Broadcasting and Sharing of Parameters in an IoT Network by Means of a Fractal of Hilbert Using Swarm Intelligence

Jaime Moreno(✉), Oswaldo Morales, Ricardo Tejeida, and Juan Posadas

Instituto Politécnico Nacional, Escuela Superior de Ingeniería Mecánica y Eléctrica, Campus Zacatenco, Mexico city, Mexico
jemoreno@esimez.mx

Abstract. Nowadays, thousand and thousand of small devices, such as Microcontroller Units (MCU's), live around us. These MCU'not only interact with us turning on lights or identifying movement in a House but also they perform small and specific tasks such as sensing different parameters such as temperature, humidity, CO_2, adjustment of the environmental lights. In addition there is a huge kind of these MCU's like SmartPhones or small general purpose devices, ESP8266 or RaspberryPi3 or any kind of Internet of Things (IoT) devices. They are connected to internet to a central node and then they can share their information. The main goal of this article is to connect all the nodes in a fractal way without using a central one, just sharing some parameters with two adjacent nodes, but any member of these nodes knows the parameters of the rest of these devices even if they are not adjacent nodes. With a Hilbert fractal network we can access to the entire network in real time in a dynamic way since we can adapt and reconfigure the topology of the network when a new node is added using tools of Artificial Intelligence for its application in a Smart City.

Keywords: IoT · Adaptive algorithms · Swarm Intelligence
Hilbert fractal · ESP8266 · RaspberryPi3

1 Introduction

In recent years, the interconnection of Internet of Things (IoT) networks have received increasing interest from many academic and engineering fields. A hot topic is to Broadcasting and sharing of parameters in this kind of networks in a efficient way. So far, convenient and economical methods are very necessary. IoT networks has been considered as a new generation of networks that combine environmental sensing or temperature, humidity, CO_2, adjustment of the environmental lights with data transmission and processing through wireless communication techniques or Wireless Sensor Networks (WSN) [1,2], provide a better choice for broadcasting and sharing of parameters. Various sensors are

I. Batyrshin et al. (Eds.): MICAI 2018, LNAI 11288, pp. 62–75, 2018.
https://doi.org/10.1007/978-3-030-04491-6_5

attached in homes or wearable to acquire data containing physical or environmental states of these parameter. After sensor data collection, transmission, sharing processing and analyzing, structural problems such as adding new members in the network and communicate these parameters in a central way as a star configuration can be successfully predicted and adapted, so that the upcoming measurements caused by a conventional request may be answered [3,4]. In addition, it is important to highlight that by 2020, The Internet of Things will have achieved *critical mass*. Linking enormous intelligence in the cloud to billions od mobile devices and having extremely inexpensive sensors and tag embedded in and on everything, will deliver enormous amount of new value to almost every human being. The full benefits, in terms of health, safety and convenience, will be enormous [4]. The perspective for this year, in this way will be 4 billion of connected people with a revenue of $4 Trillion USD because more than 25 millions of apps will be downloaded from the principal app-stores, thus we need approximately more than 25 billion MCU's which will generate 50 trillion of Gigabytes of Data inside the global IoT network.

Furthermore, The increasing amount of IoT devises in a common or a conventional House, Fig. 1 shows a usual Floor Plan. In a IoT network contains a WiFi Access Point (WAP) that connect all the IoT devises to a Internet or Intranet networking as the main node. If a MCU want to share some parameters with another MCU, these MCU's must communicate with the WAP. Even if they physically are very near each other if they do not have connection with the WAP they are not able to establish communication and sharing parameters.

Which is why in this paper, an effective method for broadcasting and sharing of parameters in WSN with Swarm Intelligence is proposed [5–7]. Here we use a large amount of networked sensors for realtime sensing and monitoring of different parameters in order to make efficient a IoT network and its application in a forthcoming Smart City. The further sections of this paper is organized as follows: General Description of the proposal is discussed in Sect. 2. In Sect. 3, a detailed description of Theoretical Framework is given describing Embedded Systems in the IoT network and Hilbert space-filling Curve. In Sect. 4, the description of the main scheme is given. In Sect. 5, simulation and performance comparison are presented to verify correctness and feasibility of the proposed scheme. Finally, some conclusions are drawn in Sect. 6.

