An Experimental Approach to the Consensus Routing Algorithm Applied to Temporal Networks

M. Arellano-Vázquez, *Member, IEEE*, H. Benítez-Pérez, *Member, IEEE*

***Abstract*—This article presents the implementation in real devices of the consensus-based routing algorithm (CRA). The CRA has the advantage of basing the route calculation on an availability function, considering the local conditions of a delimited neighborhood. This attribute allows the best candidate to be chosen to establish a connection within the temporary network. Previously, simulations of the operation of the presented algorithm have been shown, this article shows its performance in a physical environment, using open hardware. In this way, real measures of communication latency between devices and task scheduling are considered. The characteristics as mentioned earlier make the CRA a good option to apply in temporal networks since they are present for brief moments, so they do not require the route to be two-way.**

***Index Terms*—Routing algorithms, mobile distributed systems, complex systems, temporal networks, consensus.**

I. Introducción

Muchos sistemas como: las redes de comunicaciones, el metabolismo, las conexiones neurales y el contagio de enfermedades virales, pueden representarse como grafos. Además de los procesos biológicos, la interacción social y las grandes tecnologías como las redes de teléfonos móviles, en particular, los aspectos de tiempo y la disponibilidad de la red, son susceptibles a ser estudiados como sistemas complejos.

El modelado de estos sistemas dinámicos dice mucho acerca del comportamiento del sistema sin tener que profundizar en detalles de la dinámica completa. Estos sistemas pueden ser representados utilizando un conjunto de grafos que muestran un cambio de configuración en el tiempo, por lo que pueden considerarse redes temporales. Las redes temporales son aquellas en las que las interacciones entre un conjunto de vértices cambian con el tiempo. Una red temporal (RT) puede describirse como un conjunto de dispositivos capaces de cálculo y comunicación, que interactúan entre sí durante períodos de tiempo variables, es decir, dado que los nodos son móviles, es posible la incorporación y desconexión de nodos que no formaban parte de la topología de red inicial.

Una de las diferencias fundamentales entre las redes temporales y las redes estáticas es que las primeras no son transitivas y en las segundas si existe el camino de regreso. Incluso, si el vértice A está relacionado con B y B está relacionado con C, podría ser que A no esté relacionado con C. Esto implica que es posible que un par de nodos se pueden conectar a través de una red de contactos en una determinada ventana de tiempo. La latencia no es lo más importante, ya que al igual que los grafos regulares, las redes temporales se pueden desconectar [1]; esto es más común en las redes temporales que en las estáticas, ya que las rutas que unen los vértices deben recorrerse en el orden de los contactos [2] y [3].

Establecer una ruta entre un par de nodos en una RT no es una cuestión trivial, los algoritmos clásicos eligen la ruta como una trayectoria estática costosa (en tiempo de procesamiento). El hecho de que el vértice A esté conectado a B a través de una ruta durante un instante, no garantiza que la conexión exista en ningún momento posterior. Una estrategia es representar la ruta con un grafo que describe la configuración de la red en un momento específico.

En este trabajo se propone utilizar el algoritmo CRA (Consensus Routing Algorithm) [4] para trazar una ruta temporal entre dos nodos dentro de una red temporal en un ambiente experimental.

II. Trabajo Relacionado

Los algoritmos de consenso son ampliamente usados en áreas como la seguridad [5] y comunicación, por ejemplo, algunos como el propuesto en [6], el cual forma parte de protocolos de encaminamiento en redes, en este trabajo los autores separan el mecanismo de envío de mensajes en dos modos lógicos; un modo estable que asegura que la ruta sea seleccionada solo después de que se alcanzó un acuerdo de un estado, que consiste en una vista global; un modo transitorio que garantiza una alta disponibilidad cuando un paquete encuentra un enrutador que no tiene una ruta estable, debido a una falla en el enlace o porque los algoritmos todavía están calculando una ruta estable. El algoritmo de consenso se utiliza para calcular un estado global del sistema, a partir de instantáneas (estado capturado en un instante t) distribuidas que envía cada sistema autónomo. Los algoritmos de consenso en sistemas distribuidos se han utilizado con éxito para problemas en los que un grupo de agentes debe calcular una función vectorial lógica de manera cooperativa. Esta función devuelve un conjunto de decisiones dependiendo de un conjunto de eventos de entrada. Los agentes tienen acceso parcial a eventos entrantes, realizan cálculos simples y tienen capacidad de realizar comunicación broadcast, también pueden ser afectados por fallas. El problema involucra la difusión confiable de la información a través de la red, que se puede lograr mediante el uso de técnicas basadas en Robust Flooding [7] .

En [8] se propone un algoritmo de consenso para redes oportunistas, en este cada nodo lleva mensajes y se propaga

M. Arellano-Vázquez, Consejo Nacional de Ciencia y Tecnología y Centro de Investigación e Innovación en Tecnologías de la Información y Comunicación, Aguascalientes, México. e-mail: marellano@conacyt.mx

H. Benítez-Pérez, Universidad Nacional Autónoma de México, Ciudad de México, México, e-mail: hector.benitez@iimas.unam.mx

a través de la red, utilizando el encaminamiento epidémico [9]. La desconexión de un nodo no se considera un error, se considera un comportamiento esperado; se espera alcanzar la vecindad inmediata del nodo destino, el algoritmo utiliza un vector de intercambio de información que está dentro de cada rango de otro nodo y este requiere una vista global de la red. Las redes oportunistas tienen similitudes con RT, uno de los algoritmos de encaminamiento utilizados en dichas redes es el encaminamiento epidémico [9]. El encaminamiento epidémico utiliza la movilidad de los nodos para propagar mensajes a través de la red. Sin embargo, la cantidad de mensajes distribuidos y el mecanismo de intercambio de información consiste en que cada vez que un nodo encuentra a otro, establece un intercambio de vectores, estos se comparan y los mensajes se actualizan, este procedimiento es costoso cuando la red tiene una gran cantidad de nodos. Por lo tanto, este protocolo maximiza la tasa de entrega de mensajes y minimiza la entrega de latencia, pero es costoso en términos de utilización de ancho de banda y procesamiento de cada nodo. En [10] se propone un esquema de encaminamiento que explota la periodicidad temporal de los contactos nodales para determinanda selección de nodos de retransmisión en Mobile Opportunistic Networks, este esquema se basa en que los movimientos de los nodos puede ser predecible.

