ISSN 2007-9737

Optimización de una red definida por software basada en la construcción del fractal de Peano

Oswaldo Morales Matamoros[1], Jesús Jaime Moreno Escobar[1], Ricardo Tejeida Padilla[2], Ana Gabriela Ramírez Gutiérrez[3], Pedro Flores Jiménez[4]

[1] Instituto Politécnico Nacional,
Escuela Superior de Ingeniería Mecánica y Eléctrica,
México

[2] Instituto Politécnico Nacional,
Escuela Superior de Turismo,
México

[3] Universidad Panamericana,
México

[4] Secretaría de Salud,
México

jemoreno@esimez.mx

Resumen. En la actualidad, millones de unidades de microcontroladores (MCU) están conectadas simultáneamente de forma digital para que nuestra vida sea más cómoda. Estos MCU no sólo interactúan con los seres humanos encendiendo las luces o identificando el movimiento en una casa, sino que también realizan tareas pequeñas y específicas, como detectar diferentes parámetros como temperatura, humedad, CO_2, ajuste de las luces ambientales. Hay un gran tipo de estos MCU llamados pequeños dispositivos de propósito general, ESP8266 o RaspberryPi3 o cualquier tipo de dispositivos de Internet de las cosas (IoT), que están conectados a Internet por medio de un nodo central para compartir su información. El objetivo principal de este artículo es diseñar una topología de red IoT descentralizada para conectar todos los MCU o nodos, basada en la construcción del fractal de Peano, es decir, sin usar uno nodo central, simplemente compartiendo algunos parámetros con dos nodos adyacentes, considerando que cualquier miembro de estos nodos conocen los parámetros del resto de estos dispositivos incluso si no son nodos adyacentes. Específicamente, con la red propuesta es posible acceder a toda la red IoT en tiempo real de una manera dinámica, ya que la topología de la red se puede adaptar y reconfigurar cuando se agrega un nuevo nodo utilizando herramientas de Inteligencia Artificial para su aplicación en una Ciudad Inteligente. Esta propuesta permite ahorrar energía, aumentando el tiempo de vida de la red IoT, cuando se conectan más pequeños dispositivos inalámbricos y se detectan parámetros.

Palabras clave. Sistemas embebidos, inteligencia de enjambre, curva de Peano, ESP8266, RaspberryPi3, red definida por software.

Optimizing a Software-Defined Network using the Peano Curve L-System

Abstract. At the present, millions of Microcontroller Units (MCU) are connected simultaneously in a digitally way to be our life more comfortable. 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

ISSN 2007-9737

such as temperature, humidity, CO_2, adjustment of the environmental lights. There is a huge kind of these MCU called small general purpose devices, ESP8266 or RaspberryPi3 or any kind of Internet of Things (IoT) devices, which are connected to internet by means of a central node for sharing their information. The main goal of this article is to design a decentralized IoT network topology in order to connect all the MCU or nodes, based on the fractal Peano, i.e., without using a central one, just sharing some parameters with two adjacent nodes, taking into account that any member of these nodes knows the parameters of the rest of these devices even if they are not adjacent nodes. Specifically, with the proposed network we can access to the entire IoT network in real time in a dynamic way since the topology of the network can be adapted and reconfigured when a new node is added using tools of Artificial Intelligence for its application in a Smart City. This proposal allows to save energy, increasing the time of life of the IoT network, when more wireless small-devices are connected and sensing parameters.

Keywords. Embedded systems, swarm sntelligence, Peano curve, ESP8266, RaspberryPi3, software-defined network.

1. Introducción

En años recientes, ha ido aumentado el interés, tanto académico como de la industria, por la interconectividad de las redes de Internet de las Cosas (IoT, por sus siglas en inglés). Un tema en boga es la difusión y el compartimiento de parámetros de manera eficiente en esta clase de redes.

Las redes IoT se consideran como una nueva generación de redes que combinan sensado o monitoreo de parámetros ambientales, como temperatura, humedad, CO_2, y ajuste de la intensidad de la luz, mediante la trasmisión y procesamiento de datos y a través de técnicas de comunicación inalámbrica o redes Inalámbricas de sensores [3, 5], proporcionando una mejor elección para la difusión y el compartimiento de parámetros entre los sensores. Varios sensores son ubicados en casas o para generar datos sobre los estados ambientales y físicos de dichos parámetros.

Después de recolectar, transmitir, compartir, procesar y analizar los datos, se presentan problemas estructurales, tales como agregar nuevos dispositivos o nodos a la red y comunicar estos parámetros de manera centralizada y exitosa, como es el caso de la configuración estrella, a fin de predecir y adaptar de la red y, por ende, actualizar las mediciones generadas para que un nuevo requerimiento sea atendido [12].

Asimismo, es importante resaltar que para el 2020, la IoT alcanzara su masa crítica. Proporcionando una gran cantidad de inteligencia en la nube a billones de dispositivos móviles y contando con sensores extremadamente accesibles y completamente etiquetados, se entregará una enorme cantidad de nuevos valores a casi cualquier ser humano. Los beneficios completos, en términos de salud, seguridad y conveniencia, serán enormes[16].

La perspectiva es que para el 2020 estarán cuatro millones de personas conectadas con un beneficio económico de cuatro billones de dólares porque más de 55 millones de aplicaciones serán descargadas desde las tiendas de aplicaciones; por lo tanto, que se necesitarán más de 25 millones de unidades de microcontroladores (UMC), generando más de 50 billones de gigabytes de datos dentro de la red global IoT.

En la Figura 2 se muestra la distribución de planta de una casa convencional en los que se pueden ubicar dispositivos IoT. Una red IoT contiene un Punto de Acceso WiFi (PAW) que conecta todos los dispositivos a internet o intranet, como el principal nodo. Si un UMC quiere compartir algunos parámetros con otro UMC, estos UMC deben comunicarse con el PAW. Aún si estos UMC están físicamente muy cerca uno del otro, sino están conectados al PAW, no será posible establecer comunicación ni compartir parámetros.

Dado lo anterior, en este artículo se propone un método efectivo para la difusión y el compartimiento de parámetros en Redes Inalámbricas de Sensores (RIS) con inteligencia de enjambre [6, 13, 14].

Para ello, se usaron una gran cantidad de sensores conectados en red para sensar y monitorear en tiempo real diferentes parámetros, a fin de hacer más eficiente una red IoT y, por ende, estar en condiciones de aplicar dicha red en una propuesta de ciudad inteligente.

