Adaptive Dynamic Network Slicing in LoRa Networks

Samir Dawaliby, Abbas Bradai, Yannis Pousset

Adaptive Dynamic Network Slicing in LoRa Networks

Samir Dawaliby[1], Abbas Bradai, Yannis Pousset

XLIM Laboratory, 11 Boulevard Marie et Pierre Curie, Chasseneuil du Poitou, France

[a] *University of Poitiers*

Abstract

Knowing the heterogeneity of applications and services that need to to be supported in the internet of things (IoT), network slicing came out as a potential solution that virtually isolates 5G networks with various service requirements over a common physical network infrastructure. The latter needs to simultaneously support and isolate traffic issued from mobile and machine services which may require different needs in terms of reliability, latency, and bandwidth. In this paper, network slicing is investigated in LoRa networks over fixed and dynamic slicing strategies. The performance of LoRa slices is evaluated with different spreading factor (SF) configurations. Then, a dynamic inter-slicing algorithm is proposed based on a maximum likelihood estimation that avoids resource starvation and prioritizes a slice over another depending on its QoS requirements. Moreover, a novel intra-slicing strategy is proposed that maximizes resource allocation efficiency of LoRa slices with regard to their delay requirements. An energy module for LoRa in NS3 is also implemented to evaluate the energy consumption of devices in each slice. Simulation results performed in realistic LoRa scenarios highlight the utility of our proposition in improving QoS requirements of IoT devices and providing isolation between slices.

Keywords: Internet of Things (IoT), wireless networks, LoRa, network slicing, resource allocation, quality of service (QoS)

1. Introduction

With the development of the fifth generation (5G) wireless networks, it is expected that by 2020, an all-connected world of humans and machines will be reached offering with it the needed flexibility to manage networks with various service requirements using major arising technologies namely network functions virtualization (NFV) and software defined networking (SDN). With the development of the latter, network slicing is proposed as one of the most important technologies to reach this goal by using a collection of logical functions. Our objective is to provide isolation between multiple virtual networks with various QoS requirements to be created on top of a common physical device, being mutually instantiated on-demand and independently managed.

Low-power wide-area (LPWA) research efforts direct towards LoRa technology and is considered as one of the best emerging technologies for the internet of things (IoT). With network slicing, radio

Email addresses: samir.dawaliby@univ-poitiers.fr (Samir Dawaliby), abbas.bradai@univ-poitiers.fr (Abbas Bradai), yannis.pousset@univ-poitiers.fr (Yannis Pousset)

[1]Corresponding author

resources need to be virtually reserved in an isolated and efficient manner to provide specific service requirements for each slice. Three generic services are provisioned in 5G with conflicting quality of service (QoS) requirements (i.e., ultra-reliable low-latency communications (URLLC), enhanced mobile broadband (eMBB), and massive machine-type communications (mMTC)). Service requirements in mMTC category may vary between two applications running on a single IoT device and require heterogeneous behavior mainly when it comes for example to latency and reliability. The massive number of IoT devices continuously increasing and connecting alongside mobile devices to the new generation core network (NGCN) in 5G, brings an exceptional need for network slicing and virtualization to improve network flexibility. This leads to new challenges in designing resource allocation and slicing strategies which must guarantee slicing isolation and simultaneously provide the opportunity for infrastructure providers to easily meet the required QoS for IoT devices in a cost-effective manner.

1.1. Related Works

Performance evaluation over LoRa networks has been intensively reviewed by many research studies in the literature [1] [2] [3]. Other research studies focused on evaluating LoRa scalability [4] while considering co-SF interference that comes from collisions when using the same SF configuration on the same channel [5] whereas others assumed that SFs on a channel are perfectly orthogonal [6] [7]. SF represents the ratio between the chirp rate and the data symbol rate and affects directly the data rate and the range that a LoRa device can reach away from a LoRaWAN gateway. Moreover, co-SF directly impact communication reliability, reduces the packet delivery ratio (PDR) successfully decoded at the gateway [8] and limits the scalability of a LoRa network when increasing the number of devices [9]. Therefore, the latter should be considered in any upcoming study related to SF configuration strategies and network deployments. Some study examples focused on finding the optimal transmitter parameter settings that satisfy performance requirements using a developed link probing regime [10]. In [11], the authors analyze several SF configuration strategies where a group of LoRa devices can be configured with similar or heterogeneous SFs based on their position from the gateway. The goal is to find the scheme that gives the best performance in terms of PDR. However, the impact of the latter configuration on network slicing has not been previously tested.

Few research works recently tackled network slicing in IoT and focused on machine critical communications over various wireless networks. The work in [12] introduced a slicing infrastructure for 5G mobile networking and summarized research efforts to enable end-to-end network slicing between 5G use cases. Furthermore, authors in [13] and [14] adopted network slicing in LTE mobile wireless networks. The former proposed a dynamic resource reservation for machine-to-machine (M2M) communications whereas the latter present a slice optimizer component with a common objective in both papers to improve QoS in terms of delay and link reliability. In a 5G wearable network, the authors took advantage of slicing technology to enhance the network resource sharing and energy-efficient utilization [15]. Moreover in [16], the authors perform slicing in virtual wireless sensor networks to improve lease management of physical resources with multiple concurrent application providers. In [17], authors focused on URLLC and proposed several slicing methods for URLLC scenarios which require strong latency and reliability guarantees. Nowadays, guaranteeing service requirements in LoRa wireless access network (LoRaWAN) with traffic slicing remain as open research issues [18]. Therefore, unlike the previous work, in this article network slicing is investigated in LoRa technology which, to the best of our knowledge, has not been treated before by the research community.

1.2. Contributions and outlines

Our main contribution with respect to the surveyed literature are stated as follows:

1. Network slicing is investigated over different SF configurations in order to evaluate system performance and find the one that serves best LoRa devices in each slice.
2. A dynamic inter-slicing algorithm is proposed where the bandwidth will be similarly reserved on all LoRa gateways based on a maximum likelihood estimation (MLE) and then the latter is improved and extended with an adaptive dynamic method that considers each LoRa gateway separately and reserves its bandwidth after applying MLE on the devices in its range. Both dynamic slicing propositions will be compared to a straightforward fixed slicing strategy in which the GW's bandwidth is equally reserved between slices.
3. An energy model for LoRaWAN is integrated in NS3 based on LoRa energy specifications to analyze the energy consumed in each slice and an intra-slicing algorithm is proposed that meets the QoS requirements of each slice in an isolated manner.

The remainder of this paper is organized as follows. Section II presents an overview of LoRa and describes the system model and the network slicing problem established in this paper. In Section III, the slicing algorithm is proposed and implemented over the LoRa module of NS3 simulator [19]. The performance evaluation of the algorithm and simulation results are analyzed and carried out through various scenarios in Section V. Finally, Section VI concludes the paper.

2. Problem Description

2.1. LoRa Overview

LoRa is a shortcut name for **Lo**ng **Ra**nge and a spread spectrum modulation technique that derives from chirp spread-spectrum (CSS) modulation as described in the IEEE standard 802.15.4 [20]. CSS modulation transmits symbols by encoding them into multiple signals of increasing or decreasing radio frequencies making signals more robust to multi-path interference, Doppler shifts and fading [21]. Currently, LoRa physical layer is used with LoRaWAN MAC layer despite being capable of communicating with any other MAC layer. LoRaWAN supports low-power and long-range communications where IoT devices transmit directly to LoRa gateways in a star topology before forwarding data to a backbone infrastructure. Each device k adopts a specific SF configuration for information transmission. LoRa spreads each symbol in a rate of 2^{SF} chips per symbol with $SF = \{7, ..., 12\}$ resulting a data rate computed as written in **Eq. 1** below:

$$R_{k,l,m} = SF.\frac{R_c}{2^{SF}} = SF.\frac{b_{l,m}}{2^{SF}} \quad bits/s \tag{1}$$

where R_c denotes the chip rate and $R_{k,l,m}$ the data rate achieved by a device k depending on the bandwidth assigned to slice l of LoRa gateway m. Channel bandwidth varies from a region to another from 7.8 kHz to 500 kHz. Increasing the bandwidth improves the data rate of LoRa device on the

Spreading Factor	Sensitivity (dBm)
SF7	-130.0
SF8	-132.5
SF9	-135.0
SF10	-137.5
SF11	-140.0
SF12	-142.5

Table 1: List of parameters

expanse of sensitivity. In this paper, 125 kHz bandwidth is adopted for each channel following to the European frequency regulations.