Por otro lado, el algoritmo en [11] propone elegir el algoritmo de encaminamiento dependiendo de las condiciones particulares de la red; las opciones disponibles son OLSR [12], DSR [13] y AODV [14], también utiliza un algoritmo de votación, esto asigna un peso a cada nodo en función del número de nodos vecinos que han elegido tener el voto mayoritario.

Los algoritmos de encaminamiento han sido ampliamente estudiados desde hace décadas, sin embargo, siguen siendo actuales, ya que con el acelerado desarrollo tecnológico están surgiendo nuevos escenarios que necesitan la creación de nuevos algoritmos, como es el caso de los algoritmos de encaminamiento en centros de datos [15] y [16]. La inteligencia artificial desde hace algunos años ha comenzado a plantear algoritmos basados en deep learning, en particular usando redes neuronales [17]. Por ejemplo, el algoritmo planteado en [18] aprende de sus errores y predice patrones en una topología conocida, por lo tanto, reporta una notable superioridad en perdida de paquetes y evitando congestionamientos al compararse con algoritmos tradicionales como OSPF, Intermediate System to Intermediate System (IS-IS) y Routing Information Protocol (RIP), sin embargo, no reporta resultados ni comparaciones en casos de topologías dinámicas. En [19] se muestra un enfoque cooperativo y distribuido, usando como base la red neuronal Graph Neural Networks, en este trabajo se menciona mejoría en comparación con algoritmos tradicionales como OSPF y RIP. Existen protocolos como RPL [20] y LOADng [21] que han innovado el área de algoritmos de enrutamiento, mejorando los protocolos existentes, en ambos casos se hace uso del flooding y un esquema centralizado, sin embargo, se han optimizado de manera que a pesar de realizar continuas solicitudes de broadcast no se presente una sobrecarga en los canales de comunicación, estos algoritmos han mostrado grandes ventajas sobre sus antecesores.

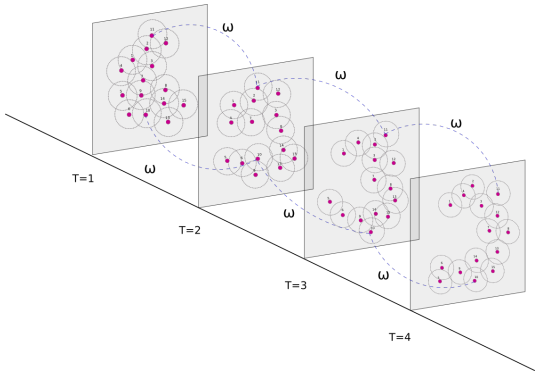

Fig. 1. Representación de una RT como grafos en transición.

III. ANTECEDENTES

Para la mejor comprensión del trabajo, a continuación se definen los conceptos principales de redes temporales. Las redes temporales son aquellas en las que las interacciones entre un conjunto de sus unidades elementales cambian en el tiempo, pueden ser modeladas usando grafos variables. Un sistema de red que evoluciona en el tiempo consiste en un conjunto $\mathcal{C}$ de contactos registrados a través de un conjunto de nodos $\mathcal{C} = \{1, 2, \ldots, N\}$ durante un intervalo de observación $[0, T]$. Un contacto entre dos nodos $i, j \in \mathcal{N}$ es representado por la $4-$tupla $c = (i, j, t, \delta t)$, dónde $0 \leq t \leq T$ es en el tiempo en que el contacto empezó, con una duración de δt [22].

Una red temporal con N nodos y de longitud T se define como una secuencia de r instantáneas de tiempo igual al tamaño $T_w = \frac{T}{r}$, como se muestra en la Fig. 1 el conjunto de estados que describen la evolución de la red durante un tiempo T. Un conjunto de datos de una red temporal consiste en una lista de contactos, cada contacto se define como las identidades de dos nodos que interactúan (i, j), el tiempo inicial t del contacto y se define como δt la duración del contacto. El número de contactos entre un par de nodos (i, j) que ocurre en la t-ésima instantánea, i.e. entre el tiempo $(t-1)T_w$ y tT_w, dónde $t = 1, \ldots, r$ se denota como $w_{ij}(t)$. La matriz de adyacencia $N \times N$ al tiempo t está dado por $w(t) = w_{ij}(t)$, esta matriz es la que describe el estado instantáneo del sistema [23].

La ruta de contactos es el conjunto de N vértices V, junto con un conjunto de M aristas dirigidas E, tal que el par (i, j) de vértices pertenece a E si y solo si existe un tiempo t tal que $(i, j, t) \in C$. Sea k el grado del vértice, como los nodos son dirigidos es posible distinguir entre grado entrante y saliente. Se denota l como el número de contactos a lo largo de una arista en particular. Se llama una *ruta de tiempo respetuoso* a una lista de contactos de tiempos crecientes. Ningún otro conjunto de contactos, aparte de rutas de tiempo respetuoso, puede transferir información de un vértice a otro en una secuencia de contacto [24].

El objetivo de este trabajo es implementar el CRA en una red temporal de comunicación dentro de un entorno experimental. El CRA [4] toma en cuenta las características de las redes temporales, como el contacto de tiempo promedio

entre nodos (si el tiempo no es suficiente para transmitir un mensaje, entonces el camino se rompe), la disponibilidad de la red, la movilidad y la distancia entre nodos. El CRA considera las condiciones locales, esta característica permite a los nodos decidir una ruta en función de tales condiciones locales limitados; para evaluar las condiciones locales utiliza un algoritmo de consenso basado en votaciones, lo que permite decidir qué nodo vecino transmitirá el mensaje y por lo tanto, seguir construyendo una ruta hacia al nodo destino.