ISSN 2007-9737

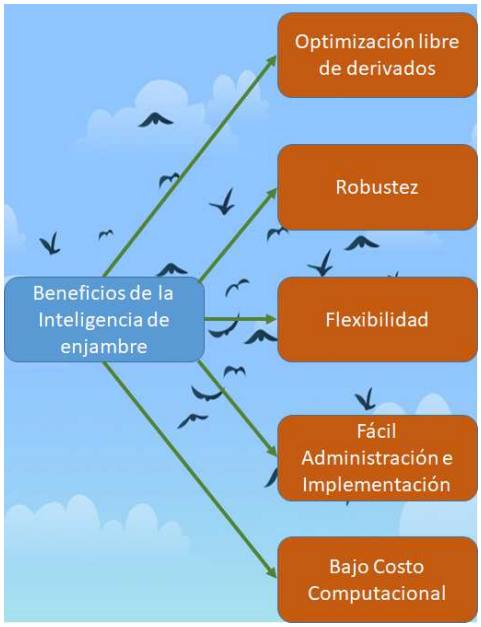


Fig. 1. Beneficios de la Inteligencia de Enjambre [1]

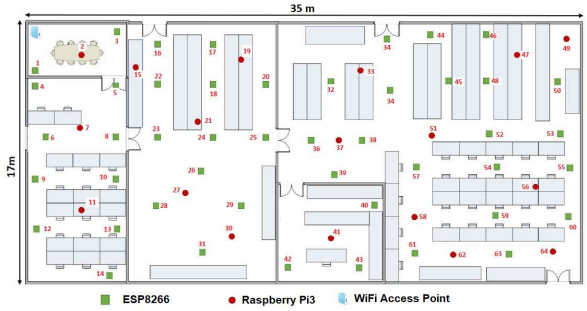


Fig. 2. Distribución de planta para ejemplificar el algoritmo propuesto

Las siguientes secciones de este artículo se organizan como sigue. En la Sección 2 se presenta la Descripción General. En la Sección 3 se ofrece una descripción detallada del Marco Teórico, describiendo los sistemas embebidos en la red IoT y la curva fractal de Peano. En la Sección 4 se describe el esquema principal propuesto. En la Sección 5 se presentan la simulación y la comparación para verificar que sea correcto y factible el esquema propuesto. Finalmente, en la Sección 6 se presentan las conclusiones a las que se llegan en este trabajo.

2. Descripción general de la propuesta

La propuesta en este artículo se basa en el principal trabajo realizado por Beni y Wang [1] en el que se introduce la expresión Inteligencia de Enjambre (IE) en su investigación sobre sistemas celulares de robots en 1993. El concepto de IE se aplica al trabajar con Inteligencia Artificial. La IE es una técnica computacional inteligente que se basa en el comportamiento colectivo de sistemas auto-organizados y descentralizados.

Un típico sistema IE se conforma de un grupo de agentes sencillos que interactúan localmente cada uno con los demás y con el medio ambiente que los rodea. Los UMC en un sistema de IE siguen reglas sencillas y actúan sin el control de entidades centralizadas. Sin embargo, las interacciones sociales entre tales UMC pueden generar enormes beneficios y frecuentemente conducir a un comportamiento global inteligente.

La Figura 1 muestra que algunos de los beneficios de la IE son: la optimización libre de derivados, la robustez, la flexibilidad, la facilidad de administración e implementación, lo que lleva a bajos costos computacionales y económicos.

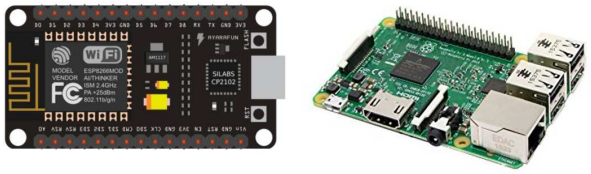
La IE toma toda la ventaja de el enjambre; por ende, la IE es capaz de proveer soluciones optimizadas que aseguren gran robustez, flexibilidad y bajo costo para problemas sofisticados a gran escala sin un ente de control centralizado. Por consiguiente, la presente propuesta se divide en tres principales partes, con base en la IE:

1. Selección del UMC inicial o principal.
2. Generación de la topología en una distribución de planta dada.
3. Inclusión de todos los UMC dentro de una red IoT.

Por tanto, en la Figura 2 se presenta una distribución de planta, la cual consta de un PAW y diversos ESP8266 y RaspberryPi3, a saber, 64 sistemas embebidos. Adicionalmente, en la primera parte el UMC más cercano al PAW es elegido como el principal UMC.

ISSN 2007-9737

Entonces, al medir el ancho de banda en megabits por segundo (Mbps), es posible predecir una topología para una distribución de planta dada; cuando se agrega un nuevo UMC, el algoritmo genera una nueva topología. Una vez generada la nueva topología, cada UMC o nodo de la red IoT se incluye mediante el fractal de Peano; entonces se mide la efectividad de la metodología para compartir parámetros a lo largo y ancho de la red, por lo que los algoritmos Wi-Fi Peer-to-Peer (P2P) y SoftAP se configuran.



(a) (b)

Fig. 3. Sistemas embebidos en la red IoT (a) Sistema embebido NodeMCU ESP8266 (UMC-ESP8266) y (b) Sistema embebido Raspberry Pi 3 (UMC-RPi3)

3. Marco teórico

Esta sección se subdivide en tres subsecciones:

- Sistemas embebidos en la red IoT propuesta,
- Wi-Fi Peer-to-Peer,
- La curva fractal de Peano.

3.1. Sistemas embebidos en la red IoT propuesta

El *sistema embebido NodeMCU ESP8266*, descrito en la Figura 3(a), es una tarjeta de desarrollo similar a la de Arduino, pero orientada especialmente a IoT. Dicha tarjeta se basa en el Sistema en Chip (SeC) ESP8266, un chip altamente integrado y diseñado para las necesidades de un mundo conectado. Además, esta tarjeta integra un procesador con la arquitectura de 32 bits (más poderosa que la del Arduino Due) y una conectividad de fidelidad inalámbrica o WiFi (wireless fidelity).

Para el desarrollo de aplicaciones se puede elegir entre los lenguajes Arduino y LUA. Al trabajar en un ambiente Arduino, se puede usar un lenguaje ya conocido y emplear un IDE simple de utilizar; asimismo, se puede hacer uso de toda la información sobre proyectos y librerías disponibles en la internet.