Moreover, as shown in **Table 1**, increasing the spreading factor reduces the transmitted data rate, increases the strength of the signal and offers a better sensitivity at the gateway receiver following to the **Eq. 2** below:

$$P_{k,l,m}^{rx} = \frac{P_{k,l,m}^{tx} g_{k,l,m}^{rx} g_{k,l,m}^{tx}}{L} e^{\xi} \tag{2}$$

where $P_{k,l,m}^{rx}$ and $P_{k,l,m}^{tx}$ denotes the received and transmitted power with a channel antenna gain expressed with $g_{k,l,m}^{rx}$ and $g_{k,l,m}^{tx}$ respectively. L is the path loss which depends on the distance between the transmitter and the receiver and e^{ξ} is the lognormal shadowing component with $\xi \sim N(0, \sigma^2)$. Regarding interference, signal-to-interference-plus-noise ratio (SINR) varies based on the adopted SF on each device. The assumptions in [19] are followed where a packet should survive interference that comes from other LoRa transmissions. Each LoRa device experiences a SINR value computed based on the **Eq. 3** below:

$$SINR_{i,j} = \frac{P_i^{rx}}{\sigma^2 + \sum_{n \in \partial_j} P_n^{rx}} \tag{3}$$

where P_i^{rx} is the power of the packet n under consideration sent by device with $SF = i$ and ∂_j a set of interfering packets with a common $SF = j$. Each element in the below matrix [22] denotes the minimum signal power margin threshold $V_{i,j}$, with $i, j \in \{7, ..., 12\}$, that a packet sent with $SF = i$ must have in order to be decoded successfully over every interfering packet with $SF = j$. Hence, packet survives interference with all interfering packets if, considering all combinations of SF, a higher power margin value (dB) is satisfied than the corresponding co-channel rejection value.

$$\left\|\begin{array}{ccccccc} & SF_7 & SF_8 & SF_9 & SF_{10} & SF_{11} & SF_{12} \\ SF_7 & -6 & 16 & 18 & 19 & 19 & 20 \\ SF_8 & 24 & -6 & 20 & 22 & 22 & 22 \\ SF_9 & 27 & 27 & -6 & 23 & 25 & 25 \\ SF_{10} & 30 & 30 & 30 & -6 & 26 & 28 \\ SF_{11} & 33 & 33 & 33 & 33 & -6 & 29 \\ SF_{12} & 36 & 36 & 36 & 36 & 36 & -6 \end{array}\right\|$$

In this paper, log-distance propagation loss model is adopted to evaluate the performance of LoRa devices in a dense environment and is expressed following to the **Eq. 4** below:

$$L = L_0 + 10.n.log_{10}(\frac{d}{d_0}) \tag{4}$$

where L denotes the path Loss (dB), d the length of the path (m), n represents the path loss distance exponent, d_0 the reference distance (m) and L_0 the path loss at reference distance (dB).

2.2. System Model

Network slicing in a LoRa-like network is considered in this work, consisting of a set of $K = \{1, 2, ..., k\}$ LoRa devices and $M = \{1, 2, ..., m\}$ LoRa Gateways (GWs) plotted over a cell and connected to external LoRa Servers via fronthaul links. Compared to Sigfox [23], NB-IoT [24] and other IoT technologies, LoRa is more resilient to interference and jamming [25] thanks to its ability to efficiently trade communication range with high data-rate. Network slicing mainly brings flexibility to the network by virtually reserving physical resources in order to meet the QoS requirements of each slice. In IoT, each device requires specific QoS requirements in terms of delay and reliability depending on the running IoT application. A slicing framework is defined that consists of a set of L virtual network slices such that $L = \{1, 2, ..., l\}$ can be created on physical network hardware, more specifically on LoRa GWs, where the bandwidth of each GWs is divided into l slices with $l \in L$, as shown in **Fig. 1** below. The main goal behind slicing is to virtually split the network by reserving resources for each slice on the same physical device with each slice l characterized by a priority sp_l and a bandwidth $b_{l,m}$ at the GW level. A set of virtual flows F is defined where a device k associated to slice l generates a flow $f_{k,l,m}$ that goes from the GW m to LoRa servers and is characterized by a utility metric $U_{k,l,m}$ specified later on in this paper. LoRa GWs in range will receive the packets but only one GW slice forwards the packet to LoRa servers to avoid duplicated packets.

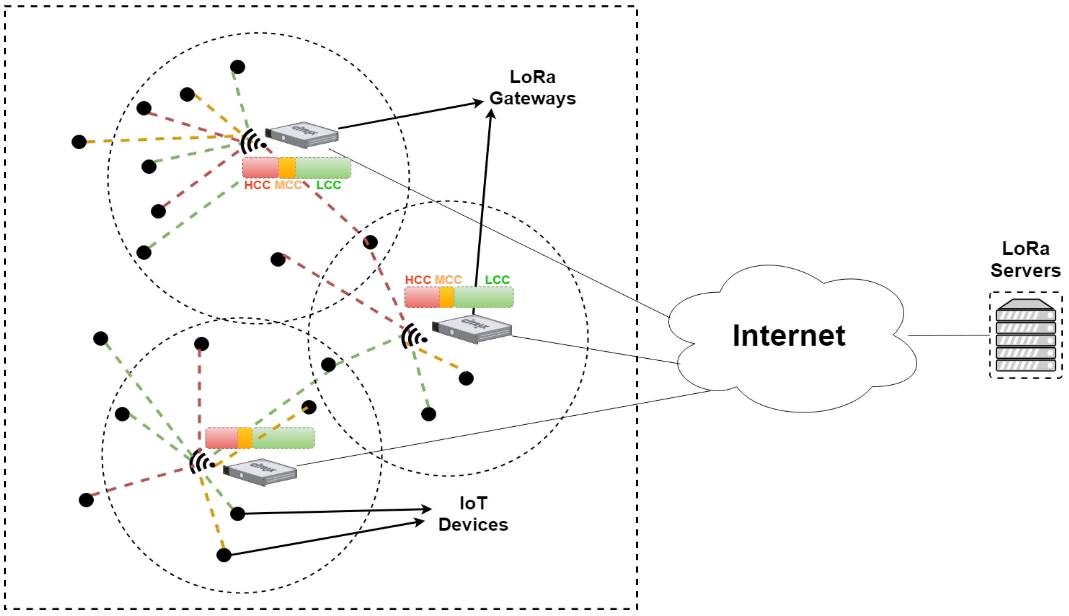


Figure 1: IoT Slicing architecture in LoRa Networks

2.3. Problem formulation

In this work, optimizing network slicing in IoT is a threefold problem and involves: *1) LoRa devices admission and association to slices*; *2) Finding the best inter-slicing resources reservation strategy*; *3) Intra-slice resources allocation.* First, L slices are defined based on the delay urgency factor and reliability requirements of each device. Each device is assigned next to the slice that meets best its service latency requirement. It is noteworthy that in IoT, the delay urgency and reliability represents the major key factors to define the priority of a device over another without neglecting the service type and the congestion that results from the large amount of IoT devices. Next, based on throughput requirements of each slice, slicing rate is estimated to define capacity c_l that needs to be reserved for each slice l. Each GW m reserves for each slice, some of its physical receiving paths and each member of a slice is characterized by a specific utility value $U_{k,l,m}$. Finally in the third step, intra-slice resource allocation is optimized by assigning each device in slice l to the most efficient virtual flow with the highest utility metric. Let $\alpha_{k,l} \in \{0,1\}$ be a binary variable that indicates whether a device k is associated with a flow $f_{k,l,m} \in F$. The goal is to maximize the number of LoRa devices assigned to virtual flows in a way that maximizes the utility function adopted by each slice members. Therefore, the slicing and resource allocation problem for IoT can be formulated as

$$Max \sum_{k\in K}\sum_{l\in L} \alpha_{k,l} U_{k,l,m}, \forall m \in M \tag{5}$$

subject to

$$C1 : \sum_{l\in L} \alpha_{k,l} = 1, \forall k \in K \tag{6a}$$

$$C2 : \sum_{k\epsilon K} \beta_{k,m} p_{k,l,m} \leq P_m^{max}, \forall m \in M, \forall l \in L \tag{6b}$$

$$C3 : \sum_{k\epsilon K} \alpha_{k,l}\beta_{k,m} r_{k,l,m} \leq R_{l,m}^{max}, \forall l \in L, \forall m \in M \tag{6c}$$

$$C4 : \beta_{k,m} = \begin{cases} 1 & \text{if device } k \text{ is assigned to gateway } m. \\ 0 & \text{Otherwise.} \end{cases} \tag{6d}$$

