UNIVERSIDAD RICARDO PALMA

ESCUELA DE POSGRADO

MAESTRÍA EN INGENIERÍA DE TELECOMUNICACIONES

Para optar el Grado Académico de Maestro en Ingeniería de Telecomunicaciones.

Diseño e implementación de un Algoritmo de Hormigas para el manejo dinámico del encaminamiento de flujos de datos a nivel de la capa de aplicación

Autor: Bachiller Alfredo Efraín Rodriguez Gutierrez.

Asesor: M.SC. Jorge Gustavo Butler Blacker.

LIMA-PERÚ

2018

Dedicatoria:

Dedico este trabajo a toda mi familia, amigos que de alguna manera me apoyaron en continuar con el desarrollo profesional.

ÍNDICE DE CONTENIDOS

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

RESUMEN

En el presente trabajo primero se ha diseñado e implementado un protocolo de comunicaciones cuyo propósito es detectar, en forma dinámica, los cambios en la topología lógica, ya sea en la desactivación de los enlaces o en la caída de los nodos. Luego se ha diseñado e implementado el algoritmo basado en la colonia de hormigas, para reconstruir las nuevas rutas de manera dinámica, es decir, el algoritmo selecciona entre las diferentes rutas que se puede tener entre dos nodos, la ruta óptima (menor costo). Este costo está en función del ancho de banda teniendo en cuenta que a mayor ancho de banda menor costo y a mayor tráfico mayor costo.

Se utilizaron cinco topologías diferentes; la *Topología 1* consta de un servidor y siete nodos (8), la *Topología 2* consta de un servidor y 19 nodos (20), la *Topología 3* consta de un servidor y 39 nodos (40), la *Topología 4* consta de un servidor y 59 nodos (60), y la *Topología 5* consta de un servidor y 99 nodos (100).

De los resultados obtenidos, por ejemplo, en la *topología 1*, se elimina la ruta o enlace de peso 4, que conecta al nodo N1 con el nodo N3. El protocolo de comunicaciones detecta este cambio y el algoritmo de hormigas encuentra que la nueva ruta para llegar del nodo N1 al nodo N3 es a través del nodo N2, cuyo peso total es de 8.

En las demás topologías se han anulado enlaces, modificando la topología y de acuerdo con los resultados obtenidos demuestra que el protocolo de comunicaciones detecta cualquier cambio topológico de la red de comunicaciones, y el algoritmo de hormigas busca las nuevas rutas, basado en los datos que le proporciona el protocolo de comunicaciones. Se ha comprobado que siempre busca la ruta más adecuada en base al menor costo de los enlaces.

***Palabras Claves*:** Algoritmo de hormigas, Manejo dinámico del encaminamiento de flujos de datos, Conmutación, Heurísticos, Nodos.

ABSTRACT

In the present investigation, a communications protocol has been designed and implemented, the purpose of which is to dynamically detect changes in the logical topology, either in the deactivation of the links or in the nodes fall. Then, the ant colony algorithm has been designed and implemented to reconstruct the new paths in a dynamic way, the algorithm selects between the different routes that can be between two nodes, the optimal route (least cost). This cost is in function of the bandwidth taking into account that the higher bandwidth the cost is lower and the higher traffic the cost is higher.

Five different topologies were used; Topology 1 consists of one server and seven nodes (8(, Topology 2 consists of one server and 19 nodes (20), Topology 3 consists of one server and 39 nodes (40), Topology 4 consists of one server and 59 nodes (60), and Topology 5 consists of one server and 99 nodes (100).

From the results obtained, for example, in the topology 1, the weight route or link 4, which connects the node N1 to the node N3, is eliminated. The communication protocol detects this change and the ant algorithm finds that the new route to arrive from node N1 to node N3 is through node N2, whose total weight is 8.

From the results obtained, for example, in the topology 1, the weight route or link 4, which connects the node N1 to the node N3, is eliminated. The communication protocol detects this change and the ant algorithm finds that the new route to arrive from node N1 to node N3 is through node N2, whose total weight is 8.

In the other topologies, links have been cancelled, modifying the topology and according to the obtained results shows that the communication protocol detects any topological changes of the communications network, and the ant algorithm searches the new routes, based on the data that gives you the communications protocol. It has been proven that it always searches for the most appropriate route based on the lowest cost of the links.

Keywords: Ant algorithm, Dynamic routing of data flows, Switching, Heuristics, Nodes.