La comunidad usuaria de Arduino está muy activa y apoya plataformas como la ESP8266. El NodeMCU tiene preinstalado el firmwire (o sistema operativo interno de la tarjeta) que permite trabajar con el lenguaje interpretado LUA, enviando comandos a través del puerto serial (CP2102).

Las minitarjetas NodeMCU y Wemos D1 son las plataformas más usadas en los proyectos IoT. Dichas tarjetas no compiten con la Arduino, ya que éstas persiguen diferentes objetivos, aunque es posible programar NodeMCU a partir del IDE de Arduino. La tarjeta NodeMCU está especialmente diseñada para trabajar en un ambiente de desarrollo integrado. Esta tarjeta tiene un regulador de voltaje de corriente directa, el cual permite alimentar directamente la tarjeta desde el puerto USB. Los pins (o patillas) de entrada/salida trabajan a 3.3 volts. El chip CP2102 maneja la comunicación con puertos seriales USB. El NodeMCU cuenta con las siguientes especificaciones técnicas:

- Alimentación (USB): 5 voltios de corriente directa,
- Voltajes entrada / salida: 3.3 voltios de corriente directa,
- SeC: ESP8266 (módulo ESP-12),
- CPU: Tenselica Xtensa LX # (32 bits),
- Frecuencia de reloj: 80 MHz / 160 MHz,
- Memoria RAM para instrucciones: 32 Kb,
- Memoria RAM para datos: 96 Kb,
- Memoria Flash externa: 4 MB,
- Pins digitales GPIO: 17 (pueden ser configurados como PWM a 3.3 voltios),

- Pin analógico de corriente alterna / directa: 1 (0 – 1 voltio),
- UART: 2,
- Chip de puerto serial: CP2102,
- Certificación FCC,
- Antena en PCB,
- 802.11 b/g/n,
- WiFi directo (P2P), Soft-AP,
- Pila de protocolos TCP / IP integrada,
- PLLs, reguladores, DCXO y administración de potencia integrada,
- Potencia de salida de 19.5 dBm en modo 802.11 b,
- Fuga de corriente menor a 10 uA,
- STBC, 1 1 MIMO, 2 1 MIMO,
- Agregación A-MPDU y A-MSDU e intervalo de guardia de 0.4 msl,
- Paquetes de activación y trasmisión $< 2ms$,
- Consumo de energía en espera $< 1.0mW$ (DTIM3).

El *sistema embebido Raspberry Pi 3*, mostrado en la Figura 3(b), es la solución multiproceso más avanzada de esta marca hasta el momento. Debido a que la primera Raspberry, llamada Raspberry Pi Modelo B, se introdujo en 2012, más de ocho millones de unidades han sido vendidas, en aquel entonces con 256 MB de RAM. Las nuevas versiones incluyen mejoras en sus características, logrando la Raspberry Pi 3 Modelo B (UMC-RPi3).

A continuación, se mencionan algunas características y ventajas de la UMC-RPi3, por si se busca trabajar con una solución multiproceso. La solución multiproceso Raspberry Pi 3 incluye una serie de ventajas y novedades con respecto al modelo anterior, la Raspberry Pi 2 Modelo B.

Primeramente, la Raspberry Pi 3 tiene un procesador mucho más rápido y potente, el cual tiene un ARM Cortex con 4 núcleos, 1.2 GHz y 64 bits. Su ejecución es, al menos, 50 por ciento mayor que la versión anterior. Asimismo, esta nueva solución multiproceso cumple con uno de los mayores deseos de sus principales clientes: conectividad inalámbrica WiFi y Bluetooth. Los modelos anteriores podían conectarse solamente a través de cable Ethernet o mediante adaptadores inalámbricos USB, de tal manera que esta versión facilita muchísimo las tareas de conectividad. La UMC-RPi3 incluye un Bluetooth 4.1 y una tarjeta de red inalámbrica WiFi 802.11n.

Aunado a lo anterior, este nuevo modelo cambió sustancialmente de tamaño y sólo agrega modificaciones en la posición de las luces LED. Por otra parte, la UMC-RPi3 cuenta con las mismas características incluidas en la versión 2: 1 GB de RAM, cuatro puertos USB, puertos HDMI y Ethernet, micro SD, cámara CSI, entre otros; lo que permite su total compatibilidad con las versiones Cero y 2. La marca Raspberry recomienda usar dicha solución multiproceso en las escuelas para propósitos generales y para los proyectos integrados Pi Cero y el Modelo A +.

Cabe destacar que una de las grandes ventajas de la UMC-RPi3 es su precio. Muchas empresas, escuelas y centros de investigación utilizan soluciones multiproceso para crear una estructura más segura, pero que ofrezca todo tipo de aplicaciones: desde crear un miniservidor de correos electrónicos hasta una estación FM de baja gama.

Las especificaciones completas para la Raspberry Pi 3 incluyen:

- CPU: Quad-core 64-bit ARM Cortex A53, con reloj a1.2 GHz,
- GPU: 400 MHz VideoCore IV multimedia,
- Memoria: 1 GB LPDDR2-900 SDRAM (por ejemplo, 900 MHz),
- Puertos USB: 4,
- Salidas de video: HDMI, video compuesto (PAL y NTSC) vía 3.5 mm jack,
- Red: 10 / 100M bps Ethernet y tarjeta inalámbrica LAN 802.11n,

ISSN 2007-9737

– Periféricos: 17 GPIO, más funciones específicas, y bus HAT ID,

– Bluetooth: 4.1,

– Fuente de alimentación: 5 voltios vía Micro-USB o GPIO header (o cabezal),

– Dimensiones: 85.6 0mm x 56.5 mm,

– Peso: 45 g (1.6 oz).

3.2. Wi-Fi Peer-to-Peer

El *Wi-Fi Peer-to-Peer (P2P)* y el SoftAP son protocolos de comunicación que permiten una conexión directa entre los dispositivos WiFi donde sea, por lo que no se requiere un punto de acceso inalámbrico o PAW. Mediante la negociación, un dispositivo llega a ser el Dueño del Grupo y los demás los clientes. La conexión inicial está diseñada para ser rápida y fácil; su implementación no requiere adicionar hardware nuevo, permitiendo a los usuarios WiFi existentes agregar nuevos dispositivos a sus productos con sólo actualizar un software.