Knowing that multiple virtual network slices are isolated and built on top of a common physical gateway, (6a) ensures that each device should always choose exactly one and only network slice even if the latter was implemented on different physical gateways. Hence in a multi-gateway scenario, the device assigned to a slice will only have the option to choose between the flows that leads to the slice it belongs to. The total transmission power of each GW m is limited in constraint (6b). Moreover, constraint (6c) guarantees the sum of uplink traffic sent by slice members do not exceed the maximum data rate capacity of the slice that can be sent through each gateway. Constraint (6d) ensures binary-association values $\beta_{k,m}$ between a physical IoT device k and a physical LoRa gateway m. **Table 2** below summarizes the key denotations adopted in the paper.

Parameter	Parameter Name
M	the set of LoRa Gateways
K	the set of LoRa Devices
L	the set of Slices
K_l	the set of devices associated to slice l
∂_j	the set of packets with $SF = j$
$f_{k,l,m}$	virtual flow for device k in slice l through GW m
$U_{k,l,m}$	utility for device k in slice l on GW m
sp_l	slice priority of slice l
$b_{l,m}$	bandwidth assigned for slice l over GW m
P_m^{max}	maximum transmission power of GW m
$g_{k,l,m}$	power gain between a GW m and a device k
e^{ξ}	lognormal shadowing component
$\alpha_{k,l}$	admission index of device k to slice l
$\beta_{k,m}$	association index of device k to GW m
u_k	urgency factor for device k
d_k	instant packet delay for device k
PDB_k	packet delay budget for device k

Table 2: List of parameters

3. The Proposed Slicing Algorithm

In LoRa networks, the general control plane and resource management module are centralized and moved to a management and control entity (MCE) in the cloud to ensure an efficient coordination of resources. Hence, LoRa servers will be the final decision maker in assigning the devices to the appropriate slice and defining the gateway that will transmit the packet following to a three-steps optimization algorithm. In the first step, each device will be assigned to the slice that meets its QoS requirements based on a balanced iterative reducing and clustering method using hierarchies (BIRCH). Next, after assigning each device to its corresponding slice, GW resources will be dynamically reserved for each slice based on a maximum likelihood estimation (MLE) before finally forwarding the packet to LoRa servers through the GW that provides the maximum utility value.

3.1. BIRCH-based Slicing Definition

Due to the ultra-dense nature in an IoT, BIRCH algorithm is adopted [26] which belongs to the agglomerative hierarchical clustering family and was proven as the best available clustering method for handling large datasets [27] [28]. The main goal behind this method is to define slices by checking the QoS requirements of each MTCD and moving from a large set of devices into a group of subsets with similar QoS requirements. The most urgent devices are the ones that have the closest instant delay d_k to their packet delay budget PDB_k and are assigned the highest priority. u_k denotes the urgency factor of device k with $u_k = d_k/PDB_k$. Given K_l devices in a cluster l, the latter will be considered as a utility point u_k of each device in a cluster with $k = 1, 2, ..., K_l$. Each node in the CF-tree is a cluster of subclusters defined by a clustering feature CF as follows:

$$CF = (K_l, LS, SS) = (K_l, \sum_{k=1}^{K_l} u_k, \sum_{k=1}^{K_l} u_k^2) \tag{7}$$

where K_l denotes the number of devices in the cluster, LS the linear sum of the K_l utility points and SS the square sum of the K_l utility points. BIRCH dynamically builds a CF-tree, at each time a new MTCD is inserted based on two parameters: a branching factor B and a threshold T. Each parent node contains a maximum number of B childs and a single child node contains at most T entries. In this problem, B represents the number of L slices created with K_l the group of devices admitted to slice l. Hence, l nodes derive from the root representing the slices created with each slice is made up of a group of subclusters. Therefore, entries in CF-tree are not considered as devices but as a set of subclusters C that belongs to slice l and groups LoRa devices with nearly similar utility points.

Pseudo-code 1 BIRCH-based Slicing Admission algorithm

```
   Input  : Set of devices $K$, diameter $D$, branching factor $L$, threshold $T$
1  begin
2      Initialize as many clusters as devices
       for each $k \in K$ do
3          Start from root
           Search for closest child node according to $D$
           Search for closest subcluster according to $D$
           if number of entries $< T$ then
4              Add $k$ to subcluster $C_{l,l}$
               Update CF of $C_{l,l}$
5          else if number of childs $< B$ then
6              Create a new subcluster $C_{l,l'}$
               Add $k$ to $C_{l,l'}$
               Update CF of the parent node $S_l$
7          else if number of parents $< B$ then
8              Split child nodes and redistribute CF entries according to closest
                 $D$
9          else
10             Split parent nodes
11         end
12     end
13     Update CF entries in CF-tree
14 end
   Output: Set of groups $G_l$(l=1,2,...,L)
```

As explained in **Pseudo-code 1**, the algorithm scans the clusters from the root **(line 3)** and recursively traverses down the CF-tree and chooses the closest node at each level with the smallest average inter-cluster distance D as follows:

$$minD = \left(\frac{\sum_{k=1}^{K_l} \sum_{k'=K_l+1}^{K_l+K_{l'}} (u_k - u_{k'})}{K_l K_{l'}} \right)^{1/2}, \forall k \in K_l, \forall k' \in K_{l'} \tag{8}$$

After defining the candidate child node, a test is performed to find the closest CF-entry and defines if the device can be added to the candidate subcluster without violating the threshold condition. If so, the algorithm groups the node with the chosen entry and updates the CF-entry of the candidate subcluster **(line 4)**. If not, a new entry is created for the node inside the candidate child node without breaking the branching factor condition **(line 5-6)**. Otherwise, the child node is splitted and the utility points are redistributed based on the closest distance criteria to obtain a set of new subclusters that do not break the branching factor constraint **(line 7-8)**. In case the number of childs already reached the maximum, the parent nodes are splitted and the childs are redistributed to the closest parents **(line 9-10)**. After inserting the CF-entry, all CF informations of the path are updated from the inserted information to the root **(line 13)**.

3.2. Dynamic MLE-based Inter-Slicing Algorithm

Knowing that the physical capacity c in terms of radio resources of a GW m is limited. The goal of this scheme is to estimate and reserve the appropriate resources by finding the maximum likelihood buffer demands for each slice l starting by the one with the highest slicing priority. In this work, the traffic that needs to be uploaded follows a Poisson distribution and LoRa servers are aware of the amount of data stored in the buffer B_i of each slice member.