2 General Description of the Proposal

This proposal is based in main work made by Beni and Wang [8] introduced the expression of Swarm Intelligence (SI) in their research of cellular robotic systems in 1993. The concept of SI is employed in work on Artifcial Intelligence (AI). SI is a computational intelligence technique which is based on the collective behavior of decentralized, self-organized systems. A typical SI system is made up of a group of simple agents which interact locally with each other and with the environment surrounding them. MCU's in an SI system follow simple rules and act without the control of any centralized entities. However, the social interactions

between such MCU's may generate enormous benefits and often lead to a smart global behavior.

SI takes the full advantage of the swarm, therefore, it's able to provide optimized solutions, which ensure high robustness, flexibility and low cost, for large-scale sophisticated problems without a centralized control entity.

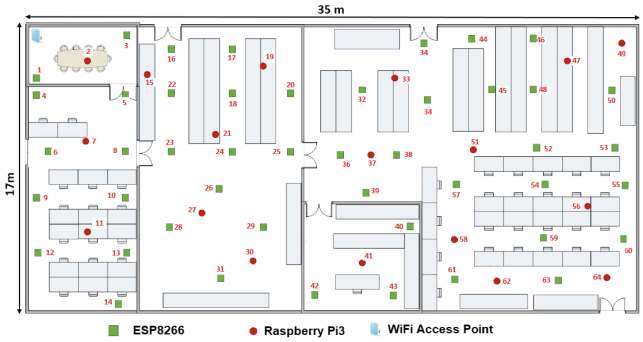


Fig. 1. Floor Plan for exemplifying the proposed algorithm.

Which is why, this proposal is divided in 3 main parts based on SI:

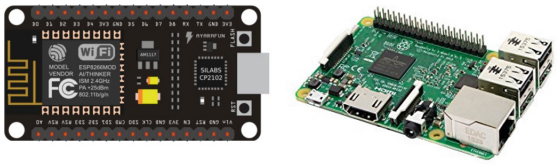
1. Selecting the main or initial MCU,
2. Generation of the topology in a given Floor Plant, and
3. Indexing all the MCU's inside the IoT network.

Henceforth, we use the Floor Plant depicted in Fig. 1, which consists of one WPA and several ESP8266 and RaspberryPi3, namely 64 embedded systems. In addition, in the first part the nearest MCU to the WPA is selected as the main MCU. Then, measuring the bandwidth in Mbps (mega bits per second) we can predict a general topology for a given Floor Plant, when a new MCU is added the algorithm recalculates a new topology. Ones the topology of the network is calculates every member of the IoT network is indexed by the Hilbert Fractal, then we measure the effectiveness of the methodology sharing parameter along the network and then Wi-Fi Peer-to-Peer (P2P) and SoftAP algorithms are configured.

3 Theoretical Framework

3.1 Embedded Systems in the IoT Network

NodeMCU ESP8266 embedded system depicted by Fig. 2(a) is a development card similar to Arduino, especially oriented to IoT. It is based on the System on Chip (SoC) ESP8266, a highly integrated chip, designed for the needs of a connected world. It integrates a powerful processor with 32 bit architecture (more powerful than the Arduino Due) and Wifi connectivity.



(a) (b)

Fig. 2. Embedded systems in the IoT network (a) NodeMCU ESP8266 embedded system (MCU-ESP8266) and (b) Raspberry Pi 3 embedded system (MCU-RPi3)

For the development of applications you can choose between the Arduino and LUA languages. When working within the Arduino environment we can use a language we already know and make use of a simple IDE to use, in addition to making use of all the information about projects and libraries available on the Internet. The user community of Arduino is very active and supports platforms such as ESP8266. NodeMCU comes with a pre-installed firmware which allows us to work with the interpreted language LUA, sending commands through the serial port (CP2102). The NodeMCU and Wemos D1 mini cards are the most used platforms in IoT projects. It does not compete with Arduino, because they cover different objectives, it is even possible to program NodeMCU from the Arduino IDE. The NodeMCU card is specially designed to work in breadboard. It has an on-board voltage regulator that allows it to feed directly from the USB port. The input/output pins work at 3.3V. The CP2102 chip handles USB-Serial communication.