CAPÍTULO I

PLANTEAMIENTO DEL ESTUDIO

1.1.Introducción

El problema de encaminamiento a nivel de la capa de Red (Nivel 3 del modelo OSI), está resuelto mediante los algoritmos de Dijkstra, de Bellman-Ford, etc. Las aplicaciones más interesantes se encuentran en los problemas de programación lineal. Los primeros modelos matemáticos que fueron empleados, derivados de las ciencias exactas, en parte por su facilidad de manejo, fueron los sistemas de ecuaciones lineales. A medida que estos métodos se fueron extendiendo en su aplicación, aparecieron nuevos problemas en los cuales el proceso de linealidad ya no fue tan sencillo, ya sea por la interconectividad en los procesos, o por la escala de magnitud de las variables involucradas, ciertos problemas resultaron imposibles de ser resueltos por los métodos convencionales.

En muchos casos, cuando el universo de posibles soluciones ha sido demasiado grande, se ha recurrido a métodos heurísticos: Conjunto de mecanismos o algoritmos específicos diseñados para cada problema. Estos métodos recurren a procesos estocásticos en la generación de soluciones, para luego afinarlas mediante la aplicación de algún proceso de selección, terminando por ofrecer soluciones; los métodos heurísticos pocas veces pueden extrapolarse con facilidad a soluciones para otros problemas similares.

En este trabajo estudió y experimentó un caso de aplicación del ACO, como método metaheurístico proveniente de un modelo biológico. Esta aplicación consiste en buscar las rutas más adecuadas en una red de nodos y muestra los diferentes caminos para llegar de un nodo inicial a un nodo final que se encuentran en los extremos de la red, estas trayectorias pasan por nodos intermedios que son los nodos de tránsito.

1.2.Formulación del problema y justificación del estudio

Formulación del problema

La tendencia actual de integrar todo tipo de servicios en una única arquitectura de red TCP/IP, conocida también como modelo “Todo IP” (All-IP), genera como resultado la convergencia de servicios; que no solo permiten beneficiarnos de una disminución de costos, al proveer diferentes servicios sobre la misma infraestructura y arquitectura, sino además dan la oportunidad de obtener provecho de nuevas posibilidades como la ubicuidad y la movilidad. Estos últimos, se consiguen con el uso de aplicaciones distribuidas que se encuentran en la “nube” al alcance de todos los usuarios en cualquier momento y desde cualquier lugar. Estos nuevos enfoques están acompañados con nuevos retos que se deben asumir; entre ellos, se debe garantizar la calidad de servicio de las aplicaciones, la disponibilidad continua de los servicios, la confiabilidad y la seguridad. Para dar soporte a estos nuevos requerimientos, han aparecido nuevas soluciones, entre ellas están las Redes de Nueva Generación (NGN: Next Generation Network), Redes SDN (Redes en Definidas por Software).

En este complejo entorno de redes convergentes y distribuidas, las funciones de enrutamiento se encuentran en la capa de Internet (equivalente a la capa 3 del modelo OSI) de acuerdo la arquitectura TCP/IP, y manejan el envío de paquetes mediante la red. Los servicios que han funcionado por años en redes con diferentes características y ahora están migrando a una red “todo IP”, necesitan que sus requerimientos se cumplan en esta nueva red, por lo que, en este escenario, donde el manejo de servicios de la capa de aplicación requiere la garantía de disponibilidad y confiabilidad de los servicios, sobre todo, desde el punto de vista de las comunicaciones, por lo tanto, se necesitan nuevos algoritmos de encaminamiento de los flujos de datos en la capa de aplicación.

Problema Principal

¿Es posible diseñar e implementar, un sistema para el manejo dinámico del encaminamiento de flujos de datos a nivel de la capa de aplicación ante cambios repentinos en la topología lógica de la red de comunicaciones?

Problemas secundarios

a) ¿Es posible Implementar un protocolo de comunicaciones para detectar los cambios en la topología lógica de comunicaciones a tiempo y de manera dinámica?
b) ¿Es posible diseñar e implementar un algoritmo basado en la colonia de hormigas para reconstruir el encaminamiento de manera dinámica?
c) ¿Es posible determinar que el tiempo de convergencia de los algoritmos depende del tamaño de la red?
d) ¿Es posible determinar cómo el ancho de banda de los enlaces influye en determinar la ruta adecuada para el algoritmo de hormigas?

