TON) 18 (2) (2010) 434–447.
[9] R. Ahlswede, N. Cai, S.-Y. Li, R.W. Yeung, Network information flow, Trans. Inf. Theory 46 (4) (2000) 1204–1216.

[10] C. Gui, H. Chen, B. Sun, Y. Song, Energy efficient with network coding multipath routing algorithm in wireless sensor networks, Future Gener. Commun. Netw. 7 (6) (2014) 205–216.
[11] S. Katti, H. Rahul, W. Hu, D. Katabi, M. Médard, J. Crowcroft, XORs In the air: practical wireless network coding, Trans. Netw. (ToN) 16 (3) (2008) 497–510.
[12] A. Antonopoulos, C. Verikoukis, Network-Coding-Based cooperative ARQ medium access control protocol for wireless sensor networks, Distrib. Sensor Netw. 8 (1) (2011) 601321.
[13] I.-H. Hou, Y.-E. Tsai, T.F. Abdelzaher, I. Gupta, Adapcode: Adaptive Network Coding for Code Updates in Wireless Sensor Networks, in: International Conference on Computer Communications (INFOCOM), IEEE, 2008, pp. 1517–1525.
[14] O.T. Valle, C. Montez, G. Medeiros de Araujo, F. Vasques, R. Moraes, Netcoder: A Retransmission mechanism for WSNs based on cooperative relays and network coding, Sensors 16 (6) (2016) 799.
[15] Z. Merhi, O. Tahan, B. Abdul-Hay, R. Rammal, S. AbdulNabi, Smart relay network coding for data collection for wireless sensor networks, Eng. Res. Appl. 7 (1) (2017) 58–64.
[16] I. Ez-zazi, M. Arioua, A. El Oualkadi, P. Lorenz, On the performance of adaptive coding schemes for energy efficient and reliable clustered wireless sensor networks, Ad Hoc Netw. 64 (2017) 99–111.
[17] I. Ez-zazi, M. Arioua, A. El Oualkadi, On the design of coding framework for energy efficient and reliable multi-hop sensor networks, Procedia Comput. Sci. 109 (2017) 537–544.
[18] J. Jimenez, J. Lopez-Vega, J. Maenpaa, G. Camarillo, RFC 7650: A Constrained Application Protocol (CoAP) Usage for Resource Location and Discovery (RELOAD), Technical Report, IETF, 2015.
[19] M. Ali, K. Langendoen, A case for peer-to-peer network overlays in sensor networks, in: International Workshop on Wireless Sensor Network Architecture (WWSNA), 2007, pp. 56–61.
[20] I. Ishaq, J. Hoebeke, F. Van den Abeele, J. Rossey, I. Moerman, P. Demeester, Flexible unicast-Based group communication for coAP-Enabled devices, Sensors 14 (6) (2014) 9833–9877.
[21] J. Mäenpää, J.J. Bolonio, S. Loreto, Using RELOAD and coap for wide area sensor and actuator networking, Wireless Commun. Netw. (1) (2012) 121.
[22] J. Skodzik, P. Danielis, V. Altmann, B. Konieczek, E.B. Schweissguth, F. Golatowski, D. Timmermann, CoHaRT: Deterministic Transmission of Large Data Amounts Using CoAP and Kad, in: International Conference on Industrial Technology (ICIT), IEEE, 2015, pp. 1851–1856.
[23] J. Skodzik, P. Danielis, V. Altmann, D. Timmermann, Hartkad: a hard real-time Kademlia approach, in: Consumer Communications and Networking Conference (CCNC), IEEE, 2014, pp. 309–314.
[24] T.-h. Kim, H. Choi, H.-S. Park, Centrality-based network coding node selection mechanism for improving network throughput, in: International Conference on Advanced Communication Technology (ICACT), IEEE, 2014, pp. 864–867.
[25] C. Fragouli, J.-Y. Le Boudec, J. Widmer, Network coding: an instant primer, SIGCOMM Comput. Commun. Rev. 36 (1) (2006) 63–68.
[26] Z. Shelby, K. Hartke, C. Bormann, RFC 7252: The Constrained Application Protocol (CoAP), Technical Report, IETF, 2014.
[27] C. Jennings, E. B. Lowekamp, E. Rescorla, S. Baset, H. Schulzrinne, RFC 6940: Resource Location and Discovery (RELOAD) Base Protocol, Technical Report, IETF, 2014.
[28] C. Jennings, B. Lowekamp, E. Rescorla, S. Baset, H. Schulzrinne, E. T. Schmidt, RFC 7904: A SIP Usage for REsource LOcation And Discovery (RELOAD), Technical Report, IETF, 2016.
[29] C. Bormann, M. Ersue, A. Keranen, RFC 7228: Terminology for Constrained-Node Networks, Technical Report, IETF, 2014.
[30] R.M. Karp, Reducibility Among Combinatorial Problems, Springer, 1972.
[31] M. Thorup, Integer priority queues with decrease key in constant time and the single source shortest paths problem, Computer and System Sciences 69 (3) (2004) 330–353.
[32] Shang-Hua Teng, et al., Scalable algorithms for data and network analysis, Foundations and Trends in Theoretical Computer Science 12 (1–2) (2016) 1–274.
[33] F.A. Onat, I. Stojmenovic, Generating Random Graphs for Wireless Actuator Networks, in: International Symposium on a World of Wireless, Mobile and Multimedia Networks, IEEE, 2007, pp. 1–12.

Eman AL-Hawri received the four-year first degree in computer science from Thamar University, Thamar, Yemen. She received her master degree in Informatics Engineering from the University of Algarve, Faro, Portugal, in 2013. Currently Eman is working toward her PhD in Optimization and Computer Network, the University of Algarve, Faro, Portugal.

Noélia Correia received the B.Sc. and M.Sc. degrees in computer science from the University of Algarve, in Faro, Portugal, in 1995 and 1998, respectively, and the Ph.D. degree in optical networks (computer science) from the University of Algarve in 2005, and done in collaboration with University College London, U.K. She is a Lecturer with the Science and Technology Faculty, University of Algarve, in Faro, Portugal. Her research interests include the application of optimization techniques to several network design problems, in the optical, wireless, and sensor networks fields, and development of algorithms. She is a Founding Member of the Center for Electronics, Optoelectronics and Telecommunications, University of Algarve, a research center supported by the Portuguese Foundation for Science and Technology. She is also the Networks and Systems Group Coordinator.

Alvaro Barradas received the four-year first degree in computer science and the Ph.D degree in electronics engineering and computing from the University of Algarve, Faro, Portugal, in 1995 and 2009 respectively. His major field of study has been on load balancing for optical networks. Currently he is an Auxiliary Professor at the Faculty of Science and Technology from the University of Algarve. His previous professional experience includes two years working on system test for computer science and telecommunications consulting companies in Lisbon and Dusseldorf. His current research interest includes the study and simulation of hybrid networks where QoS provisioning is challenging. He is a member of the Center of Electronics Optoelectronics and Telecommunications (CEOT) at the University of Algarve, a research center supported by the Portuguese Foundation for Science and Technology.