Por un lado, el protocolo WiFi Peer-to-Peer está diseñado para reemplazar el protocolo ad-hoc heredado para la interconexión de dispositivos WiFi. Las mejoras sobre el protocolo ad-hoc incluyen conexión más segura y lo último en seguridad, 802.11i. Este protocolo se diseñó para satisfacer la creciente necesidad de conexiones dinámicas entre grupos de dispositivos electrónicos. Esto les permite a los fabricantes de equipo original crear productos que puedan interoperar con los demás y con dispositivos de consumo comunes, tales como teléfonos inteligentes, tabletas, cámaras e impresoras. Como cualquier otro dispositivo WiFi, el rango alcanza hasta 200 m, haciendo una conexión apropiada aún con dispositivos que no se encuentren muy cercanos.

Por otro lado, el protocolo SoftAP ofrece a un dispositivo capacidades limitadas de punto de acceso WiFi. Cuando se combina este protocolo con el WiFi P2P, es posible que el dispositivo elegido como punto de acceso WiFi sea el Dueño del Grupo; sin esta combinación, este dispositivo podría participar sólo como un cliente más.

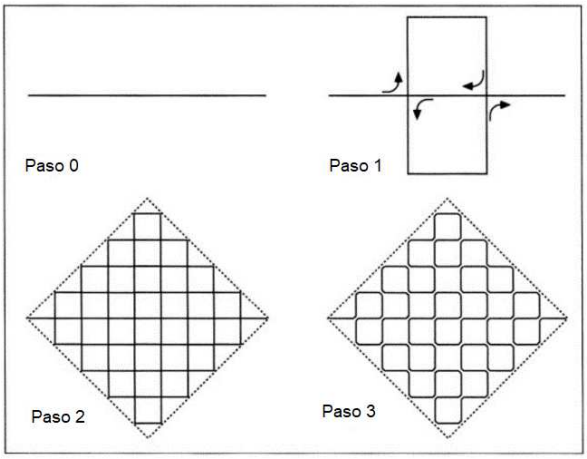


Fig. 4. Construcción de la curva de Peano: Construcción de una curva que llena el espacio que la contiene con iniciador y generador. En cada paso un segmento de línea se remplaza por 9 segmentos reducidos por un factor de 3

3.3. La curva Fractal de Peano

A Mandelbrot se le considera el padre de la geometría fractal. Sin embargo, muchos fractales y sus descripciones se remontan a las matemáticas clásicas y los matemáticos del pasado como Georg Cantor (1872), Guiseppe Peano (1890), David Hilbert (1891), Helge von Koch (1904), Waclaw Sierpinski (1916), Gaston Julia (1918) o Felix Hausdorff (1919), entre otros. Mandelbrot demostró que estos fractales matemáticos iniciales tienen muchas características en común con las formas encontradas en la naturaleza; de ahí el título de su libro en 1982: *La geometría fractal de la naturaleza* [8].

Mandelbrot dio un giro en la interpretación matemática ortodoxa e invirtió el valor de estos *monstruos matemáticos*. En 1890 Guiseppe Peano [10] (1858-1932) y en 1891 David Hilbert [4] (1862-1943) abordaron las llamadas curvas que *llenan* el plano que las contenga. La Figura 4 indica los primeros pasos de la construcción iterativa de la curva original de Peano.

Para construir la curva de Peano se inicia con un segmento de línea unitario, el iniciador, que se sustituye por una curva generadora, como se muestra en la Figura 4.

ISSN 2007-9737

Aparentemente, el generador tiene dos puntos de auto-intersección en la curva. Obsérvese que esta curva generadora ajusta muy bien a un cuadrado, el cual se muestra en líneas punteadas. Este es el cuadrado cuyos puntos serán alcanzados por la curva de Peano. Para el paso 2 se toma cada segmento de la línea recta del paso 1 y se remplaza apropiadamente por el generador; el factor de escalamiento es 3. Hay un total de 32 puntos que se auto-intersectan en la curva. Esto se repite, es decir, en cada paso, los segmentos de línea se reducen por un factor de 3.

Por tanto, en el paso k^{th}, un segmento de línea tiene longitud $\frac{9^k}{3^k} = \frac{3^{2k}}{3^k} = 3^k$, que es un número que decrece muy rápidamente. Ya que cada segmento de línea se remplaza por 9 segmentos de línea de longitud de $1/3$ de la longitud de los segmentos de línea previos, es posible calcular fácilmente la longitud de las curvas en cada paso. Se asume que la longitud del segmento de línea original es $k = 1$, constituyendo el iniciador, por lo que se obtiene en el paso 1: 3^1, y en el paso 2 $k = 1$, es decir, 3^2. Expresado como regla general, en cada paso de la construcción, resulta una curva que aumenta su longitud por un factor de 3. En el paso k, la longitud es 3^k.

En 1968 el biólogo Aristid Lindenmayer inventó una formalización para describir el crecimiento de una planta, que es también muy apropiada en implementaciones computacionales, conocida actualmente como sistema de reescritura paralela o sistema *L* [7, 15].

En un sistema L existen reglas de (re)producción que establecen los mecanismos a través de los cuales se da el crecimiento del sistema o fenómeno en estudio, así como un estado inicial del sistema o fenómeno llamado axioma. Las reglas de producción (o de reescritura) determinan qué tanto una cadena de entrada se transforma en una cadena de salida. Los sistemas L son máquinas de reescritura de cadenas que se caracterizan por el hecho de que las reglas de producción se aplican simultáneamente a todos los símbolos de la cadena de entrada. Lindenmayer intentó capturar, por ejemplo, la división celular en organismos multicelulares, en donde muchas divisiones pueden ocurrir al mismo tiempo.

Con el propósito de establecer el mecanismo de crecimiento para los fractales clásicos, se recomienda aplicar el enfoque de los sistemas L, resultando en cadenas de símbolos que deben interpretarse gráficamente. Esta interpretación es independiente de la generación de la cadena. A continuación se presenta una interfaz gráfica muy sencilla para cadenas de símbolos [11], la cual se basa en el concepto de gráficas de tortuga de Seymour Papert [9], especialmente apropiada para curvas en el plano. Con tal interpretación es posible formular las construcciones originales de los fractales clásicos de manera compacta y muy concisa, es decir, como un sistema L con solo algunas reglas de producción.