Justificación del estudio

Como se mencionó en la sección anterior, la integración de los servicios que antes operaban sobre redes de conmutación de circuitos o de difusión, actualmente deben operar sobre un modelo de conmutación de paquetes. En esta arquitectura, las funciones de enrutamiento se manejan en la capa de Internet, a este nivel los paquetes de todos los servicios se conmutan nodo a nodo hasta su destino.

Para garantizar la confiabilidad existen soluciones de ingeniería de tráfico que permiten priorizar paquetes basados en los flujos de información establecidos; sin embargo, estas soluciones, se aplican en la capa de Internet, mientras los servicios se ejecutan a nivel de la capa de aplicación de la arquitectura TCP/IP (Protocolo de control de transmisión/Protocolo de Internet).

A nivel de la capa de aplicación, los servicios que han sido migrados desde redes tradicionales como las de conmutación de circuitos (por ejemplo, la red telefónica), necesitan encaminar los flujos de voz o video a través de nodos específicos, siguiendo las topologías lógicas de los nodos de comunicaciones que conforman el servicio. Las rutas entre los nodos pueden ser rutas principales o rutas alternativas, de tal manera que garantice la continuidad del servicio ante posibles caídas de uno o más nodos. Este comportamiento se puede pre configurar en cada nodo, estableciendo de forma manual el orden que debe utilizarse para alcanzar el siguiente nodo de comunicación, en caso de fallas; sin embargo, esta solución no ofrece un uso eficiente del ancho de banda y de los enlaces disponibles, considerando que, bajo la arquitectura actual, en la conmutación de paquetes, el ancho de banda será compartido con el resto de los servicios que utilicen los usuarios.

En este trabajo se aplican los algoritmos basados en colonias de hormigas, para la actualización dinámica de las rutas que se establecen entre los nodos proveedores de un mismo servicio a nivel de la capa de aplicación.

Para analizar el comportamiento y eficiencia de la solución, se ha utilizado Máquinas Virtuales en el entorno Linux, instalados en cada computador como nodos de una plataforma de red de nodos, del Laboratorio de Informática de la Facultad de Ingeniería Eléctrica y Electrónica (LIFIEE). Se propone:

- Mayor productividad; las rutas se actualizan dinámicamente de tal manera que para los usuarios sea transparente y estén siempre conectados a los servicios.
- Optimización del ancho de banda; no es necesario establecer rutas de reserva preestablecidas, las rutas se actualizarán dinámicamente bajo demanda, por lo tanto, solo se usarán bajo demanda en caso de fallas, optimizando el uso del ancho de banda.
- Reducción de costos; al identificar las rutas más usadas es posible destinar los recursos de manera óptima ahorrando costos de acceso a la red; además de mantener la disponibilidad constante del servicio a través de los nodos, garantiza un ahorro en servicios de atención a usuarios (reducción de reclamos).

Por lo tanto, este proyecto se justifica basado en:

- Aspecto económico; la optimización de los recursos de ancho de banda, hardware y software, así como los bajos tiempos de restablecimiento del servicio que permitan ahorrar costos. Servicios de disponibilidad constante con tiempos pequeños de restablecimiento. Como ya se mencionó, aumentan la usabilidad y por lo tanto la rentabilidad de los servicios cuando se maneja en entornos corporativos.
- Aspecto Tecnológico; el diseño y la implementación del algoritmo de optimización basado en colonia de hormigas, permite dejar un aporte importante al área de la ingeniería de tráfico de telefonía, sentando las bases para la futura investigación de otros tipos de flujos de datos (video, TV, datos, etc.), en esta área importante de las Comunicaciones.

Ambos criterios justifican el desarrollo de este proyecto para que sea un aporte a la implementación de servicios que se integran sobre redes de conmutación de paquetes y necesiten, como parte de su comportamiento, el interconectar nodos y manejar flujos de datos en la capa de aplicación.

1.3.Antecedentes relacionados con el tema

Aguilar & Labrador (2007)**.** Proponen un algoritmo de enrutamiento distribuido para redes de comunicaciones basado en Sistemas de Hormigas, donde el enrutamiento para redes de comunicaciones es modelado como un problema de optimización dinámico. En el algoritmo propuesto, el espacio de soluciones del problema es el espacio donde pasean las hormigas, la probabilidad de transición y la función de actualización de la tabla de feromona, son definidos según la función objetivo del problema de comunicación. El enfoque propuesto permite su utilización en diferentes redes de comunicación, cambiando únicamente el criterio de rendimiento a optimizar. Los resultados obtenidos después de comparar el algoritmo propuesto en diferentes ambientes de comunicación demuestran que esta propuesta provee mejores rendimientos en cuanto al retraso y eficacia en el uso de energía, entre otras medidas.