Lemma 1. *Let T_i be the throughput needed by each device $i, \forall i \in K_l$ captured at each slicing interval time and identified by a corresponding probability distribution. For a fixed physical capacity, the optimum slicing strategy is to virtually reserve resources for each slice based on the mean throughput of its members.*

Proof: T_i follows a Poisson distribution $\mathrm{P}(\lambda_i)$ where λ_i denotes the throughput needed by device i assigned to slice $l, \forall i \in K_l$. Let $f(T_i|\lambda_i)$ be a probability density function similar to $L(\lambda_i|T_i)$ that represents the likelihood of λ_i given the observed throughput.

$$L(\lambda|T_1, T_2, ..., T_{K_l}) = f(T_1|\lambda_1)f(T_2|\lambda_2)....f(T_l|\lambda_l)$$
$$L(\lambda|T_1, T_2, ..., T_{K_l}) = \prod_{i=1}^{K_l} \frac{e^{-\lambda_i}\lambda_i^{T_i}}{T_i!}$$
$$logL(\lambda|T_1, T_2, ..., T_{K_l}) = log\left[\prod_{i=1}^{K_l} \frac{e^{-\lambda_i}\lambda_i^{T_i}}{T_i!}\right]$$
$$logL(\lambda|T_1, T_2, ..., T_{K_l}) = \sum_{i=1}^{K_l} log\left[\frac{e^{-\lambda_i}\lambda_i^{T_i}}{T_i!}\right]$$
$$logL(\lambda|T_1, T_2, ..., T_{K_l}) = \sum_{i=1}^{K_l}\big[-\lambda + T_i log\lambda - log(T_i!)\big]$$
$$logL(\lambda|T_1, T_2, ..., T_{K_l}) = K_l\lambda_i + \sum_{i=1}^{K_l} T_i log\lambda_i$$

To find the maximum likelihood parameter, the first derivative is applied and solved to zero.

$$\frac{\partial logL(\lambda|T_1, T_2, ..., T_{K_l})}{\partial\lambda} = -K_l + \frac{\sum\limits_{i=1}^{K_l} T_i}{\lambda_i} = 0$$
$$\widehat{\lambda_i} = \frac{\sum\limits_{i=1}^{K_l} T_i}{K_l}, \forall i \in K_l$$

Hence, $\widehat{\lambda_i}$ represents the optimal parameter estimation which proves that the optimal slicing decision is to consider the mean throughput of each slice members. However, slices are not equal in terms of priority. Therefore, the resource on GWs will be dynamically allocated to the most urgent slice starting by the channel with the highest reliability. Let $\Theta_i = \widehat{\lambda}_i / \sum_{i=1}^{l} T_i$ be the slicing rate based on which the algorithm reserves for each slice a capacity $c_{i,m} = c_m.\Theta_i, \forall i \in L$. **Pseudo-code 2** summarizes the intra-slicing algorithm and starts with the most critical slice **(line 2)**. Depending on the slicing strategy, the algorithm equally reserves the bandwidth between slices based on a straight-forward fixed slicing (FS) **(line 14-16)** or estimates the needed throughput $\widehat{\lambda}_i$ of all slice l members in the case of Dynamic Slicing (DS) strategy, defines Θ_l for channels reservation and reserve a part of the bandwidth on all LoRa GWs in a similar manner **(line 3-7)**. If the adaptive dynamic slicing (ADS) was adopted, slicing rate of each slice Θ_l varies from a GW to another because in this case, MLE estimates throughput of each slice members deployed in the range of the corresponding GW m **(line 8-14)**. The algorithm moves next to the following slice, repeats the process and stops when no resources are left for reservation.

Pseudo-code 2 Adaptive Dynamic Intra-Slicing Algorithm

```
Input  : Capacities $c_m$, $c'_n$; Number of slices $L$;
         Set of Throughput Requirements $T_l$
1  begin
2      Put slices in decreasing order based on priority $sp_l$
       if method=DS then
3          for each GW $m$ do
4              for each slice $l \in L$ do
5                  Apply MLE Estimation based on the throughput
                    required by all slice $l$ members
                    Define Slicing Rate $\Theta_l$ and Reserve capacity $c_{l,m}$
6              end
7          end
8      else if method=ADS then
9          for each GW $m$ do
10             for each slice $l \in L$ do
11                 Apply MLE Estimation based on the throughput
                    required by slice $l$ members in the range of GW $m$
                    Define Slicing Rate $\Theta_l$ and Reserve capacity $c_{l,m}$
12             end
13         end
14     else
15         Reserve capacity $c_{l,m}$ equally between slices
16     end
17 end
Output: Set of resources reserved for each slice $l$
```

3.3. Intra-Slicing Resource Allocation Algorithm

After defining and reserving the radio resources for each slice, the goal in this section is to maximize the utility function of slice members. Here, utility function for each slice is computed based on multiple criteria weights w_r and w_{ld} for reliability and load respectively manipulated using the analytical and hierarchy process (AHP) approach. The latter is proved as a very decent method for multi-criteria decisions and was adopted in many applications [29].

Based on the QoS table proposed in **Table 3**, one can note that in IoT, devices can be classified into three categories:

QCI	SLice ID	Packet Delay Budget	Services	Percentage of IoT flows
5	1	<100 ms	Surveillance and Emergency Alerting	10 %
1-2	2	100-1000 ms	Health Sensors	15 %
3-4	2	100-1000 ms	Home Security System	15 %
6	3	>1000 ms	Smart Metering Applications	60 %

Table 3: Application Parameters [30]

High critical communications (HCC) slice: requires the highest slicing priority due to urgency and reliability requirements of its members, i.e: surveillance, emergency alerting and alarm monitoring. Based on **Eq. 9**, U_{HCC} is computed to define the utility for critical communications with $\sigma_r = SINR_{k,l,m}/SINR_{max}$ the rate of reliability of SINR that a device k achieves on a flow $f_{k,l,m}$ over the highest flow reliability that can be achieved through slice l and δ_r, a binary variable that guarantees a minimum threshold when searching for the highest reliability links.

$$U_{HCC} = \delta_r(\sigma_r w_r + \sigma_{ld} w_{ld}) \quad with \quad \delta_r \in \{0,1\} \tag{9}$$

Medium critical communications (MCC) slice: requires lower priority consideration and are less critical in terms of delay. This slice presents a trade-off between reliability and load, i.e: health sensors and home security systems.

$$U_{MCC} = \sigma_r w_r + \sigma_{ld} w_{ld} \tag{10}$$

Low critical communications (LCC) slice: requires the lowest priority due to their non-guaranteed data rate and delay-tolerant QoS requirements, i.e: smart metering applications.

$$U_{LCC} = \sigma_{ld} w_{ld} \tag{11}$$

The algorithm searches in each slice for the gateway that offers the most robust and reliable link with lowest delay [31], finds the highest U_{HCC} metric and allocates resources accordingly. Increasing the number of devices will decrease the reliability of links due to congestion. In some cases, the most reliable link may be overloaded due to the increasing number of devices and should not be taken into consideration. Hence in **Eq. 10**, U_{MCC} is defined to search for the flow that gives the best trade-off solution and offers the highest reliability with the lowest possible load. And finally in **Eq. 11**, LCC slice includes delay-tolerant devices with high packet delay budgets. Therefore, only the load is considered in the latter without taking reliability into consideration.