Imagínese una tortuga completamente entrenada para entender algunos comandos que se le dan en forma de símbolos (a partir de la lista de símbolos usados en el sistema L). Estos símbolos serán sólo letras comunes del alfabeto y algunos símbolos especiales como $+$ o $-$. A continuación se presenta el primer conjunto de instrucciones para la tortuga: F (moverse hacia adelante un paso de longitud l y dibujar una línea desde la vieja a la nueva posición), $+$ (girar a la izquierda, contrario a las manecillas del reloj, en un ángulo δ) y $-$ (girar a la derecha, en sentido a las manecillas del reloj, en un ángulo δ).

La Figura 5 muestra de nuevo la construcción de la curva de Peano. Su construcción comienza solo con un segmento de línea. Esta curva (que se sustituirá por simples segmentos de línea) puede codificarse de diversas maneras. Después de un paso hacia adelante, la tortuga gira a la izquierda o a la derecha, o puede seguir en línea recta.

Por ejemplo, después de girar a la derecha, debe haber dos giros a la izquierda con segmentos de línea entre los tres giros. Por tanto, la tortuga ha logrado llegar a un punto en donde la curva se encuentra con ella misma, y hay una elección, continuar en la misma dirección o dar un giro a la izquierda. En ambos casos la tortuga trazara el bucle superior del generador en sentido contrario a las manecillas del reloj o en dirección a las manecillas del reloj; y sólo entonces, habrá terminado el último segmento de línea al final de la curva.

ISSN 2007-9737

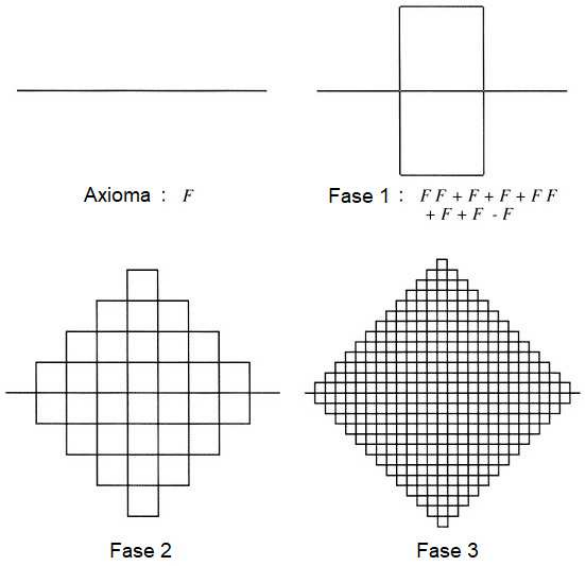


Fig. 5. La curva de Peano. Construcción de una curva que llena el espacio que la contiene: la curva de Peano. El axioma y la producción (parte superior); la segunda y la tercera etapas (parte inferior)

En términos de los comandos de la tortuga, estas dos alternativas pueden describirse como sigue (en cualquier caso, se elige $\delta = 90^o$):

- F–$F + F + F + F$–F–F–$F + F$,
- F–$F + F + FF + F + F + FF$.

Si la tortuga continua en la misma dirección en el punto de la primera decisión, se tiene:

- FF–F–F–FF–F–$F + F$, o
- $FF + F + F + FF + F + F$–F.

El sistema L completo puede trabajar con cualquiera de estas decisiones, por ejemplo, se puede establecer:

Axioma: F,
Reglas de producción:

$FFF + F + F + FF + F + F$–F,
$+ \longrightarrow +$,
$- \longrightarrow -$,
Parámetro: $\delta = 90^o$.

4. Difusión y compartimiento de parámetros en una red IoT

4.1. Selección del UMC inicial o principal

Nuestro algoritmo considera que el punto de acceso WiFi, PAW, está en una posición aleatoria dentro del cuarto, pero conectado a al menos un UMC en la red IoT. Al inicio, todos los UMC intentan conectarse al PAW; si éstos no establecen la conexión, la opción de WiFi directo se habilita. Los UMC que se conectan al PAW inhabilitan el modo WiFi directo, para que sean instruidos de otra manera. Así, el PAW se configura con el modo de difusión, enviando paquetes para medir la banda ancha de cada miembro o nodo en la red IoT.

Desde este momento empleamos un Identificador de Red UMC (IRU), el cual contiene el tipo de UMC, para UMC-ESP8266 o para UMC -RPi3, así como un byte objetivo para la identificación interna de la IoT. Por ejemplo, un IRU=UMC-ESP8266-i se refiere al ESP8266, el cual se identifica como el *ith* sistema embebido dentro del cuarto, sin importar el tipo o características de la UMC. Debido al material con el que están hechas las paredes del cuarto, la conexión se realiza sólo con algunos UMC; en cantidad, estos UMC deben ser menores o iguales que 3^k, como se muestra en la ecuación 1:

$$IoTe = \sum_{i=1}^{3^{2k}} UMC_i, \tag{1}$$

donde $IoTe$ es el número de sistemas embebidos en la red, i es el índice del Identificador de la Red UMC, IRU, y k es el nivel (o iteración) del fractal de Peano, definido por la ecuación 2:

$$n = \lceil \log_9 (IoTe) \rceil . \tag{2}$$

De acuerdo a lo anterior, es posible definir a $UMC_{i=1}$ como el UMC conectado más cercano con el PAW, ya que este alcanza la velocidad más rápida de conectividad de los enlaces $IoTe$. Por tanto, UMC_1 es indexado como el primer nodo en la red.

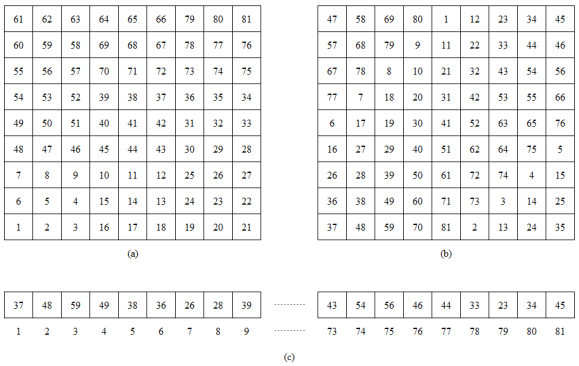


Fig. 6. (a) Matriz θ, (b) Matriz $\mathcal{P}$, y (c) Vector $\overrightarrow{\mathcal{P}}$

Algorithm 1: Función para generar la matriz de mapeo de Peano θ de tamaño $3^k \times 3^k$