NodeMCU has the following technical specifications:

- Power Voltage (USB): 5 V DC
- Input/Output Voltage: 3.3 V DC
- SoC: ESP8266 (Module ESP-12)
- CPU: Tensilica Xtensa LX3 (32 bit)
- Clock Frequency: 80 MHz/160 MHz
- Instruction RAM: 32 KB
- Data RAM: 96 KB
- External Flash Memory: 4 MB
- GPIO Digital Pins: 17 (can be configured as PWM at 3.3 V)
- Analogue Pin ADC: 1 (0-1 V)
- UART: 2
- USB-Serial Chip: CP2102
- FCC certification
- Antenna in PCB
- 802.11 b/g/n
- Wi-Fi Direct (P2P), soft-AP
- Integrated TCP/IP Protocol Stack
- PLLs, regulators, DCXO and integrated power management
- Output power of + 19.5 dBm in 802.11b mode
- Leakage current less than 10 uA
- STBC, 1 × 1 MIMO, 2 × 1 MIMO
- A-MPDU & A-MSDU aggregation & 0.4 ms guard interval
- Wake up and transmit packets in < 2 ms
- Standby power consumption $<$ 1.0 mW (DTIM3)

Raspberry Pi 3 embedded system is one the most advanced thin client solution of the brand so far, which is shown by Fig. 2(b). Since the first Raspberry, called Raspberry Pi Modelo B, was introduced in 2012, more than 8 million units have been sold. Then it had only 256 MB of RAM. The new versions have been adding improvements in their features, thus reaching the Raspberry 3 pi Model B (MCU-RPi3).

In this work we would like to talk to you about some of the features and advantages of the MCU-RPi3, in case you are looking for a thin client to work

with. The Raspberry Pi 3 thin client includes a series of advantages and novelties compared to the previous model, the Raspberry Pi 2 Model B. Among them, it is worth mentioning some of its main features. First, it has a much faster and more powerful processor than the versions. It is an ARM Cortex A53 with 4 cores, 1.2 GHz and 64 bits. Its performance is at least 50% higher than version 2 of Raspberry. The new thin client of Raspberry fulfilled another of the wishes of its main clients: wireless Wifi connectivity and Bluetooth. The previous models could only be connected through an Ethernet cable or through wireless USB adapters, so this version makes things much easier. The MCU-RPi3 includes a Bluetooth 4.1 and an 802.11n wireless Wifi network card. For the rest, it should be said that the new model hardly changes the size and only adds some changes in the position of the LED lights. The MCU-RPi3 account, on the other hand, with the same features that already included version 2.1 GB of RAM, fourth USB ports, Ethernet and HDMI port, Micro SD, CSI camera, etc. It is also fully compatible with the Zero and 2 versions. The Raspberry brand recommends using it in schools or for general use. Other recommendations of the brand for integrated projects are the Pi Zero and the Model A +. One of the great advantages of the MCU-RPi3 is precisely its price. Many companies, schools and centers use thin clients to create a more secure structure, but it can be given all kinds of uses: from creating a mini mail server to a low-end FM station.

The full specs for the Raspberry Pi 3 include:

- CPU: Quad-core 64-bit ARM Cortex A53 clocked at 1.2 GHz
- GPU: 400 MHz VideoCore IV multimedia
- Memory: 1 GB LPDDR2-900 SDRAM (i.e. 900 MHz)
- USB ports: 4
- Video outputs: HDMI, composite video (PAL and NTSC) via 3.5 mm jack
- Network: 10/100 Mbps Ethernet and 802.11n Wireless LAN
- Peripherals: 17 GPIO plus specific functions, and HAT ID bus
- Bluetooth: 4.1
- Power source: 5 V via MicroUSB or GPIO header
- Size: 85.60 mm × 56.5 mm
- Weight: 45 g (1.6 oz)

Wi-Fi Peer-to-Peer (P2P) and SoftAP allow easy, direct connection among Wi-Fi devices, anywhere. There is no need for an Access Point. Through negotiation, one device becomes the Group Owner and the other the client. Initiation of the connection is designed to be quick and easy. Implementation does not require addition of new hardware, allowing existing smxWiFi users to add them to their products with just a software update. Wi-Fi Peer-to-Peer is a protocol designed to replace the legacy ad-hoc protocol for interconnecting Wi-Fi devices. Improvements over ad-hoc include easier connection and the latest security, 802.11i. It was designed to satisfy the growing need for dynamic connections among groups of electronics. It allows OEMs to create products that can interoperate with each other and with common consumer devices, such as phones, tablets, cameras, and printers. As with any Wi-Fi device, the range is up to 200m, making connection convenient even when devices are not in immediate proximity to one another.