In **Fig. 2**, a directed network $N = (V, E)$ is considered, where each device k is a source node s uploading traffic to external server considered as sink node t such that $s, t \in V$. Moreover, each GW

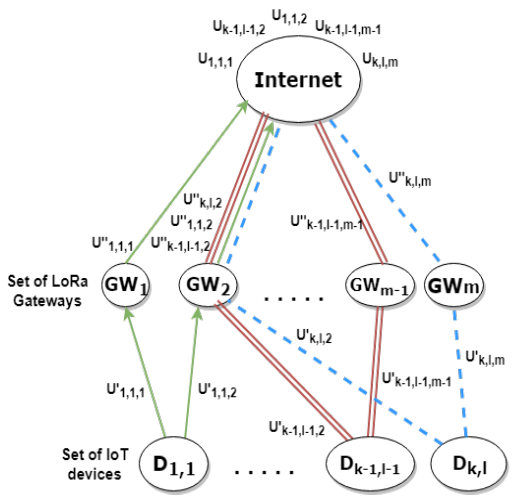


Figure 2: Flow modeling for IoT Network Slicing

m is considered as edge node and bounded by the amount of flow allowed in each slice l. In the latter, the flow that maximizes the utility function of each device k is selected. Without loss of generality, t is assumed that no edges enter the sources or exist sinks. For each edge, the respective utilities $U'_{k,l,m}$ and $U''_{k,l,m}$ are computed in the network based on **Eq. 12** below:

$$U_{k,l,m} = U'_{k,l,m} + U''_{k,l,m} \tag{12}$$

Each LoRa device k assigned to slice l searches for the most efficient virtual flow through GW m with the objective to find the highest utility metric $U_{k,l,m}$ as shown in the **Pseudo-code 3** below.

```
Pseudo-code 3 Max-Utility Inter-Slice Resource Allocation
Input  : Set of LoRa devices $K$, GWs $M$, slices $L$ and capacity $c$
begin
    Initialize flow utilities to null for all $e \in E$
    for each slice $l \in L$ do
        Put devices in decreasing order based on $u_k$
        for each device $k \in K_l$ do
            Draw network $N(V, E)$
            Find path with the highest utility $U_{k,l,m}$
            Allocate device $k$ to $f_{k,l,m}$
            Update capacity $c_{l,m}$
        end
    end
end
Output: Max-Utility flows allocation for LoRa devices
```

4. Performance Evaluation

In uplink, centralized servers enable the opportunity to make efficient slicing configurations based on data traffic in the buffer of each LoRa device. In this work, LoRa model is adopted [19] to simulate the network in the open source NS3 simulator [32]. For additional implementation details, we invite the readers to check the work in [33] which includes a complete description of the model and integrate it in NS3 platform. Each simulation is replicated 50 times and results are plotted with 95% confidence intervals with respect to the parameters shown in the first section of **Table 4**.

Simulation Parameters	
Simulation Time	300 seconds
Slicing Interval Time	50 seconds
Cell Radius	10 KM
Number of replications	50
MAC retransmissions	8
LoRa devices and GWs distribution	Random Uniform
Propagation loss model	Log-distance
Bandwidth	125 kHz
Spreading Factor	{7,8,9,10,11,12}
Confidence intervals	95%
European ISM sub-band	863-870 MHz
Power Consumption Parameters [21]	
Battery Maximum Capacity	950 mAh
LoRa Supply Voltage	3.3V
Amplifier Power's added Efficiency	10%
Connected (Tx/Rx-SF7)	2 dBm
Connected (Tx/Rx-SF8)	5 dBm
Connected (Tx/Rx-SF9)	8 dBm
Connected (Tx/Rx-SF10)	10 dBm
Connected (Tx/Rx-SF11-12)	14 dBm
Standby	0.09 mW
Sleep	0 mW

Table 4: Simulation Parameters

The experiment is realized in a realistic LoRa scenario where devices are choosing a random time for transmission but periodically uploading to LoRa servers small packet payloads that varies from 10 to 20 Bytes. Simulations start with 100 devices to emulate a load of one due to the legal duty-cycle limitations of 1% in the European region [34]. The maximum number connected to a single gateway is limited to 1000 devices following to the scalability study in [35]. LoRa servers allow 8 MAC retransmissions for IoT devices before defining a packet delivery failure. Moreover, LoRa devices and gateways are both placed over a cell of 10 KM radius following to a uniform random distribution. Each device is configured with spreading factors that varies from 7-12 when uploading traffic to LoRa GWs. Each GW is characterized by 8 receiving channels in the 867-868 MHz european sub-band. Based on the **Eq. 13** below, energy consumption is evaluated when the number of LoRa devices increases in each slice.

$$E_{k,l,m} = \frac{p_i^{tx} + p_i^{rx}}{V + epa}.d_{tx/rx} \tag{13}$$

where $E_{k,l,m}$ is the energy consumed by an IoT device, V the LoRa supply voltage, epa the amplifier's added efficiency, d_{tx} the duration of transmission, p_i^{rx} the power of reception and p_i^{tx} the power of transmission that varies between 2 and 14 dBm based on the spreading factor i adopted. An energy module for LoRa module is integrated in NS3, inspired by the one that already exists for Wifi, and is characterized with specific energy parameters and power model for LoRa [21] as listed in the second section of **Table 4** below.

4.1. Proof of Isolation

The very first step before investigating slicing strategies is to prove the isolation concept. Assuming that all devices are uploading packets to a single LoRa GW. The number of LoRa devices is fixed to 20 in *HCC* slice and the rest of devices in the network are assigned to *MCC* and *LCC* slices. **Fig. 3** proves the isolation concept because when the number of devices increases in *MCC* and *LCC* slices, *HCC* members were not affected and the percentage of packet loss rate (PLR%) remained constant and nearly null whereas PLR increased in *MCC* and *HCC* slices in a more congested scenario.

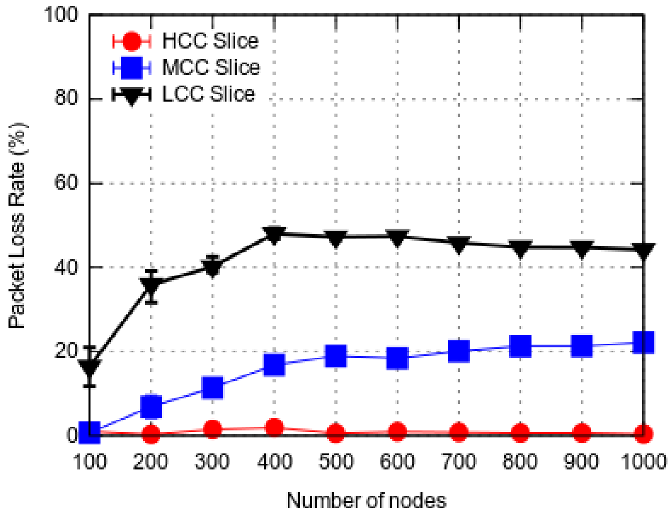


Figure 3: Proof of Isolation

4.2. SF Configuration Variation

In this section, the performance of LoRa slices is evaluated with different SF configurations for a fixed number of 300 devices. Three major slicing strategies are considered, namely *static* configuration where all devices in the cell are configured with the same SF, $dynamic - random$ where each device randomly picks a SF value and finally the $dynamic - adaptive$ where each LoRa device estimates the best SF configuration depending on the receiving power measured from the gateway. In static configurations, the test is repeated for each SF value. However, regarding dynamic configurations, a device with a powerful receiving signal picks a small SF value whereas edge nodes are generally configured with larger SF values. **Table 5** and **Table 6** summarize the mean PLR% for each SF configuration with a fixed and variant packet transmission intervals respectively. Packets may be lost when the gateway is saturated due to the load in the network (Congestion PLR%), due to co-channel rejection (Interference PLR%) or due to lack of sensitivity when the packet is out of range or when it doesn't reach the gateway due to an appropriate SF configuration (Sensitivity PLR%).

4.2.1. Fixed Packets Transmission Period

In this subsection, a decent comparison is performed between SF configuration methods for a fixed packet transmission interval. Each device randomly select a time for transmission and then it periodically uploads a packet each 50s. $static - SF12$ scored the highest PLR percentage. By adopting this configuration, packets transmitted occupy the spectrum for the longest time on air. Therefore, the highest impact on PLR% was reached due to congestion. Packets arrive at constant intervals and cannot be decoded due to gateway saturation. It is noteworthy to mention that no packets were lost due to lack of sensitivity because increasing the spreading factor increases at its turn the range and the probability for successfully decoding a packet. Unlike $static - SF12$, devices with $static - SF7$ configuration lost more than half of the packets. However this time, the main loss was due to lack of sensitivity for packets that are mainly transmitted by edge nodes and cannot reach the gateway because SF7 offers the shortest range capability between SF configurations. Following these assumptions, one can now understand why $static - SF9$ could be placed as a trade-off between range and spectrum occupation with the best overall PLR% between the measured static configurations. As previously mentioned, increasing SF configuration also increases the time occupation of packets sent, which also increases the interference PLR% because the probability of receiving packets with the same SF configuration at the same time will also increase.