Input: n
Output: θ
if $n = 1$ **then**
  $\theta = \begin{bmatrix} 7 & 8 & 9 \\ 6 & 5 & 4 \\ 1 & 2 & 3 \end{bmatrix}$
else
  $\beta =$ **Algoritmo 1** $(n-1)$
  $\lambda_1 = \begin{bmatrix} \beta & \overline{(\beta^T)} + 9^{n-1} & B + (2 \times 9^{n-1}) \end{bmatrix}$
  $\lambda_2 = \begin{bmatrix} \left(\overline{\lambda_1}\right)^T + (3 \times 9^{n-1}) \end{bmatrix}$
  $\lambda_3 = \begin{bmatrix} \left(\overline{\lambda_2}\right)^T + (3 \times 9^{n-1}) \end{bmatrix}$
  $\theta = \begin{bmatrix} \lambda_3 & \lambda_2 & \lambda_1 \end{bmatrix}$

4.2. Generación de la topología en una distribución de planta

Una vez identificado $UMC_{i=1}$ como el nodo principal en una red IoT, los sistemas embebidos $UMC_{i=1}$ contenidos en la red IoT habilitan el modo WiFi directo y completan una tabla con el ancho de banda de los nodos próximos a estos sistemas embebidos, a fin de generar una Lista de Nodos Alcanzables (LNA) específica; entonces, la red IoT propuesta puede generar 9^k LNA.

Aunado a lo anterior, cada LNA$_i$ contiene el ancho de banda de todos los UMC_i que deben conectarse, al menos, a otro UMC_i.

Las LNA están compartidas y $\sum_{i=1}^{9^k} UMC_i$ conoce a cualquier nodo en la red, a saber, cada nodo conoce la topología de la red.

Este segundo paso no tiene como paradigma la optimización ni la inteligencia. Por ende, es importante aplicar herramientas de la inteligencia de enjambre para mejorar estos resultados iniciales.

4.3. Indexando todos los UMC dentro de la red IoT

Con base a lo anterior, cada nodo tiene un índice inicial en la red, el cual al principio no está optimizado. Para optimizarlo, todos los elementos deben numerarse, según el fractal de Peano y la posición por donde cada nodo pasa.

Una indexación lineal se desarrolla para identificar el arreglo matricial UMC_i como un vector.

Se define el arreglo matricial de microcontroladores UMC como $\mathcal{P}$ y el vector resultante intercalado como $\overrightarrow{\mathcal{P}}$, siendo $3^k \times 3^k$ el tamaño de $\mathcal{P}$ y 9^k el tamaño de $\overrightarrow{\mathcal{P}}$, donde k es el nivel (o iteración) de la curva de Peano. El Algoritmo 1 genera una matriz de mapeo de Peano θ con un nivel k, expresando cada curva como nueve índices consecutivos.

El nivel k de θ se adquiere concatenando cuatro diferentes transformaciones θ en el nivel previo $n-1$. El Algoritmo 1 genera la matriz de mapeo de Peano θ, donde $\overline{\beta}$ se refiere a una rotación de 90 grados de β y β^T es la transpuesta algebraica lineal de β o $\overline{\beta}$. La Figura 6 se muestra un ejemplo de la matriz de mapeo θ en el nivel $k = 2$.

Por tanto, cada UMC_i en $\mathcal{P}_{(i,j)}$ se guarda y ordena en $\overrightarrow{\mathcal{P}}_{\theta_{(i,j)}}$, siendo $\theta_{(i,j)}$ el índice de ubicación de UMC_i en $\overrightarrow{\mathcal{P}}$. Asimismo, la Figura 6(b) muestra la posición dentro de una habitación y el IRM de cada UMC_i, a saber, la mejor forma de comunicar para UMC_i es la MCU_{i+1}, ofreciendo como resultado el incremento del ancho de banda.

De acuerdo a lo anterior, los nodos i son consecutivos y están comunicados al nodo $i+1$. En el caso de que en la topología actual no haya UMC_i en una determinada posición i, el índice que esté más cercano $i-1$ se alcanza y este UMC_i se considera como un nodo no significante (NNS); de otra manera, el nodo se considera nodo significativo (NS). Es importante resaltar que en el caso de que este nodo UMC_{i+1} sea NNS, el

Table 1. Ejemplos of identificador de Sensor o IS

Etiqueta	Parámetro
00H	Temperatura
01H	Humedad
02H	Sensores infrarrojos pasivos
03H	RFID
04H	Control de puerta
05H	Control de temperatura
06H	Bioritmia
07H	Biosensores

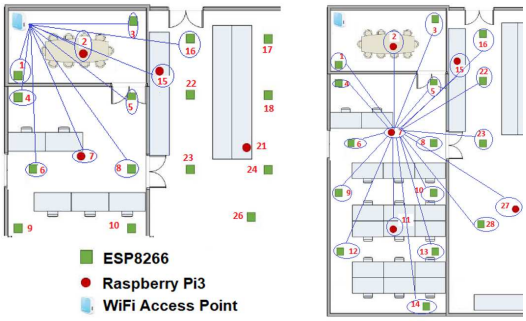


Fig. 7. Resultados experimentales: (a) Estimación de la velocidad de enlace de cada nodo en la red IoT y (b) Nodos alcanzables para UMC-RPi3-7

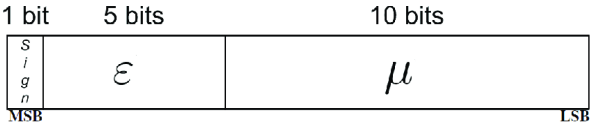


Fig. 8. Estructura del Identificador del Sensor o IS

nodo UMC_i alcanza a un NS, incrementando i hasta alcanzarlo.

A fin de conocer la significancia de la red IoT, proponemos una Cadena de Significancia de la Red (CSR) con 9^k bits; 0 para NSN y 1 para NS. La CSR es propagada a $\sum_{i=1}^{9^k} UMC_i$ con pocos nanosegundos.

Cada UMC_i mide o evalúa ciertos parámetros como la temperatura, humedad, CO_2 y ajuste de las luces ambientales. Para diferenciar los parámetros, proponemos un marcador de 1 byte que indique específicamente cuál parámetro que un sensor mida o identificador del sensor (IS) esté en unidades de grados Celsius o voltaje, por ejemplo. La Tabla 1 muestra algunos ejemplos de parámetros que un UMC_i puede medir, así como un IS de 1 byte que indique hasta 256 parámetros.

Cada valor del parámetro que mida un UMC_i se representa por un identificador de 2 bytes de longitud SM, el cual se divide en tres partes: signo, exponente ε y mantisa μ (Figura 8).