El CRA calcula una ruta para una topología dinámica, estableciendo consensos sucesivos entre los nodos. Durante la evolución de la red temporal es posible observar la formación de grupos (clusters). En estos grupos, cada nodo obtiene información de enrutamiento de sus vecinos mientras tienen una conexión activa, este comportamiento proporciona resistencia a los cambios a nivel local. Es decir, cualquier cambio se mantiene localmente dentro de la RT. Cuando un nodo móvil que se mueve, conecta o desconecta, esto no afecta el estado general de la RT, sino que solo afecta al grupo de nodos involucrados. Por lo tanto, no es necesario recalcular la ruta, solo la sección que involucra el grupo de nodos afectados por el cambio. Para calcular el costo de enviar un paquete entre dos nodos, se define una métrica. Esta métrica se basa en el tiempo requerido para entregar un paquete utilizando una ruta particular. Normalmente, cada algoritmo de enrutamiento define métricas que asignan un valor a una ruta en función de un factor como el conteo de saltos, el ancho de banda, la confiabilidad del enlace, etc [25]. El CRA define su propia métrica, basándose en una función de disponibilidad que se describe a continuación.

IV. Materiales y Métodos

El CRA [4] utiliza la salida de una función de disponibilidad como valor de entrada (valor entre 0 y 1). La función de disponibilidad recibe como valores de entrada la información local: la carga en el enlace entre un par de nodos y el tiempo de contacto promedio de cada nodo.

A. Función de Disponibilidad

Sea $node$ una estructura de datos definida por cada nodo, esta estructura incluye los siguientes campos: ID único, posición en el eje x, posición en el eje y, β es el tiempo de contacto promedio de un nodo, λ la carga de cada enlace entre todos los pares de nodos conectados directamente. Sea k el número total de pasos de la ruta calculada, l el vecindario, la magnitud de este depende del número de saltos definidos. J_i es el conjunto de nodos que pertenecen al vecindario especificado por el nodo (radial) i, ρ radio, este depende del alcance de la antena del nodo. El conjunto J_i se puede definir como en la ecuación 1.

$$J_i = \{j : |x_i - x_j| \wedge |y_i - y_j| \leq \rho; j = 1, \cdots, n\}; \quad i = 1, \cdots, n. \tag{1}$$

Sea $|J_i| = m$ el número de entradas para el CRA por cada ronda. Sea $f^k_{i,J_i}(\rho)|i = 1, \ldots, n$ sean los valores resultantes de la función de disponibilidad [4].

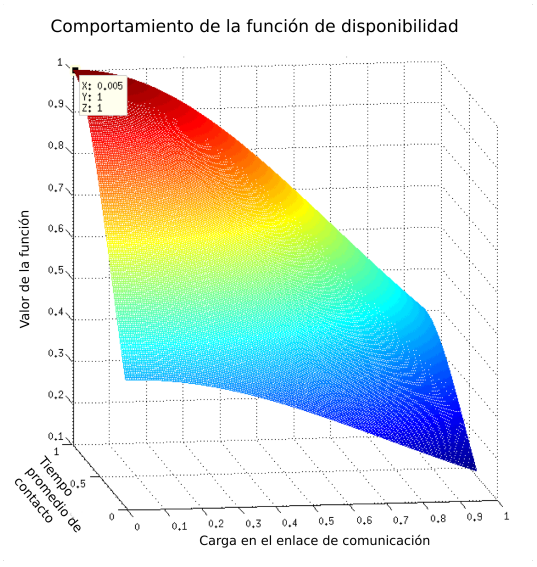

Fig. 2. Comportamiento de la función de disponibilidad.

La ecuación (2) describe dicha función, la cual se calcula para cada nodo de la RT.

$$f^k_{(i,J_i)}(\rho) = e^{-\left(\frac{(\lambda(J_i^h)-\Gamma)^2}{\sigma_1^2} + \frac{hops^2}{\sigma_2^2} + \frac{(\tau-\beta(J_i^h))^2}{\sigma_3^2}\right)}. \tag{2}$$

Donde $i = 1, \ldots, n; h = 1, \cdots, m$.

Esta función toma en cuenta el ancho de banda disponible por enlace en el instante de la solicitud (τ), el tiempo promedio de contacto (Γ), y el número de saltos entre un par de nodos en particular ($hops$). El valor resultante de esta función es un número real en el intervalo cerrado [0,1], el cuál indica que tan disponible está ese enlace para formar parte de la ruta, en la Fig. 2 se muestra el comportamiento de la función de disponibilidad. Cabe destacar que $\lambda(J_i^h)$ está dada en μs, mientras que $\beta(J_i^h)$ está dada en $kbps$, para evitar inconsistencias, ambos parámetros se escalan a 10^1. Sea $\sigma_i|i = 1, 2, 3$ las desviaciones estándar de λ, τ y $hops$ respectivamente.

B. Consensus Routing Algorithm (CRA), Versión Móvil

El algoritmo se basa en un árbol de búsqueda, el cual evalúa cada rama para elegir al mejor candidato. A medida que crece el número de nodos, el vecindario l crece, por lo tanto, el árbol de búsqueda también crece en profundidad a medida que aumenta el número de vecinos por nodo. El algoritmo se describe en detalle en [4] y en todas sus versiones, en este trabajo el algoritmo se implementa en dispositivos reales y se utilizó como base la versión móvil. A continuación se describe brevemente.

El sistema distribuido móvil (SDM en adelante) con n nodos, utiliza una estructura de datos por cada nodo, la cual contiene:

1) Identificador del nodo (ID).
2) La ubicación espacial en la malla (coordenadas x,y).
3) Tiempo disponible en su planificador de tareas (t).

4) Lista de sus vecinos (conjunto J_i) en un rango de transmisión ρ.
5) Carga entre cada uno de sus enlaces con sus vecinos (ca).
6) Una bandera que indica si el nodo es móvil (flag).