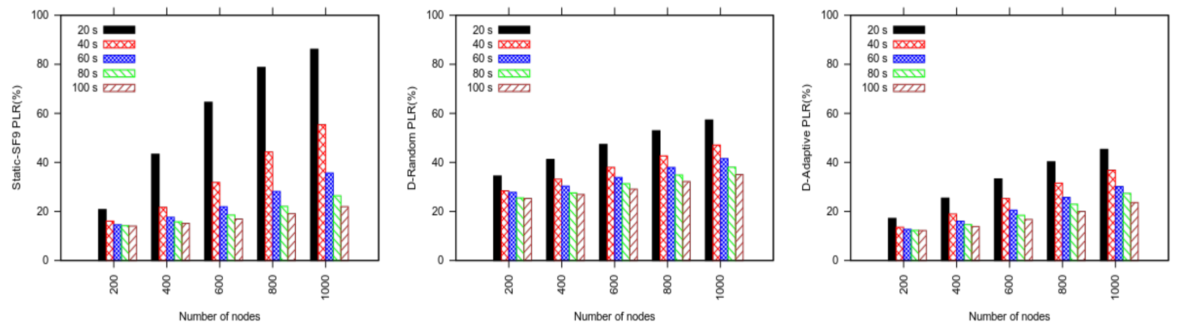
	Slice Name	Static						Dynamic	
		SF7	SF8	SF9	SF10	SF11	SF12	Random	Adaptive
Mean PLR %	Overall	54.14	39.24	39.03	43.93	78.19	94.15	43.02	30.07
Sensitivity PLR %	Overall	76.14	61.75	28.84	2.06	0	0	19.63	0
	HCC	17.99	17.90	17.99	19.74	0	0	18.73	0
	MCC	26.98	26.91	25.97	24.21	0	0	27.75	0
	LCC	55.03	55.20	56.04	56.05	0	0	53.52	0
Congestion PLR %	Overall	22.16	32.35	53.78	63.61	61.9	69.53	69.51	86.43
	HCC	0.12	0.62	2.91	6.91	11.08	15.9	8.99	8.48
	MCC	0.41	1.75	9.42	30.65	46.75	49.76	36.28	34.76
	LCC	99.47	97.63	87.66	62.44	42.17	34.34	54.73	56.76
Interference PLR %	Overall	0.89	4.87	16.15	33.32	37.30	30.47	9.84	12.39
	HCC	7.45	11.85	13.35	15.33	16.44	20.05	16.16	15.43
	MCC	42.40	42.21	40.01	35.88	30.08	28.01	35.12	36.29
	LCC	50.15	45.83	46.64	48.78	53.48	51.95	48.72	48.28

Table 5: Packet Loss Rate Variation with various SF configurations

Table 5 illustrates PLR percentage for each category in each slice. Results show that $dynamic - adaptive$ configuration was the most reliable technique because SFs are dynamically configured on LoRa devices by measuring the receiving power that a GW gets from the device depending on its position. The advantages that the latter configuration present are two-fold: first, depending on how far the device is from the gateway, a smaller distance requires a smaller SF configuration and secondly, the fact of adopting different SFs configuration reduces interference PLR and the probability of collisions. Regardless of the adopted SF configuration method, the urgency character of HCC slice members explains the low percentage in terms of PLR compared to MCC and LCC slices. Urgent packets are not sent as often as other slices which reduces the probability of packets collision.

4.2.2. Variant Packets Transmission Interval

In **Fig. 4**, $static - SF9$ is considered as the best static SF configuration and is compared to $dynamic - random$ and $dynamic - adaptive$ SF configurations when the packets transmission period increases. PLR increases in more congested scenarios. However, it is noteworthy that regardless of the adopted configuration, increasing packets transmission interval decreases the intensity and the congestion in the network. This can be shown with the decreasing behavior of all configurations for a common set of devices simulated. $static - SF9$ meets the performance configuration for high transmission intervals which proves the utility of the former in realistic scenarios where the congestion is normally higher due to the massive number of IoT devices. Moreover, reducing congestion had the same impact on slices. Detailed results are shown in **Table 6** below. For each transmission interval, it is shown how the percentage of PLR is distributed on each slice. Moreover, increasing the transmission period decreased PLR percentage in all slices while having the smallest impact on HCC slice with the highest reliability requirements.



(a) PLR Variation with Static-SF9 configuration (b) PLR Variation with Dynamic-Random configuration (c) PLR Variation with Dynamic-Adaptive configuration

Figure 4: Performance Study with/without considering load in metric calculations

4.3. Fixed (FS) vs Dynamic (DS) vs Adaptive-Dynamic (ADS) Slicing

Following to previous simulations, $dynamic - adaptive$ SF configuration is adopted which has proved its worthiness for this study. The goal in this section is to evaluate the performance of the $fixed$ (FS), $dynamic$ (DS) and the $adaptive-dynamic$ (ADS) slicing strategy. With FS, the number of receiving paths is reserved in an equal manner and is compared to DS and ADS strategies where slicing decisions are performed using MLE throughput estimation for each slice starting with the one with the highest priority. Moreover, the impact of adding load metric to utility calculations is studied for each slicing strategy when the number of LoRa devices assigned to each slice increases. Each slice in a LoRa gateway suffers from congestion, decreasing with it the probability of successfully decoding the packet. Simulation results in **Fig. 5** prove the efficiency of load consideration when computing

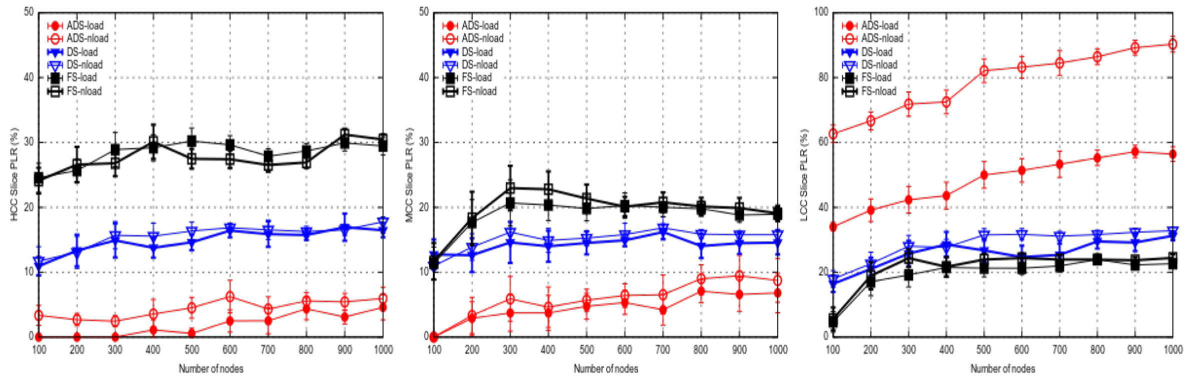
PLR %	PTP (s)	Static-SF9			Dynamic-random			Dynamic-adaptive		
		HCC	MCC	LCC	HCC	MCC	LCC	HCC	MCC	LCC
Sensitivity PLR %	20	24.68	22.98	52.34	18.86	26.12	55.02	0	0	0
	40	20.35	25.81	53.84	18.69	27.43	53.88	0	0	0
	60	19.97	24.15	55.88	18.25	27.36	54.39	0	0	0
	80	18.23	24.22	58.46	18.52	26.82	54.66	0	0	0
	100	17.32	23.92	57.85	18.80	26.96	54.24	0	0	0
Congestion PLR %	20	11.57	45.75	42.68	10.46	39.39	50.15	10.93	39.30	49.77
	40	8.00	37.26	54.74	8.78	36.33	54.89	8.88	36.73	54.39
	60	5.77	24.71	69.52	8.01	31.49	60.50	7.92	32.38	59.70
	80	3.95	13.74	82.31	6.45	29.25	64.30	6.30	29.40	64.30
	100	3.13	8.33	88.55	5.29	23.84	70.87	5.11	24.16	70.73
Interference PLR %	20	15.92	32.20	51.88	16.32	35.63	48.05	16.12	36.77	47.11
	40	15.82	35.53	48.65	15.66	34.95	49.40	15.43	36.98	47.59
	60	15.56	37.12	47.32	14.92	36.28	48.80	15.14	35.81	49.05
	80	14.50	38.37	47.13	15.62	36.58	47.81	15.63	36.20	48.17
	100	14.42	38.14	47.44	15.91	36.16	47.93	13.93	36.53	49.55