El bit más significativo del marcador es el signo del parámetro, 0 para positivo y 1 para negativo. Los últimos diez bits significativos se emplean para la ubicación de μ, el cual está definido por [2] como:

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

La ecuación (4) expresa cómo ε se obtiene, el cual se almacena en los cinco bits restantes del marcador SM:

$$\varepsilon = R - \lceil \log_2 |Parámetro| \rceil , \qquad (4)$$

donde R es el número de bits usados para representar el valor más alto de cierto parámetro, definido como:

$$R = \lceil \log_2 \left[\text{máx} \left\{ Parámetro \right\} \right] \rceil \ . \qquad (5)$$

5. Resultados experimentales

En esta sección extendemos lo expuesto en la sección anterior. Asimismo, utilizamos la distribución de planta de la Figura 2 y damos seguimiento a la siguiente cantidad de UMC:

– 46 UMC-ESP8266,

– 18 UMC-RPi3.

Es importante notar que usamos un PAW para conectar la red IoT a la internet como el nodo inicial con las siguientes características: router inalámbrico Tenda N300, repetidor inalámbrico de señal, IEEE del estándar 802.11 b/g/n RJ45 con 4 puertos a 300 Mbps. Con base en la Figura 2, en la Figura 7 (a) se aprecia que el PAW se ubica en la esquina superior derecha.

ISSN 2007-9737

Table 2. Ancho de banda (Mbps) de los primeros diez UMC dentro de la red IoT

Identificador de Red UMC	Distancia Lineal (m)	Distancia Real (m)	Ancho de banda (Mbps)
UMC-ESP8266-1	3	3	198.75
UMC-ESP8266-4	4	12	116.15
UMC-ESP8266-6	7.5	13	106.93
UMC-RPi3-7	7	11	127.65
UMC-ESP8266-8	9	11.5	120.52
UMC-ESP8266-5	6.5	6.5	166.98
UMC-RPi3-2	3.5	3.5	193.50
UMC-RPi3-15	8	16	79.64
UMC-ESP8266-16	8.5	19	52.94
UMC-ESP8266-3	6	6	170.45

Al inicio todos los UMC intentan conectarse al PAW; si éstos no establecen la conexión, la opción de WiFi directo es habilitada. Los UMC que se conectan al PAW deshabilitan el modo de WiFi directo hasta que reciben otra instrucción. Entonces, el PAW se configura en el modo de difusión, enviando paquetes para medir el ancho de banda de cada miembro o nodo en la red IoT. A partir de este momento, usamos el Identificador de Red UMC, IRU, el cual contiene el tipo de UMC, ya sea del UMC-ESP8266 o del UMC-RPi3, así como el byte objetivo para la identificación dentro de la intranet IoT.

Por ejemplo, un IRU=UMC-ESP8266 se refiere a un ESP8266, el cual fue identificado como el octavo sistema embebido dentro del cuarto, sin importar el tipo o características del UMC. Debido al tipo de material del que están hechas las paredes, la conexión se realiza sólo para los diecisiete UMC, cuyas velocidades se describen en la Tabla 2. Dicha tabla sólo muestra los primeros diez UMC conectados al PAW. Es por ello que podemos definir el UMC-ESP8266-1 como el UMC más cercano que está conectado con el PAW, ya que este UMC alcanza una velocidad de enlace de 198.75 Mbps y el UMC más lejano es el UMC-ESP8266-16 con una velocidad de conexión de 52.94 Mbps. El UMC-ESP8266-1 está indexado como el primer nodo en la red.

Una vez que el UMC-ESP8266-1 es identificado como el nodo principal o Nodo 1 en la red IoT, todos los UMC, incluido el UMC-ESP8266-1, habilitan el modo WiFi directo y completan la tabla con el ancho de banda de los nodos próximos a ellos, a fin de generar la Lista de Nodos Alcanzables, LNA. Cada UMC genera una LNA particular; por ejemplo, en la Figura 7 (b) se visualizan los identificadores de red UMC, IRU, a los que el nodo UMC-RPi3 puede conectarse.

Table 3. Lista de los Nodos Alcanzables LRN_7, generados por el UMC-RPi3-7

IRU	Distancia Lineal (m)	Distancia Real (m)	Ancho de banda (Mbps).
UMC-ESP8266-1	5.50	10.40	129.99
UMC-ESP8266-4	4.50	4.50	183.19
UMC-ESP8266-6	2.50	2.50	201.23
UMC-ESP8266-9	5.00	5.00	178.68
UMC-ESP8266-12	8.00	8.00	151.63
UMC-RPi3-11	6.00	6.00	169.67
UMC-ESP8266-14	10.50	10.50	129.09
UMC-ESP8266-13	7.50	7.50	156.14
UMC-ESP8266-10	4.50	4.50	183.19
UMC-ESP8266-28	8.20	10.40	129.99
UMC-RPi3-27	9.30	9.70	136.30
UMC-ESP8266-8	2.80	2.80	198.52
UMC-ESP8266-23	6.00	6.00	169.67
UMC-ESP8266-22	7.20	9.80	135.40
UMC-ESP8266-16	8.40	13.50	102.04
UMC-RPi3-15	6.30	11.25	122.32
UMC-ESP8266-5	4.45	4.45	183.64
UMC-ESP8266-3	7.25	7.25	158.39
UMC-RPi3-2	5.30	8.75	144.87

Según la Figura 2, nuestro ejemplo genera 64 LNA. Además, en la Tabla 3 se muestra el LNA-7, es decir, el ancho de banda de diecinueve UMC, el cual establece conexión con el UMC-RPi3-7. Como se observa, los UMC-ESP8266-6 y UMC-ESP8266-8 alcanzan los mayores anchos de banda para el Nodo 7, con 201.23 Mbps y 198.52 Mbps, respectivamente.

Asimismo, la Figura 7(b) muestra la posición dentro del cuarto y el IRU de cada UMC_i, a saber, la mejor manera de comunicación para UMC_{64} es el UMC_9, aumentando el ancho de banda, y así sucesivamente. De igual forma, la Figura 9 muestra una interpretación visual de la generación de una indexación lineal mediante el fractal de Peano en $k = 2$.