Habiendo definido la estructura de cada nodo, el algoritmo de encaminamiento por consenso se describe a continuación:

1) Sea J_i es el conjunto de nodos pertenecientes al vecindario del nodo i, en un vecindario radial ρ, el conjunto J_i se define como (1). Sean los valores de la función de disponibilidad de los enlaces, calculada por la ecuación 2 las entradas del algoritmo. Sea y_i^k la salida del consenso.
2) El nodo $i = n_s$ comienza la búsqueda del nodo destino n_d, primero busca entre sus vecinos directos. En forma paralela, mientras i realiza el consenso (que se describe en los siguientes pasos), todos los elementos de J_i buscan a n_d en sus vecinos inmediatos, en caso de no encontrarlo comienzan el consenso con sus propios vecinos directos. Si no se encuentra el nodo destino n_d en ese conjunto, entonces se calcula la función de disponibilidad de los vecinos de i con la ecuación 3.

$$f^k_{(i,J_i)}(\rho) = e^{-\left(\frac{(t(i,J_i^h)-\Gamma_{J_i^h})^2}{\sigma_1^2}+\frac{hops^2}{\sigma_2^2}+\frac{(\tau_{J_i^h}-ca(i,J_i^h))^2}{\sigma_3^2}\right)} \quad h = 1, \cdots, m. \quad (3)$$

Sea τ la capacidad del canal de comunicación entre el nodo i y el nodo J_i^h y Γ el tiempo necesario para atender el servicio de transmisión de datos.
Como $t(i, J_i^h)$ y $c(i, J_i^h)$ están dados en términos de μs y $kbps$ respectivamente, para evitar problemas con las escalas de ambos (μs y kbps) han sido escalados al orden 10^1.
3) Calcular las distancias numéricas de los $\frac{m(m-1)}{2}$ pares de entradas de votantes . Las distancias numéricas se calculan usando la ecuación 4.

$$d_{(J_i^h, J_i^{h+1})} = |f^k_{(i,J_i^h)}(\rho) - f^k_{(i,J_i^{h+1})}(\rho)|, \quad h = 1, \cdots, m. \quad (4)$$

4) Determinar el nivel de acuerdo de los $\frac{m(m-1)}{2}$ pares de entradas de votantes, basándose en la ecuación (5).

$$s_{(J_i^h, J_i^{h+1})} = \frac{1}{1 + pd^q_{(J_i^h, J_i^{h+1})}}, \quad h = 1, \cdots, m. \quad (5)$$

Los parámetros p y q son constantes. El parámetro q ajusta el rango de extensión de esta función y p es usado para establecer el valor del punto medio de la función. Para el CRA la distancia numérica es la comparación de la disponibilidad de cada enlace de datos entre los nodos vecinos.
5) Como el valor del nivel de acuerdo ha sido calculado con respecto a los pares de entradas, se asigna un valor de peso para los pares de entradas (iJ_i) votante basándose en la ecuación 6

$$w_{(i,J_i^\omega)} = \frac{1}{1 + \prod_{h\neq\omega,\, h=1}^{m} s_{(J_i^\omega, J_i^h)}}, \quad 0 < w_{(i,J_i^\omega)} < 1, \omega = 1, \cdots, m. \quad (6)$$

6) Se calcula la salida del consenso (ecuación 7)

$$y_i^k = \frac{\sum_{h=1}^m w_{(i,J_i^h)} f^k_{(i,J_i^h)}(\rho)}{\sum_{h=1}^m w_{(i,J_i^h)}}. \quad (7)$$

7) La variable y_i^k es un valor nuevo (cercano a los valores de las entradas), resultado de la ecuación 7, entonces este valor no está en el conjunto de entradas. El valor de y_i^k es comparado con los valores de las entradas, de esta manera el nodo J_i^h correspondiente a la entrada asociada a la ecuación 8.

$$\min(|y_i^k - f^k_{(i,J_i^h)}(\rho)|) = g^k, \quad h = 1, \cdots, m. \quad (8)$$

g^k es escogido como el ganador del consenso, será el nodo fuente (i) para la siguiente iteración. Todos los elementos del conjunto J_i han calculado su propia $y^k_{J_i^h}$ con respecto a si mismos y a sus vecinos inmediatos (su propio conjunto J_i). Los nodos que participaron en el consenso anterior tendrán precalculada su propia $y^k_{J_i^h}$ que podrán aportar en otra ronda de consenso en un paso k posterior.
8) Finalmente, después de varios procesos de descubrimiento de ruta, algún nodo tiene al n_d en su conjunto J_i, entonces, una ruta ha sido encontrada entre n_s y n_d. Esta ruta está conformada por todos los ganadores de cada consenso calculado. Por lo tanto, una ruta de k pasos puede ser construida por estos nodos, representados por la siguiente secuencia:

$$n_s \rightarrow g^1 \rightarrow g^2 \rightarrow \cdots \rightarrow g^k \rightarrow n_d. \quad (9)$$

La ruta no se mantiene, por las condiciones de movilidad. En caso de que se presente un ciclo de encaminamiento, este ciclo puede ser roto por un nodo móvil que cumpla con las mejores condiciones para trasmitir y de esta manera alcanzar el nodo destino [4].

C. Consensus Routing Algorithm Implementado en Hardware

Para la implementación en hardware se han hecho modificaciones a los parámetros del CRA, a continuación se describe la terminología que se utilizará en adelante. El nodo de origen es n_s, los nodos destino son n_d, $i = n_s$ comienza a buscar n_d, al buscar inicialmente en una vecindad radial de radio ρ. El nodo n_s obtiene información sobre la carga y el tiempo promedio de contacto en el planificador del nodo. Para esto, n_s verifica los valores, para sus vecinos directos; para obtener información sobre su carga de enlace; y *(c)* λ, verificando el tiempo promedio de contacto en el planificador de cada nodo en el vecindario. A su vez, n_s informa a sus vecinos que busquen n_d. Tenga en cuenta que mientras n_s realiza un consenso (descrito en los siguientes pasos), los nodos incluidos