Table 6: Packet Loss Rate Variation with various SF configurations

the mean values of slices with and without considering load in metric calculations. Being load-aware improves reliability in the network. When congestion in the network increases, the traffic is balanced to the corresponding slice but on a less-loaded gateway. Reliability on all slices improved especially in *LCC* slice because its most of its members lose previously lost their packets due to congestion. In a comparison between each slicing strategy, *ADS* with load consideration showed the most reliable performance for *HCC* and *MCC* slice as plotted in **Fig. 5a** and **Fig. 5b** respectively. This returns for example to the case of *HCC* slice where the sporadic nature of packet transmissions requires low latency and high reliability with unsteady throughput needs. Therefore, an appropriate estimation of throughput improves slicing and should be considered on each GW separately because it differs from a gateway to another. Moreover, **Fig. 5b** shows that considering load in metric calculations scored approximately 50% improvement in the PLR% of *LCC* slice members. However, this did not prevent *ADS* from being the lowest reliable strategy in *LCC* slice. The reason returns to the fact that *ADS* prioritizes a slice over another and reserves for it the needed bandwidth unlike *FS* where the bandwidth is equally reserved between slices. *LCC* members do not always get the needed bandwidth required for transmission when a small capacity is fixed for this slice. The performance of each slice is evaluated next using *ADS* with a load strategy in terms of energy consumption and the percentage of devices that respected their delay deadlines.

4.3.1. Percentage of Unserved nodes

The efficiency of *ADS* is mainly shown in **Fig. 6** below. With *ADS*, LoRa devices had the highest percentage of devices that respected their delay deadlines compared to *DS* and *FS* strategies with an unserved rate that never exceeded 10% of the total number of packets transmitted. This highlights the importance of including urgency priority in slicing strategies and considering reliability in intra-slice resource allocation algorithm due to its direct impact on the spreading factor configuration and the spectrum occupation time.



(a) HCC Slice PLR Variation (b) MCC Slice PLR Variation (c) LCC Slice PLR Variation

Figure 5: Packet Loss Rate in each Slice with various Slicing Strategies

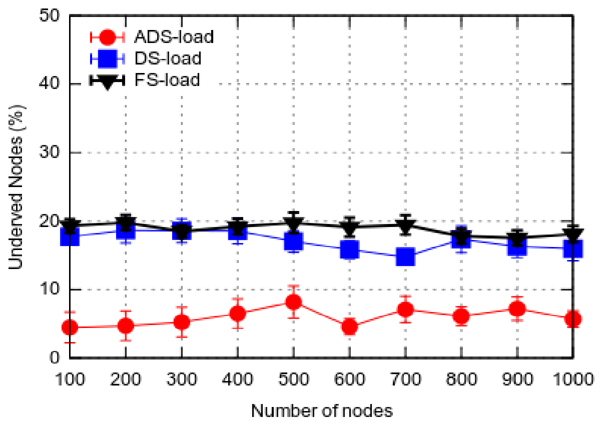


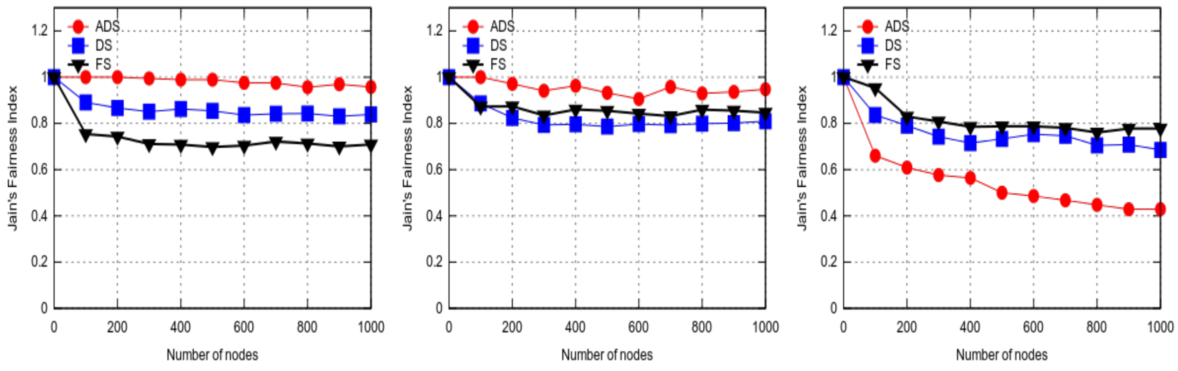
Figure 6: Percentage of Unserved nodes

340 *4.3.2. Jain's Fairness Index*

The goal of this study is to measure the metric that identifies underutilized channels in each slice with FS, DS and ADS strategies. Based on **Eq. 14**, we evaluate in **Fig. 7** the Jain's fairness index of each slicing strategy as follows:

$$Fairness_{index} = \frac{(\sum_{i=1}^{n} x_i)^2}{n \sum_{i=1}^{n} x_i^2} \tag{14}$$

where x_i denotes the normalized throughput of each IoT device and n is the total number of active devices in each slice. Jain's fairness index varies between 0 and 1 with 1 being perfectly fair. ADS strategy provides the best distribution compared to DS and FS strategies as plotted in **Fig. 7a** and **Fig. 7b** below. With FS strategy, resources are divided equally between HCC,MCC and LCC slices. 345 This explains fairness results of FS that are quite similar in all simulated slices. It is noteworthy to mention performance degradation of ADS and DS strategies when moving from urgent to less urgent slices. This is normal due to slicing priority consideration where resource reservation algorithm begins with the most critical slice. However, *ADS* always had a clear upper hand over *DS* strategy in urgent slices except for *LCC* slice where less channels are reserved for its members as shown in **Fig. 7c** below.



(a) HCC Slice Fairness index (b) MCC Slice Fairness index (c) LCC Slice Fairness index

Figure 7: Fairness Evaluation in each Slice with various Slicing Strategies

350 *4.3.3. Energy Consumption*

When increasing the number of nodes, the total energy consumed increases for all the simulated slices, as plotted in **Fig. 8** below. However, *HCC* slice always consumed less energy even when the number of its LoRa members increased. This returns to relation between SF and TP configuration shown in in the second section of **Table 4**. Increasing SF will increase the transmission power and the 355 energy consumption of a slice member. Therefore, the consideration of reliability in utility calculations forces delay-sensitive devices to take the most reliable path with the lowest spreading factor values and transmission power compared to *MCC* and *LCC* slices.

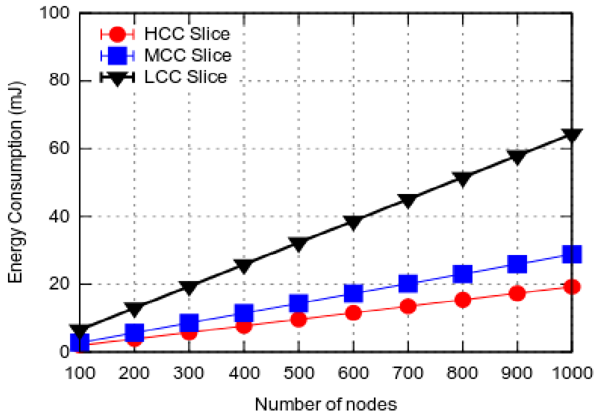


Figure 8: Mean Energy Consumption Variation

5. Conclusion

In this paper, network slicing is evaluated in LoRa technology with the goal of maximizing utilities in each LoRa slice. Therefore, static slicing is improved with an adaptive dynamic inter-slicing algorithm that was proposed based on a maximum likelihood estimation. An intra-slicing algorithm is also introduced that improves resource allocation to meet the QoS requirements of each slice. Numerical results show the effectiveness of the proposed adaptive dynamic slicing strategy and how it outperformed static and dynamic slicing and improved the efficiency of LoRa devices in terms of reliability, energy consumption and the percentage of satisfied devices with regard to their delay requirements. However, there's still a room to improve the proposed slicing strategy in terms of reliability and energy consumption. It would be interesting to focus in the future on improving the energy efficiency in LoRa network slicing, by optimizing LoRa parameters configuration without degrading the QoS performance in the network.