SoftAP gives a device limited Access Point capabilities. When combined with Wi-Fi P2P, it allows the device to be the Group Owner. Without it, a device can participate only as a client.

3.2 Hilbert Space-Filling Curve

The Hilbert curve is an iterated function that is represented by a parallel rewriting system, concretely a L-system L-system. In general, a L-system structure is a tuple of four elements:

1. *Alphabet*: the variables or symbols to be replaced.
2. *Constants*: set of symbols that remain fixed.
3. *Axiom* or *initiator*: the initial state of the system.
4. *Production rules*: how variables are replaced.

In order to describe the Hilbert curve alphabet let us denote the upper left, lower left, lower right, and upper right quadrants as $\mathcal{W}$, $\mathcal{X}$, $\mathcal{Y}$ and $\mathcal{Z}$, respectively, and the variables as $\mathcal{U}$ (*up*, $\mathcal{W} \rightarrow \mathcal{X} \rightarrow \mathcal{Y} \rightarrow \mathcal{Z}$), $\mathcal{L}$ (*left*, $\mathcal{W} \rightarrow \mathcal{Z} \rightarrow \mathcal{Y} \rightarrow \mathcal{X}$), $\mathcal{R}$ (*right*, $\mathcal{Z} \rightarrow \mathcal{W} \rightarrow \mathcal{X} \rightarrow \mathcal{Y}$), and $\mathcal{D}$ (*down*, $\mathcal{X} \rightarrow \mathcal{W} \rightarrow \mathcal{Z} \rightarrow \mathcal{Y}$). Where $\rightarrow$ indicates a movement from a certain quadrant to another. Each variable represents not only a trajectory followed through the quadrants, but also a set of 4^m transformed pixels in m level. The structure of our Hilbert Curve representation does not need fixed symbols, since it is just a linear indexing of pixels.

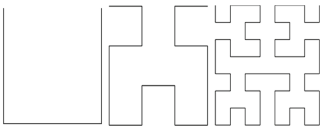


Fig. 3. First three levels of a Hilbert fractal curve. Axiom = $\mathcal{U}$ employed for this work.

The original work by Hilbert [9] proposes an axiom with a $\mathcal{D}$ trajectory, while we propose to start with an $\mathcal{U}$ trajectory (Fig. 3). Our proposal is based on the most of the energy is concentrated in the nearest MCU, namely at the right or left. The first three levels of a Hilbert Curve are portrayed in left-to-right order by Fig. 3. In this way the production rules of the Hilbert Curve are defined by:

- $\mathcal{U}$ is changed by the string $\mathcal{LUUR}$
- $\mathcal{L}$ by $\mathcal{ULLD}$
- $\mathcal{R}$ by $\mathcal{DRRU}$
- $\mathcal{D}$ by $\mathcal{RDDL}$.

In this way high order curves are recursively generated replacing each former level curve with the four later level curves. The Hilbert Curve has the property of remaining in an area as long as possible before moving to a neighboring spatial region. Hence, correlation between neighbor pixels is maximized, which is an important property in image compression processes. The higher the correlation at the preprocessing, the more efficient the data transmission.

4 Broadcasting and Sharing of Parameters in an IoT Network

4.1 Selecting the Main or Initial MCU

Our algorithm considers that the WPA is on random position inside the room, but with connection with at least one MCU in the IoT network. At the beginning all MCU's try to be connected to the WPA, if they do not establish the connection, the WiFi Direct option is enabled. MCU's that connect to the WPA disable the Direct WIFI mode until instructed otherwise. So, the WPA sets up with the broadcasting mode sending packages in order to measure the bandwidth of every member in the IoT network. Henceforth, we use a MCU Network Identifier (MNI), which contains the type of MCU, either MCU-ESP8266 or MCU-RPi3, and byte tag for identification inside the IoT network. For example a MNI=MCU-ESP8266-i refers a ESP8266, which was identified as the *ith* embedded system inside the room, no matter the type or characteristics of the MCU. Due to the material with which the room walls are made, connection is made to only with some MCU's, in quantity, these must be less or equal than 4^n. Hence, The IoT network is made up of 4^n elements, as much, that must be interconnected and it is defined by Eq. 1.