En este trabajo ejecutamos miles de requerimientos o solicitudes a partir de los nodos más lejanos, es decir, desde UMC_{64} hasta UMC_1 y desde UMC_8 hasta UMC_{57}. En promedio, el enlace $UMC_{64} \leftrightarrow UMC_1$ obtiene un ancho de banda de 192.85 Mbps, y para el caso de $UMC_8 \leftrightarrow UMC_{57}$ se obtiene un ancho de banda de 190.21 Mbps. En ambos casos se habla de una

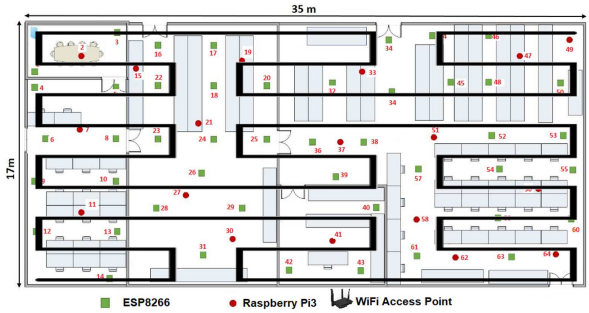


Fig. 9. Distribución de UMC en una planta convencional utilizando la curva fractal de Peano

topología central, en la que no es posible conectar $UMC_{64} \leftrightarrow UMC_1$ ni $UMC_8 \leftrightarrow UMC_{57}$.

6. Conclusiones

Reducir el consumo de energía es un objetivo primordial en el diseño de cualquier sistema de comunicación para Redes Inalámbricas de Sensores. La mayor parte de esta energía puede ahorrarse agregando más nodos o dispositivos inalámbricos a estas redes, ya que la mayoría de la información obtenida es redundante debido a la ubicación geográfica de los sensores. Por tanto, el esquema eficiente de difusión considera el compartir parámetros que hayan sido propuestos. Sin embargo, esto continúa implicando altos costos en la comunicación, lo que dificulta el compartimiento de los parámetros elegidos.

Es por ello que en este trabajo se ha propuesto un algoritmo ligero de difusión en redes inalámbricas de sensores. A partir de un análisis realizado, confirmamos que el esquema de reconfiguración de una red IoT genera un mayor tiempo de vida de la red como información más eficiente del parámetro a sensar, con respecto a los esquemas o topologías tradicionales.

Además, se determinaron algunos parámetros conectados a un UMC_i particular de toda la red IoT en microsegundos, lo que es congruente con la definición de tiempo real. Asimismo, el algoritmo propuesto se apoya en la Inteligencia de Enjambre, ya que cada UMC_i aprende del resto de los miembros o nodos de la red IoT y sabe lo que sucede, aún si no está conectado a a otro nodo. Para el caso práctico presentado en este trabajo, los enlaces más lejanos entre MCU's obtiene un ancho de banda promedio efectivo entre 190.21 Mbps y 192.85 Mbps, cuando en una topología tradicional tipo estrella no tendrían ni siquiera comunicación o enlace entre ellos.

Agradecimientos

El presente artículo ha sido apoyado por el Instituto Politécnico Nacional, a través de los proyectos de investigación SIP20190046 y SIP20195208, y por el Consejo Nacional de Ciencia y Tecnología de México. Asimismo, los autores agradecen a Juan Alfredo Durand Rivera y a Pedro Arrechea Alfaro por su apoyo técnico y adminsitrativo, respectivamente, para la realización del presente trabajo.

Referencias

1. **Beni, G., Wang, J. (1993).** Swarm intelligence in cellular robotic systems. Robots Biological Systems: Towards a New Bionics, Vol. Springer, pp. 703–712.
2. **Boliek, M., Christopoulos, C., Majani, E. (2000).** Information technology: JPEG2000 Image Coding System. ISO/IEC JTC1/SC29 WG1, JPEG 2000, JPEG 2000 Part I final committee draft version 1.0 edition.
3. **Hassan, T., Aslam, S., Jang, J. W. (2018).** Fully automated multi-resolution channels and multithreaded spectrum allocation protocol for IoT based sensor nets. IEEE Access, Vol. 6, pp. 22545–22556.
4. **Hilbert, D. (1891).** Über die stetige Abbildung einer Linie auf ein Flächenstück. Mathematische Annalen, Vol. 38, No. 3, pp. 459–460.
5. **Jo, O., Kim, Y. K., Kim, J. (2018).** Internet of things for smart railway: Feasibility and applications. IEEE Internet of Things Journal, Vol. 5, No. 2, pp. 482–490.
6. **Li, T., Yuan, J., Torlak, M. (2018).** Network throughput optimization for random access narrowband cognitive radio internet of things (NB-CR-IoT). IEEE Internet of Things Journal, Vol. 5, No. 3, pp. 1436–1448.

7. **Lindenmayer, A. (1968).** Mathematical models for cellular interaction in development: Parts I and II. Journal of Theoretical Biology, Vol. 18.
8. **Mandelbrot, B. (1997).** La geometría fractal de la naturaleza. Libros para pensar la ciencia. Tusquets Editores.
9. **Papert, S. (1980).** Mindstorms: Children, Computers, and Powerful Ideas. Basic Books, Inc., New York, NY, USA.
10. **Peano, G. (1890).** Sur une courbe, qui remplit toute une aire plane. Mathematische Annalen, Vol. 36, No. 1, pp. 157–160.
11. **Prusinkiewicz, P., Hanan, J. (1989).** Lindenmayer Systems, Fractals and Plants. Springer-Verlag, Berlin, Heidelberg.
12. **Sandoval, R. M., Garcia-Sanchez, A. J., Garcia-Haro, J. (2018).** Improving RSSI-based path-loss models accuracy for critical infrastructures: A smart grid substation case-study. IEEE Transactions on Industrial Informatics, Vol. 14, No. 5, pp. 2230–2240.
13. **Sharma, P. K., Chen, M. Y., Park, J. H. (2018).** A software defined fog node based distributed blockchain cloud architecture for IoT. IEEE Access, Vol. 6, pp. 115–124.
14. **Taghizadeh, S., Bobarshad, H., Elbiaze, H. (2018).** CLRPL: Context-aware and load balancing RPL for IoT networks under heavy and highly dynamic load. IEEE Access, Vol. 6, pp. 23277–23291.
15. **Thompson, D. W. (1992).** On Growth and Form. Cambridge University Press.
16. **Zhang, C., Ge, J., Pan, M., Gong, F., Men, J. (2018).** One stone two birds: A joint thing and relay selection for diverse IoT networks. IEEE Transactions on Vehicular Technology, Vol. 67, No. 6, pp. 5424–5434.

Article received on 12/12/2019; accepted on 16/01/2020.
Corresponding author is Jesús Jaime Moreno Escobar.