Caballero (2007). Manifiesta que en el ámbito empresarial el tema de las telecomunicaciones toma un interés especial cuando se trata de precios y calidad, la razón es el correcto funcionamiento de las estructuras. Hoy por hoy es bien sabido que si no hay una correcta comunicación entre los niveles que conforman cualquier estructura empresarial, simplemente no funciona como se espera, es por esto por lo que las empresas actuales necesitan de una estructura de comunicación a bajo precio, eficiente y acorde a sus necesidades. Muchas empresas del ramo de las telecomunicaciones ofrecen productos que satisfacen estas necesidades, pero sus soluciones comerciales son cerradas, propietarias, y frecuentemente costosas. Con el Asterisk cambia todo: es una tecnología abierta que proporciona un estándar de comunicaciones VoIP, por lo que no se está sujeto a las limitaciones de ningún fabricante, se está en libertad para desarrollar las innovaciones que se requieran y en general no se impone ningún límite.

Corbalán (2006). En su trabajo desarrola una visión general sobre la actividad científica relacionada con los sistemas inteligentes aplicados en áreas inherentes a las redes de datos. Se aborda la temática del diseño óptimo de redes confiables y algunas soluciones aportadas por los Algoritmos Evolutivos. También se trata el tema de enrutamiento en redes y la aplicación de los sistemas basados en Colonia de Hormigas.

Gainza (2008)**.** La Comisión Nacional de Comunicaciones de Argentina, presenta este trabajo para dar una visión sobre la tecnología y los servicios comúnmente denominados VoIP. En primer lugar, se hace una breve reseña sobre las redes que operan con conmutación de circuitos y se comparan con sistemas que conmutan paquetes de datos, analizando la conexión entre ambas redes. La óptica de este resumen pone énfasis en los aspectos que se relacionan con los servicios de telefonía y transmisión de datos. Luego se introduce una explicación de los sistemas que emplean protocolos IP para brindar servicios de telefonía. El tratamiento de las cuestiones técnicas se completa con una descripción de los escenarios más comunes del empleo de los servicios IP. Luego, se analizan las

implicancias de los servicios con tecnología IP desde el punto de vista de la regulación de los servicios de telecomunicaciones. Finalmente, se discute brevemente el tema de la numeración para servicios VOIP, sus alcances y problemas, la aparición de números nómades y variantes de portabilidad.

Jiménez (2012). Presenta en este artículo el concepto de Inteligencia de Enjambre (Swarm-Intelligence), así como las características de ciertas especies de hormigas, utilizadas para modelar la solución de problemas de clasificación de datos. Se utiliza un algoritmo de clasificación y se presentan los resultados obtenidos en grupos de datos binarios.

López (2011). Su trabajo tiene como objetivo, implementar los protocolos estándares de VoIP, tales como SIP y IAX para probar el funcionamiento y la convivencia de estos protocolos sobre un entorno de red Dual Stack, esta configuración permite que cada dispositivo, dentro de la red, pueda contar con direccionamiento IPv4 e IPv6. Como herramienta de monitoreo de tráfico VoIP se utilizó Wireshark, lo cual permitió obtener datos para un análisis comparativo entre IPv4 e IPv6 en la transmisión de paquetes VoIP.

Domínguez (2011). Plantea que, las redes inalámbricas de sensores se han convertido en un tópico muy activo de investigación en los últimos años. El problema del encaminamiento de datos en las redes inalámbricas de sensores es una parte que debe ser tomada muy en cuenta si se desea maximizar el tiempo de vida de la red y minimizar la latencia en la transmisión de datos. Puesto que el tamaño de este tipo de redes puede incrementarse según la aplicación, el método de encaminamiento de datos se vuelve más complejo debido a la cantidad de nodos sensores que se tienen en la red. Los nodos sensores de estas redes son dispositivos con limitantes considerables tanto en capacidad de computo, como en memoria, comunicación inalámbrica y batería. Por otro lado, los algoritmos de optimización por colonia de hormigas han sido propuestos para tratar de resolver el problema del encaminamiento de datos en redes inalámbricas de sensores. En este trabajo de tesis, se presenta una comparación de dos algoritmos de

encaminamiento de datos basados en la optimización por colonia de hormigas para redes inalámbricas de sensores en diferentes escenarios.