$$IoTe = \sum_{i=1}^{4^n} MCU_i \tag{1}$$

where $IoTe$ is the number of embedded systems in the network, i is index of the MCU Network Identifier, and n is the level of the Hilbert fractal, which is defined by the Eq. 2.

$$n = \lceil \log_4 (IoTe) \rceil \tag{2}$$

Which is why we can define the $MCU_{i=1}$ as the nearest MCU connected with the WPA because it get the highest link speed of the $IoTe$ links. So, MCU_1 is indexed as the first node in the network.

4.2 Generation of the Topology in a Given Floor Plant

Ones $MCU_{i=1}$ is identified as the main node in the IoT network. Hence, the $IoTe$ embedded systems, $MCU_{i=1}$ included, enable WiFi Direct mode and full a table with the bandwidth of the nodes are next to them, in order to generate a List of Reachable Nodes (LRN). Every MCU generates a particular LNR, then a the proposed IoT network can generate 4^n LNR's. In addition, every LRN_i, contains the bandwidth of all MCU_i, which it establish connection. In order to belong to the IoT network every MCU_i must connect to at least one link with other MCU_i LRN's are shared and $\sum_{i=1}^{4^n} MCU_i$ know the way to any node in the network, namely everyone knows the topology of the network. This second step has a paradigm neither optimized nor intelligent. So, it is important to employ some tools of SI to improve these initial results.

4.3 Indexing All the MCU's Inside the IoT Network

Thus, each node has an initial index in the network which, for the time being, is not optimized. To optimize it, all the elements must be numbered according to the Hilbert fractal and the position where each node passes.

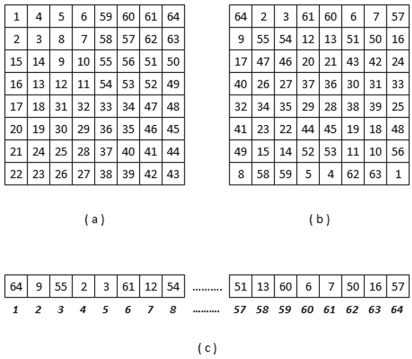


Fig. 4. (a) Matrix θ, (b) Matrix $\mathcal{H}$, and (c) Vector $\overrightarrow{\mathcal{H}}$

A linear indexing is developed in order to identify the MCU_i matrix array into a vector. Let us define the Microcontroller Units matrix array as $\mathcal{H}$ and the interleaved resultant vector as $\overrightarrow{\mathcal{H}}$, being $2^n \times 2^n$ be the size of $\mathcal{H}$ and 4^n the size of $\overrightarrow{\mathcal{H}}$, where n is the Hilbert curve level. Algorithm 1 generates a Hilbert mapping matrix θ with level n, expressing each curve as four consecutive indexes. The level n of θ is acquired concatenating four different θ transformations in the previous level $n-1$. Algorithm 1 generates the Hilbert mapping matrix θ, where $\overrightarrow{\beta}$ refers a 180° rotation of β and β^T is the linear algebraic transpose of β. Figure 4 shows an example of the mapping matrix θ at level $n = 3$. Thus, each MCU_i at $\mathcal{H}_{(i,j)}$ is stored and ordered at $\overrightarrow{\mathcal{H}}_{\theta_{(i,j)}}$, being $\theta_{(i,j)}$ the location index of it into $\overrightarrow{\mathcal{H}}$. In addition, Fig. 4(b) shows the position inside a room and the MNI

Algorithm 1. Function to generate Hilbert mapping matrix θ of size $2^n \times 2^n$.

Input: n
Output: θ
if $n = 1$ **then**
  $\theta = \begin{matrix} 1 & 4 \\ 2 & 3 \end{matrix}$
else
  $\beta =$ **Algorithm 1** $(n-1)$
  $\theta = \begin{matrix} \beta^T & (\widetilde{\beta})^T + (3 \times 4^{n-1}) \\ \beta + 4^{n-1} & \beta + (2 \times 4^{n-1}) \end{matrix}$

of each MCU_i, namely, the best way to communicate for MCU_i is the MCU_{i+1}, giving as a result the increment of the bandwidth.