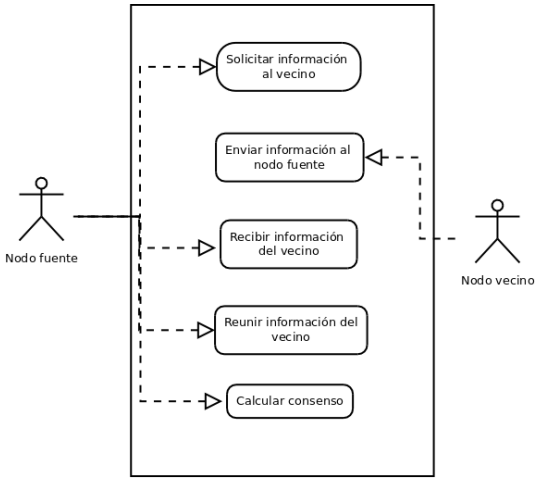

Fig. 3. Diagrama de caso de uso para cada nodo en la topología.

en J_i buscan simultáneamente n_d en sus vecinos inmediatos. Si se encuentra n_d en este punto, el algoritmo finaliza. De lo contrario, los vecinos comienzan un nuevo consenso, ahora con sus propios vecinos directos, calculando la función de disponibilidad. Los pasos del consenso son:

- Calcular las distancias numéricas de los pares de entradas.
- Determinar el nivel de acuerdo de todos los pares de entradas.
- Después de obtener los valores del indicador, de acuerdo para todos los pares de entradas, se asigna un valor de peso para cada par de entrada.
- Luego, se obtiene el resultado del consenso.
- Por lo tanto, el valor de y_i^k se compara con los valores de entrada, por lo que el nodo en J corresponde a la entrada asociada.
- Finalmente, después de varios descubrimientos de ruta, algún nodo debería tener n_d en su J_i, y así, se ha encontrado una ruta entre n_s y n_d. Esta ruta está compuesta por todos los ganadores de cada ronda. Entonces, una ruta de k, que se puede representar como una secuencia: $n_s \rightarrow G^1 \rightarrow G^2 \rightarrow \cdots \rightarrow G^k \rightarrow n_d$. Cada G^k es un mínimo local, para ser mantenido, los valores de β y λ no cambian. La elección del mínimo local proporciona el nodo más adecuado para transmitir entre nodos accesibles en el paso k th.

El CRA hace uso del concepto del vecindario l para prevenir la inundación completa de la RT. El algoritmo se probó considerando el número de saltos, como parámetro para definir el alcance del vecindario l. La implementación del hardware se realizó utilizando dispositivos Arduino Yun, el software se desarrolló en lenguaje Python. En este artículo, se muestran los resultados de la experimentación de la versión móvil del CRA. Se identificaron dos casos de uso para cada nodo, cada nodo tiene ambos casos de uso programados, pero solo uno puede estar activo a la vez. A continuación se muestra una descripción de cada caso de uso

El primer caso de uso es el correspondiente al nodo fuente, es decir, el nodo que busca al objetivo, este nodo ejecuta las funciones para buscar, recibir y recopilar y procesar información del vecindario. El segundo caso de uso corresponde a un nodo vecino que solo realiza la función de enviar la información solicitada. Las actividades realizadas por casos de uso se especifican en Fig. 3.

Para los experimentos, se utiliza una infraestructura de 15 dispositivos (Arduino Yun) con una conexión inalámbrica, que se muestra en la Fig. 4. Cada uno de estos dispositivos tiene un sistema operativo Linino, que está basado en OpenWRT, así como las bibliotecas de Python en el que se implementó el CRA. Cada nodo funciona de forma independiente y el estado base ejecuta permanentemente un server-socket. Se desarrolló una biblioteca que gestiona la comunicación entre dispositivos, esta biblioteca se basa en la biblioteca de server-socket de Python.

Cuando un nodo (n_s) comienza a buscar al nodo destino(n_d), primero busca en su vecindario, en su propia información de vecino, si no encuentra el destino en el vecindario, se comunica con todo el vecindario solicitando información de carga y tiempo de contacto promedio, activando a los nodos del vecindario para que sus nodos vecinos puedan obtener información de su propio vecindario acotado.

Dadas las limitaciones técnicas, como el hecho de no tener baterías que permitan la independencia de los dispositivos, se diseñó una topología para RT y se simuló la movilidad en el código (los parámetros simulados son la topología, el vecindario local y el tiempo promedio de contacto), el resto de los datos se obtuvieron en tiempo real de los datos obtenidos de los dispositivos durante la ejecución del algoritmo en cada uno de estos. Para hacer el conjunto experimental, se desarrollaron tres bibliotecas en lenguaje Python. La funcionalidad de cada uno se describe a continuación:

- *Comunicación:* Esta biblioteca de funciones gestiona la comunicación entre dispositivos. Se basa en la biblioteca de server-socket. Contiene los módulos necesarios para enviar, recibir y solicitar información, así como para que cada nodo ejecute los dos casos de uso descritos anteriormente. Además, contiene módulos responsables de obtener la información local de cada nodo y buscar el nodo destino en el vecindario definido y delimitado. Para evitar inundaciones, tiene una función que limita la propagación de solicitudes de información a dos saltos del nodo de origen real. Cada nodo actúa de forma independiente y permanece en un estado base, en este estado se ejecuta un server-socket, esperando que otro nodo solicite información de él.
- *Buscar:* Cuando un nodo comienza el proceso de búsqueda, primero busca en su vecindad inmediata; si no puede encontrarlo, solicita información a sus vecinos (tiempo promedio de contacto y carga en el canal de comunicación), al mismo tiempo, activa el vecindario para encontrar el nodo destino en sus propios vecindarios.
- *Calcular:* Se desarrolló una biblioteca para llevar a cabo los cálculos de consenso utilizados por el CRA.

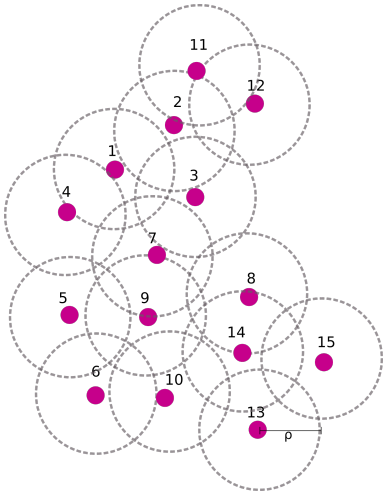

Fig. 4. Topología experimental compuesta por 15 dispositivos arduino yun con un ρ fijo.