References

1. Wixted, A.J., Kinnaird, P., Larijani, H., Tait, A., Ahmadinia, A., Strachan, N.. Evaluation of lora and lorawan for wireless sensor networks. In: *SENSORS, 2016 IEEE*. IEEE; 2016:1–3.
2. Li, L., Ren, J., Zhu, Q.. On the application of lora lpwan technology in sailing monitoring system. In: *Wireless On-demand Network Systems and Services (WONS), 2017 13th Annual Conference on*. IEEE; 2017:77–80.
3. Vatcharatiansakul, N., Tuwanut, P., Pornavalai, C.. Experimental performance evaluation of lorawan: A case study in bangkok. In: *Computer Science and Software Engineering (JCSSE), 2017 14th International Joint Conference on*. IEEE; 2017:1–4.
4. Mikhaylov, K., Petäjäjärvi, J., Janhunen, J.. On lorawan scalability: Empirical evaluation of susceptibility to inter-network interference. In: *Networks and Communications (EuCNC), 2017 European Conference on*. IEEE; 2017:1–6.
5. Georgiou, O., Raza, U.. Low power wide area network analysis: Can lora scale? *IEEE Wireless Communications Letters* 2017;6(2):162–165.
6. Bor, M.C., Roedig, U., Voigt, T., Alonso, J.M.. Do lora low-power wide-area networks scale? In: *Proceedings of the 19th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems*. ACM; 2016:59–67.
7. Bor, M., Vidler, J.E., Roedig, U.. Lora for the internet of things. *International Conference on Embedded Wireless Systems and Networks* 2016;.
8. Croce, D., Gucciardo, M., Tinnirello, I., Garlisi, D., Mangione, S.. Impact of spreading factor imperfect orthogonality in lora communications. In: *International Tyrrhenian Workshop on Digital Communication*. Springer; 2017:165–179.
9. Waret, A., Kaneko, M., Guitton, A., Rachkidy, N.E.. Lora throughput analysis with imperfect spreading factor orthogonality. *arXiv preprint arXiv:180306534* 2018;.

10. Bor, M., Roedig, U.. Lora transmission parameter selection. *International Conference on Distributed Computing in Sensor Systems* 2017;.

11. Lim, J.T., Han, Y.. Spreading factor allocation for massive connectivity in lora systems. *IEEE Communications Letters* 2018;22(4):800–803.

12. Nakao, A., Du, P., Kiriha, Y., Granelli, F., Gebremariam, A.A., Taleb, T., Bagaa, M.. End-to-end network slicing for 5g mobile networks. *Journal of Information Processing* 2017;25:153–163.

13. Gadallah, Y., Ahmed, M.H., Elalamy, E.. Dynamic lte resource reservation for critical m2m deployments. *Pervasive and Mobile Computing* 2017;40:541–555.

14. Rezende, P.H., Madeira, E.R.. An adaptive network slicing for lte radio access networks. In: *Wireless Days (WD), 2018*. IEEE; 2018:68–73.

15. Hao, Y., Tian, D., Fortino, G., Zhang, J., Humar, I.. Network slicing technology in a 5g wearable network. *IEEE Communications Standards Magazine* 2018;2(1):66–71.

16. Delgado, C., Canales, M., Ortín, J., Gállego, J.R., Redondi, A., Bousnina, S., Cesana, M.. Joint application admission control and network slicing in virtual sensor networks. *IEEE Internet of Things Journal* 2018;5(1):28–43.

17. Kalør, A.E., Guillaume, R., Nielsen, J.J., Mueller, A., Popovski, P.. Network slicing for ultra-reliable low latency communication in industry 4.0 scenarios. *arXiv preprint arXiv:170809132* 2017;.

18. Adelantado, F., Vilajosana, X., Tuset-Peiro, P., Martinez, B., Melia-Segui, J., Watteyne, T.. Understanding the limits of lorawan. *IEEE Communications Magazine* 2017;55(9):34–40.

19. Magrin, D., Centenaro, M., Vangelista, L.. Performance evaluation of lora networks in a smart city scenario. In: *Communications (ICC), 2017 IEEE International Conference on*. IEEE; 2017:1–7.

20. Group, I..W., et al. Ieee standard for local and metropolitan area networkspart 15.4: Low-rate wireless personal area networks (lr-wpans). *IEEE Std* 2011;802:4–2011.

21. Blenn, N., Kuipers, F.. Lorawan in the wild: Measurements from the things network. *arXiv preprint arXiv:170603086* 2017;.

22. Goursaud, C., Gorce, J.M.. Dedicated networks for iot: Phy/mac state of the art and challenges. *EAI endorsed transactions on Internet of Things* 2015;.

23. Zuniga, J.C., Ponsard, B.. Sigfox system description. *LPWAN@ IETF97, Nov 14th* 2016;.

24. Ratasuk, R., Vejlgaard, B., Mangalvedhe, N., Ghosh, A.. Nb-iot system for m2m communication. In: *Wireless Communications and Networking Conference (WCNC), 2016 IEEE*. IEEE; 2016:1–5.

25. Cattani, M., Boano, C.A., Römer, K.. An experimental evaluation of the reliability of lora long-range low-power wireless communication. *Journal of Sensor and Actuator Networks* 2017;6(2):7.

26. Zhang, T., Ramakrishnan, R., Livny, M.. Birch: an efficient data clustering method for very large databases. In: *ACM Sigmod Record*; vol. 25. ACM; 1996:103–114.

27. Zhang, T., Ramakrishnan, R., Livny, M.. Birch: A new data clustering algorithm and its applications. *Data Mining and Knowledge Discovery* 1997;1(2):141–182.

28. Fahad, A., Alshatri, N., Tari, Z., Alamri, A., Khalil, I., Zomaya, A.Y., Foufou, S., Bouras, A.. A survey of clustering algorithms for big data: Taxonomy and empirical analysis. *IEEE transactions on emerging topics in computing* 2014;2(3):267–279.

29. Vaidya, O.S., Kumar, S.. Analytic hierarchy process: An overview of applications. *European Journal of operational research* 2006;169(1):1–29.

30. AlQahtani, S.A.. Analysis and modelling of power consumption-aware priority-based scheduling for m2m data aggregation over long-term-evolution networks. *IET Communications* 2017;11(2):177–184.

31. Rogier, B.. Network performance : Links between latency throughput and packet loss. 2016. URL: `https://www.performancevision.com/blog/network-performance-links-between-latency-throughput-and-packet-loss/`.

32. NSNAM, . ns-3.28 Documentation. `https://www.nsnam.org/ns-3-28/`; 2018. [Online; accessed 27-November-2018].

33. Magrin, D.. LoRaWAN Module Documentation. `https://github.com/signetlabdei/lorawan/blob/master/doc/lorawan.rst`; 2016. [Online; accessed 27-November-2018].

34. Augustin, A., Yi, J., Clausen, T., Townsley, W.M.. A study of lora: Long range & low power networks for the internet of things. *Sensors* 2016;16(9):1466.

35. Haxhibeqiri, J., Van den Abeele, F., Moerman, I., Hoebeke, J.. Lora scalability: A simulation model based on interference measurements. *Sensors* 2017;17(6):1193.