That is why the nodes i are consecutive and communicate with the node $i+1$. In the case that in the current topology there is no MCU_i in a certain position i, the index that is closest to $i-1$ is searched and this MCU_i is considered as a No-Significant Node (NSN) otherwise it is considered as Significant Node or (SN). It is important to mention that the node MCU_1 is always SN and it is the only one that establishes a connection with the WPA. In the case that this node MCU_{i+1} is NSN, MCU_i searches a SN incrementing i until finding it. In order to know the significance of the IoT, we propose a String of Network Significance (SNS) with 4^n bits or 2^{2n-3}bytes; 0 for NSN and 1 for SN. The SNS is broadcasted to $\sum_{i=1}^{4^n} MCU_i$ and it takes few nanoseconds. Every MCU_i measures or evaluates certain parameters such as temperature, humidity, CO_2, adjustment of the environmental lights. To differentiate them, we propose a 1-byte marker that specifically indicates which parameter that a sensor measures or MS and its value in Celsius degrees or voltage, for instance. Table 1 shows some examples of parameters that a MCU_i can measures, also a 1-byte MS can indicate up to 256 parameters.

Table 1. Some examples of marker of sensor or MS.

Label	Parameter
00H	Temperature
01H	Humidity
02H	Passive infrared sensors
03H	RFID
04H	Door control
05H	Temperature control
06H	Biorhythm
07H	Biosensors

Each parameter value that measures an MCU_i is represented by a two-byte long Marker, which is divided in three parts: Sign, Exponent ε and Mantissa μ (Fig. 5).

The most significant bit of the marker is the sign of parameter, whether 0 for positive or 1 for negative. The ten least significant bits are employed for the allocation of μ, which is defined by [10] as:

$$\mu = \left\lfloor 2^{10} \left(\frac{Parameter}{2^{R-\varepsilon}} - 1 \right) + \frac{1}{2} \right\rfloor \tag{3}$$

Equation (4) expresses how ε is obtained, which is stored at the 5 remaining bits of the marker

$$\varepsilon = R - \lceil \log_2 |Parameter| \rceil \tag{4}$$

where R is the number of bits used to represent the highest value of a certain parameter, defined as

$$R = \lceil \log_2 [\max \{Parameter\}] \rceil . \tag{5}$$

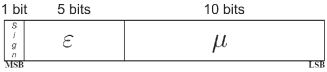


Fig. 5. Structure of the sensor marker or SM.

5 Experimental Results

In this section we extend the exposed in previous section. Also, we use the Floor Plant of the Fig. 1 and the following amount of MCU's:

- 46 MCU-ESP8266
- 18 MCU-RPi3

It is important to note that we use a WPA to connect the IoT network to Internet as initial node with following features: Tenda N300 Wireless WIFI Router WI-FI Repeater Booster Extender 802.11 b/g/n RJ45 with 4 Ports at 300 Mbps.

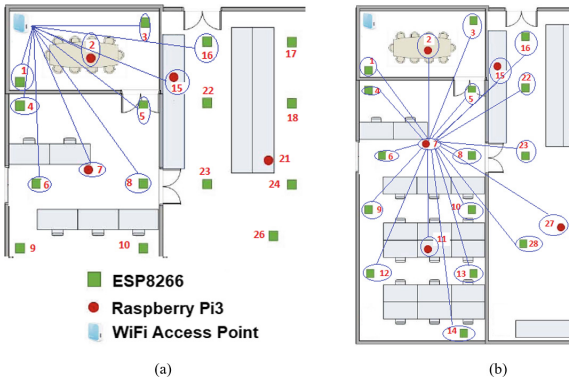


Fig. 6. Experimental results: (a) estimation of link speed every member in the IoT network and (b) reachable Nodes for MCU-RPi3-7