Para fines de la experimentación, se utilizó la topología que se muestra en la Fig. 4, esta es fija y se generó cuidando que el nodo objetivo y destino no tuvieran contacto directo, a lo a largo de la búsqueda del nodo destino, se mostrará la evolución de dos casos de búsqueda. Cada nodo tiene una lista de los nodos que están dentro de su vecindario, la función para obtener la carga se encuentra dentro del módulo comunicación, son valores reales de los dispositivos usados. En el primer caso n_s será el nodo 4 y n_d será el nodo 3, como se puede ver en Fig.4, estos nodos no tienen comunicación directa; en el segundo, el n_s será el nodo 2 y el n_d el nodo 9, los cuales se encuentran más lejanos.

El software desarrollado se encuentra disponible de manera pública en [26].

V. Resultados y Discusión

En esta sección, se muestran los resultados de los dos experimentos.

A. Experimento 1: Búsqueda con un Destino Cercano

Usando la topología que se muestra en Fig. 4, el proceso de búsqueda se describe a continuación:

1) El nodo 4 es el nodo fuente, comienza a buscar el nodo 3 (Fig. 5).

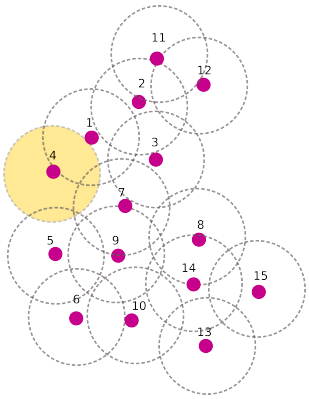

Fig. 5. Nodo fuente del experimento 1.

El nodo 4 busca la vecindad inmediata ya que el nodo 3 no está dentro de la vecindad (Fig. 6), el CRA comienza el siguiente paso para calcular la ruta. Hay dos nodos que pueden ganar el proceso de consenso local y comunicarse con el nodo destino (esto depende de las condiciones locales de cada nodo). Es esencial decir que los nodos que no obtienen el consenso local están deshabilitados y vuelven al estado base, como es el caso de los nodos 4 y 5.

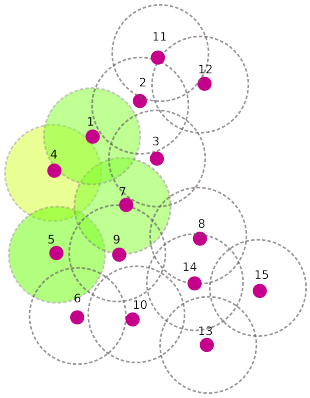

Fig. 6. El nodo 4 busca el vecindario inmediato ya que el nodo 3 no está dentro del vecindario.

2) Puesto que no hay comunicación distinta de las solicitudes de información nodo local, no hay manera de informar a los nodos que hay más de un ganador, por lo que el mensaje será recibido por el nodo 3 a partir de dos fuentes diferentes, en este caso, nodos 1 y 7 (Fig. 7).

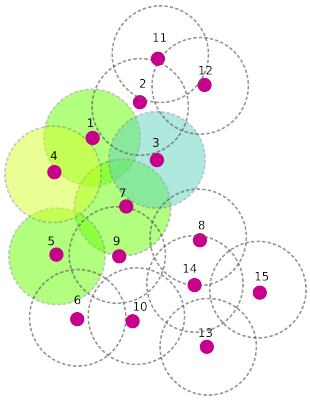

Fig. 7. Si existe más de un ganador, el mensaje será recibido por el nodo 3 desde t nodos 1 y 7.

El tiempo total de búsqueda exitosa comenzando en el nodo 3 para encontrar el nodo 4 fue de 0.31 s en promedio. Se ha medido los retardos ocasionados por las funciones correspondientes a: obtener la información local del nodo, búsqueda del nodo destino en el vecindario y la activación del nodo ganador; estos se muestran en Fig. 8, en ausencia de un reloj global, las horas que se muestran en los gráficos son locales.

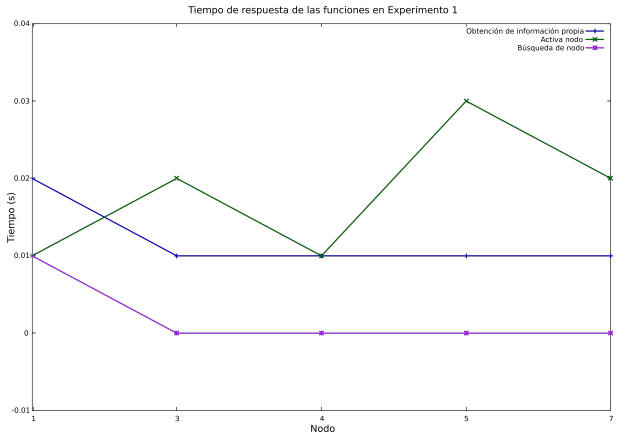

Fig. 8. Retardos de obtener la información local, búsqueda del nodo destino y activación del nodo ganador de cada ronda (en el experimento 1).

B. Experimento 2: Búsqueda con Destino Lejano

1) En este caso el nodo fuente es el 2 y el nodo destino es 9.
2) En el inciso a) de la Fig. 9 se muestra la localización del nodo 2. El nodo fuente busca en su vecindario inmediato, que está conformado por los nodos $11, 12, 1, 3$ ($J_2^1 = \{11, 12, 1, 3\}$), como se muestra en inciso b).

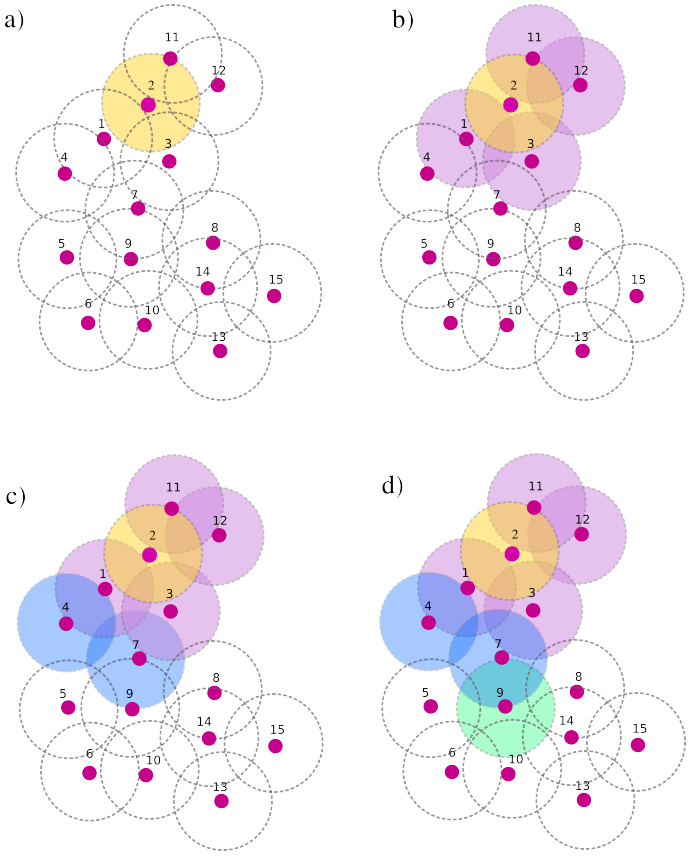

Fig. 9. Secuencia de búsqueda del CRA desde el nodo 2 al nodo 9.

3) Al no estar el nodo 9 en el vecindario (J_2^1), cada uno de los vecinos comienza su propia búsqueda por lo que se forman los vecindarios $J_{11}^2 = \{2, 12\}$, $J_{12}^2 = \{2, 11, 3\}$, $J_1^2 = \{4, 7, 3, 2\}$ y $J_3^2 = \{12, 2, 1, 7\}$.
4) Al no encontrarse el destino en el vecindario extendido, se elige un nodo que seguirá con la búsqueda, por lo que se comienza la ronda de consenso, en este caso el nodo 1 el que continuará el proceso.
5) Cada miembro del vecindario de J_1^2 comenzará su propia búsqueda por lo que se generan los vecindarios $J_4^3 = \{1, 7, 5\}$, $J_7^3 = \{4, 9, 5, 1, 3, 8\}$, $J_3^3 = \{12, 2, 1, 7\}$ y $J_2^3 = \{11, 12, 1, 3\}$. Como el destino se encuentra en el vecindario inmediato del nodo 7, este es el nodo ganador de la ronda y termina la búsqueda, como se muestra en el inciso d) de la Fig. 9.
6) La ruta final está conformada por los ganadores de cada etapa que en este caso son $2^1 \rightarrow 1^2 \rightarrow 7^3 \rightarrow 9$, dónde el superíndice es el número de ronda en la que el nodo resultó ganador.

El tiempo total de búsqueda exitosa comenzando en el nodo 2 para encontrar el nodo 9 fue de 0.39 s en promedio. Se ha medido los retardos ocasionados por la funciones: obtener la información local del nodo, búsqueda del nodo destino en el vecindario y la activación del nodo ganador (Fig. 10), en ausencia de un reloj global, las horas que se muestran en los gráficos son locales.

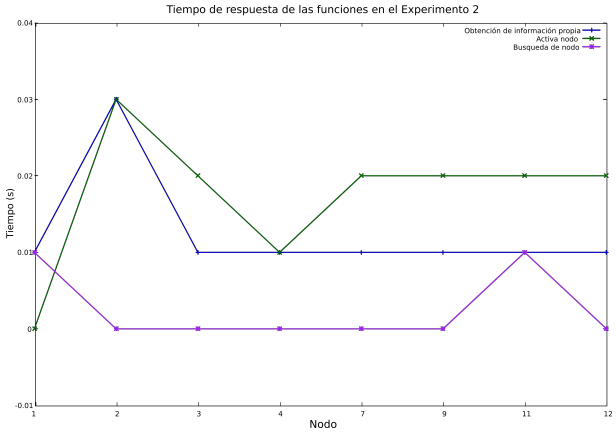

Fig. 10. Retardos de obtener la información local, búsqueda del nodo destino y activación del nodo ganador de cada ronda (en el experimento 2).

VI. Conclusiones

El uso del CRA para la búsqueda de rutas en redes temporales resulta viable, ya que este se adapta a las características de ellas. En la adaptación del algoritmo presentado en este trabajo se muestra como tomando en cuenta las condiciones de los nodos es posible establecer una ruta a través de establecer una serie de consensos. Dado que la red es dinámica el CRA es útil mientras el tiempo de contacto promedio sea mayor al tiempo de cálculo de la ruta. Como trabajo futuro se plantea tener más de un nodo fuente e incluir la movilidad de nodos intermedios, lo que modificaría en gran medida la topología.

Agradecimientos

Este trabajo está parcialmente financiado por el proyecto 735 de Cátedras CONACYT, “Fortalecimiento del laboratorio

Nacional de Internet del Futuro". Los autores agradecen al Ing. Adrián Durán Chavesti por su colaboración en la revisión de este texto.

REFERENCES

[1] P. Holme and J. Saramäki, "Temporal networks," *Physics Reports*, vol. 519, no. 3, pp. 97–125, 2012.

[2] S. H. Lee and P. Holme, "Navigating temporal networks," *Physica A: Statistical Mechanics and its Applications*, vol. 513, pp. 288–296, 2019.

[3] M. Li and H. Dankowicz, "Impact of temporal network structures on the speed of consensus formation in opinion dynamics," *Physica A: Statistical Mechanics and its Applications*, vol. 523, pp. 1355–1370, 2019.

[4] M. Arellano-Vázquez, "Estudio de sistemas adaptables de ruteo para sistemas distribuidos móviles," Ph.D. dissertation, Posgrado en Ciencia e Ingeniería de la Computación. Universidad Nacional Autónoma de México, 2015.