From Fig. 1, Fig. 6(a) shows that the WPA is on the upper right corner. At the beginning all the MCU's try to be connected to the WPA, if they do

not establish the connection, the WiFi Direct option is enabled. MCU's that connect to the WPA disable the Direct WIFI mode until instructed otherwise. So, the WPA sets up with the broadcasting mode sending packages in order to measure the bandwidth of every member in the IoT network. Henceforth, we use a MCU Network Identifier (MNI), which contains the type of MCU, either MCU-ESP8266 or MCU-RPi3, and byte tag for identification inside the IoT intranet. For example a MNI = MCU-ESP8266-8 refers a ESP8266, which was identified as the eighth embedded system inside the room, no matter the type or characteristics of the MCU. Due to the material with which the walls are made, connection is made to only seventeen MCU's the speeds depicted in Table 2, we only show the first ten MCU connected to the WPA. Which is why we can define the MCU-ESP8266-1 as the nearest MCU connected with the WPA because it get a link speed of 199.37 Mbps and the furthest MCU is MCU-ESP8266-16 with 53.14 Mbps. MCU-ESP8266-1 is indexed as the first node in the network.

Table 2. Bandwidth (Mbps) of the first ten MCU's inside the IoT network

MCU network identifier	Linear distance (m)	Actual distance (m)	Bandwidth (Mbps)
MCU-ESP8266-1	3	3	199.37
MCU-ESP8266-4	4	12	117.12
MCU-ESP8266-6	7.5	13	107.98
MCU-RPi3-7	7	11	126.26
MCU-ESP8266-8	9	11.5	121.69
MCU-ESP8266-5	6.5	6.5	167.38
MCU-RPi3-2	3.5	3.5	194.80
MCU-RPi3-15	8	16	80.56
MCU-ESP8266-16	8.5	19	53.14
MCU-ESP8266-3	6	6	171.95

Ones MCU-ESP8266-1 is identified as the main node or Node 1 in the IoT network, all the MCU's, MCU-ESP8266-1 included, enable WiFi Direct mode and full a table with the bandwidth of the nodes are next to them, in order to generate a List of Reachable Nodes (LRN). Every MCU generates a particular LNR, for instance, Fig. 6(b) shows a visual representation of the MNI's whom the MCU-RPi3-7 can connect. According to Fig. 1 our example generates 64 LNR's.

In addition, Table 3 shows the LRN-7, i.e. the bandwidth of the nineteen MCU's, which it establish connection the MCU-RPi3-7. As we can see MCU-ESP8266-6 and MCU-ESP8266-8 obtain the two highest bandwidth for Node 7, with 203.94 Mbps and 201.20 Mbps, respectively.

In addition, Fig. 4(b) shows the position inside a room and the MNI of each MCU_i, namely, the best way to communicate for MCU_{64} is the MCU_9, increasing the bandwidth and so on. In addition, Fig. 7 shows a visual interpretation of the linear indexing making by means of a $n = 3$ Hilbert Fractal.

Table 3. Seventh list of reachable nodes or LRN_7 generated by the MCU-RPi3-7.

MNI	Linear distance (m)	Actual distance (m)	Bandwidth (Mbps)
MCU-ESP8266-1	5.50	10.40	131.74
MCU-ESP8266-4	4.50	4.50	185.66
MCU-ESP8266-6	2.50	2.50	203.94
MCU-ESP8266-9	5.00	5.00	181.09
MCU-ESP8266-12	8.00	8.00	153.67
MCU-RPi3-11	6.00	6.00	171.95
MCU-ESP8266-14	10.50	10.50	130.83
MCU-ESP8266-13	7.50	7.50	158.24
MCU-ESP8266-10	4.50	4.50	185.66
MCU-ESP8266-28	8.20	10.40	131.74
MCU-RPi3-27	9.30	9.70	138.14
MCU-ESP8266-8	2.80	2.80	201.20
MCU-ESP8266-23	6.00	6.00	171.95
MCU-ESP8266-22	7.20	9.80	137.22
MCU-ESP8266-16	8.40	13.50	103.41
MCU-RPi3-15	6.30	11.25	123.97
MCU-ESP8266-5	4.45	4.45	186.12
MCU-ESP8266-3	7.25	7.25	160.53
MCU-RPi3-2	5.30	8.75	146.82

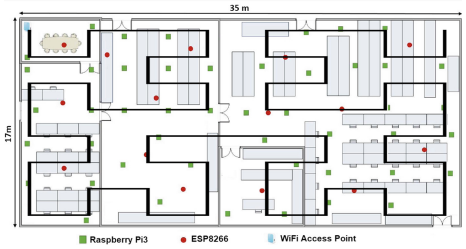


Fig. 7. Floor Plan for conventional house.