[5] F. Aponte, M. Pineda, L. Gutierrez, I. Meriño, A. Salazar, and P. Wightman, "Cluster-based classification of blockchain consensus algorithms," *IEEE Latin America Transactions*, vol. 100, no. 1e, 2020.

[6] J. P. John, E. Katz-Bassett, A. Krishnamurthy, T. Anderson, and A. Venkataramani, "Consensus routing: The internet as a distributed system," in *Proceedings of the 5th USENIX Symposium on Networked Systems Design and Implementation*, 2008, pp. 351–364.

[7] A. Fagiolini and A. Bicchi, "On the robust synthesis of logical consensus algorithms for distributed intrusion detection," *Automatica*, vol. 49, no. 8, pp. 2339–2350, 2013.

[8] A. Benchi, P. Launay, and F. Guidec, "Solving consensus in opportunistic networks," in *Proceedings of the 2015 International Conference on Distributed Computing and Networking*, 2015, pp. 1–10.

[9] A. Vahdat, D. Becker *et al.*, "Epidemic routing for partially connected ad hoc networks," 2000.

[10] Y.-F. Hsu, C.-L. Hu, and H.-J. Hsiao, "On exploiting temporal periodicity for message delivery in mobile opportunistic networks," in *2018 IEEE 42nd Annual Computer Software and Applications Conference (COMPSAC)*, vol. 1. IEEE, 2018, pp. 809–810.

[11] Y. O. Yazir, R. Farahbod, A. Guitouni, S. Ganti, and Y. Coady, "Adaptive routing in mobile ad hoc networks based on decision aid approach," in *Proceedings of the 8th ACM International Workshop on Mobility Management and Wireless Access*, 2010, pp. 1–10.

[12] P. Jacquet, P. Muhlethaler, T. Clausen, A. Laouiti, A. Qayyum, and L. Viennot, "Optimized link state routing protocol for ad hoc networks," in *Proceedings. IEEE International Multi Topic Conference, 2001. IEEE INMIC 2001. Technology for the 21st Century.* IEEE, 2001, pp. 62–68.

[13] D. B. Johnson and D. A. Maltz, "Dynamic source routing in ad hoc wireless networks," in *Mobile Computing*. Springer, 1996, pp. 153–181.

[14] C. E. Perkins and E. M. Royer, "Ad-hoc on-demand distance vector routing," in *Proceedings WMCSA'99. Second IEEE Workshop on Mobile Computing Systems and Applications*. IEEE, 1999, pp. 90–100.

[15] K. Miyazaki, "First-come first-served routing for the data center network," *WTC2012, Mar.*, 2012.

[16] E. Rojas, G. Ibañez, J. M. Gimenez-Guzman, J. A. Carral, A. Garcia-Martinez, I. Martinez-Yelmo, and J. M. Arco, "All-path bridging: Path exploration protocols for data center and campus networks," *Computer Networks*, vol. 79, pp. 120–132, 2015.

[17] B. Mao, Z. M. Fadlullah, F. Tang, N. Kato, O. Akashi, T. Inoue, and K. Mizutani, "Routing or computing? the paradigm shift towards intelligent computer network packet transmission based on deep learning," *IEEE Transactions on Computers*, vol. 66, no. 11, pp. 1946–1960, 2017.

[18] F. Tang, B. Mao, Z. M. Fadlullah, N. Kato, O. Akashi, T. Inoue, and K. Mizutani, "On removing routing protocol from future wireless networks: A real-time deep learning approach for intelligent traffic control," *IEEE Wireless Communications*, vol. 25, no. 1, pp. 154–160, 2017.

[19] F. Geyer and G. Carle, "Learning and generating distributed routing protocols using graph-based deep learning," in *Proceedings of the 2018 Workshop on Big Data Analytics and Machine Learning for Data Communication Networks*, 2018, pp. 40–45.

[20] T. Winter, P. Thubert, A. Brandt, J. W. Hui, R. Kelsey, P. Levis, K. Pister, R. Struik, J.-P. Vasseur, R. K. Alexander *et al.*, "Rpl: Ipv6 routing protocol for low-power and lossy networks." *rfc*, vol. 6550, pp. 1–157, 2012.

[21] T. Clausen, J. Yi, and A. C. De Verdiere, "LOADng: Towards AODV Version 2," in *2012 IEEE Vehicular Technology Conference (VTC Fall)*. IEEE, 2012, pp. 1–5.

[22] V. Nicosia, J. Tang, C. Mascolo, M. Musolesi, G. Russo, and V. Latora, "Graph metrics for temporal networks," in *Temporal Networks*. Springer, 2013, pp. 15–40.

[23] L. E. Rocha and N. Masuda, "Random walk centrality for temporal networks," *New Journal of Physics*, vol. 16, no. 6, p. 063023, 2014.

[24] P. Holme, "Network reachability of real-world contact sequences," *Physical Review E*, vol. 71, no. 4, p. 046119, 2005.

[25] J. Macfarlane, *Network routing basics: Understanding IP Routing in Cisco Systems*. John Wiley & Sons, 2007.

[26] M. Arellano-Vázquez, "Consensus routing algorithm," https://github.com/magali-e/ConsensusRoutingAlgorithm, 2015.

Magali Arellano-Vázquez Doctora en Ciencias de la Computación por la UNAM. Actualmente es investigadora Cátedra Conacyt, asignada a INFOTEC Centro de Investigación e Innovación en Tecnologías de la Información y Comunicación, en el laboratorio Nacional de Internet del Futuro, en Aguascalientes, México. Sus áreas de interés son: cómputo distribuido, energía eólica, sistemas complejos, redes temporales, análisis de datos, algoritmos de agrupación.

Héctor Benítez-Pérez Investigador a tiempo completo IIMAS UNAM (México). Recibió una licenciatura en Ingeniería Electrónica en 1994 en la UNAM. Obtuvo su Ph.D. en la Universidad de Sheffield, Reino Unido, en 1999. Sus intereses de investigación son el control en tiempo real, el modelado de sistemas distribuidos en tiempo real y los sistemas de detección de fallas.