We perform one thousand requests from the farthest nodes, i.e. from MCU_{64} to MCU_1 and from MCU_8 to MCU_{57}. On the average the link $MCU_{64} \leftrightarrow MCU_1$ obtains a bandwidth of 194.69 Mbps and for the one $MCU_8 \leftrightarrow MCU_{57}$ 192.35 Mbps. In both in a central topology, we cannot connect neither $MCU_{64} \leftrightarrow MCU_1$ nor $MCU_8 \leftrightarrow MCU_{57}$.

6 Conclusions

Reducing energy consumption is a compulsory objective in the design of any communication protocol for Wireless Sensor Networks. Most of this energy can be saved through member aggregation, given that most of the sensed information is redundant due to geographically collocated sensors. Therefore, efficient broadcasting scheme to consider sharing parameters have been proposed. However, they still suffer from the high communication cost and have not fully resolved the sharing parameters. We proposed a lightweight broadcasting algorithm in wireless sensor networks. From the performance analysis, we can confirm that the IoT network reconfiguration scheme improves both the network lifetime and efficient parameter information than the traditional schemes. Also, we obtain some parameters connected in a certain MCU_i of all the IoT network in microseconds, what meets the definition of real time definition. In addition, this proposal make use of Swarm Intelligence, since every MCU_i learns of the rest of the members of the IoT network and knows what happen even it is not physically connected to other member.

Acknowledgment. This article is supported by National Polytechnic Institute (Instituto Poliécnico Nacional) of Mexico by means of Project No. 20180514 granted by Secretariat of Graduate and Research, National Council of Science and Technology of Mexico (CONACyT). The research described in this work was carried out at the Superior School of Mechanical and Electrical Engineering (Escuela Superior de Ingeniería Mecánica y Eléctrica), Campus Zacatenco.

References

1. Hassan, T., Aslam, S., Jang, J.W.: Fully automated multi-resolution channels and multithreaded spectrum allocation protocol for IoT based sensor nets. IEEE Access **6**, 22545–22556 (2018)
2. Jo, O., Kim, Y.K., Kim, J.: Internet of things for smart railway: feasibility and applications. IEEE Internet Things J. **5**(2), 482–490 (2018)
3. Sandoval, R.M., Garcia-Sanchez, A.J., Garcia-Haro, J.: Improving RSSI-based path-loss models accuracy for critical infrastructures: a smart grid substation case-study. IEEE Trans. Ind. Inform. **14**(5), 2230–2240 (2018)
4. Zhang, C., Ge, J., Pan, M., Gong, F., Men, J.: One stone two birds: a joint thing and relay selection for diverse IoT networks. IEEE Trans. Veh. Technol. **67**(6), 5424–5434 (2018)
5. Li, T., Yuan, J., Torlak, M.: Network throughput optimization for random access narrowband cognitive radio internet of things (NB-CR-IoT). IEEE Internet Things J. **5**(3), 1436–1448 (2018)
6. Sharma, P.K., Chen, M.Y., Park, J.H.: A software defined fog node based distributed blockchain cloud architecture for IoT. IEEE Access **6**, 115–124 (2018)
7. Taghizadeh, S., Bobarshad, H., Elbiaze, H.: CLRPL: context-aware and load balancing RPL for IoT networks under heavy and highly dynamic load. IEEE Access **6**, 23277–23291 (2018)

8. Beni, G., Wang, J.: Swarm intelligence in cellular robotic systems. In: Dario, P., Sandini, G., Aebischer, P. (eds.) Robots Biological Systems: Towards a New Bionics?. NATO ASI Series (Series F: Computer and Systems Sciences), vol. 102, pp. 703–712. Springer, Berlin (1993). https://doi.org/10.1007/978-3-642-58069-7_38
9. Hilbert, D.: Über die stetige Abbildung einer Linie auf ein Flächenstück. Math. Ann. **38**(3), 459–460 (1891)
10. Boliek, M., Christopoulos, C., Majani, E.: Information Technology: JPEG2000 Image Coding System, JPEG 2000 Part I final committee draft version 1.0 ed., ISO/IEC JTC1/SC29 WG1, JPEG 2000, April 2000