Muñoz, López, & Caicedo (2008). En este artículo se presenta una revisión de los conceptos de inteligencia de enjambres, y algunas perspectivas en la investigación con estas técnicas, con el objetivo de establecer un punto de partida para trabajos futuros en diferentes áreas de la ingeniería. Se llega a establecer la diferencia entre la inteligencia de enjambres y otros algoritmos evolutivos, y una visión amplia de las diferentes técnicas y aplicaciones. Entre los algoritmos mencionados están la optimización por Enjambre de partículas (PSO), optimización por colonia de hormigas (ACO) y optimización por enjambre de bacterias.

Pinto, Estigarribia, & Barán (2005). Manifiestan que los algoritmos de optimización basados en Sistemas de Colonias de Hormigas (Ant Colony Optimization - ACO) son métodos metaheurísticos recientes, inspirados en el comportamiento de colonias de hormigas reales. Este trabajo propone un algoritmo multi-objetivo, para la solución del problema de Enrutamiento Multicast en Ingeniería de Tráfico, denominado Multiobjective Ant Colony System (MOACS), que está basado en ACO y que es utilizado para la construcción del árbol multicast, en el contexto de transmisión de datos en redes de computadoras. El MOACS optimiza de manera simultánea tres parámetros: el costo del árbol multicast, el retardo promedio y el retardo máximo (de origen a destino), y obtiene como resultado un conjunto de soluciones óptimas, denominadas conjunto Pareto. Este conjunto de soluciones óptimas es calculado en una sola ejecución del algoritmo sin la necesidad de considerar decisiones a priori.

Quintana (2007). Su trabajo consiste en analizar, diseñar e implementar una red piloto de telefonía IP en la Red Académica Peruana (RAAP) usando software libre. Durante el desarrollo de este proyecto se realizó una comparación de los diversos protocolos de señalización: SIP, IAX2; del hardware a utilizar:

Teléfonos IP, ATAs; así como también de las diversas clases de CODECS. Luego del análisis, se implementa la red VoIP. Esta red consiste en un servidor principal y otro de respaldo para brindar una alta disponibilidad en caso de fallas. Ambos servidores deben contar con el software Asterisk y un Sistema Operativo GNU/Linux. Una vez implementada la red de VoIP, se realizan pruebas de esfuerzo para determinar la capacidad máxima de llamadas simultáneas que pueda soportar el sistema. Por último, se elabora una recomendación formal a la RAAP sobre el uso de estas tecnologías.

Rupérez (2009). Manifiesta que las redes móviles ad hoc (MANETs) son redes inalámbricas multi-salto de nodos móviles sin infraestructura previa alguna. La topología de estas redes puede cambiar aleatoriamente debido a la movilidad imprevisible de los nodos y a las características de propagación. El objetivo de un protocolo de encaminamiento para redes móviles es conseguir el envío de un mensaje de un nodo a otro sin existir un enlace directo. Este trabajo propone un protocolo de encaminamiento ACO (Ant Colony Optimization) híbrido para este tipo de redes basado en el protocolo AntHocNet. El protocolo se caracteriza por la utilización de rutas de enlace/nodo disjunto, por la separación del proceso de difusión de las feromonas y por el proceso de exploración de rutas, que tiene en consideración el número de saltos de las mejores rutas encontradas anteriormente.

Santamaría & Tejada (2010). Realizan una investigación para implementar un servicio de comunicaciones a distancia de bajo costo. Para esto, se implementa un servicio de telefonía VoIP sobre una plataforma de red inalámbrica Mesh, la cual es ideal para entornos accidentados y donde una señal centralizada se degenera rápidamente por los elementos del entorno. Como valor agregado, se presenta la posibilidad de crear un directorio web, desde el cual se pueden generar llamadas entre teléfonos IP sin realizar el marcado; una solución ideal para establecer enlaces de comunicación a grandes distancias, aprovechando Internet. Se desarrolla también un sistema remoto de monitoreo del estado de la red mesh a través de Google maps; en este sistema desde el sitio Google se puede

obtener el estado de los enrutadores inalámbricos e información de red asociado a cada nodo.

Almeida (2015). Manifiesta que el servicio de telefonía es un medio de comunicación indispensable en todo momento, desde el hogar, centros educativos (Universidades, colegios, escuelas), negocios de toda índole (empresas estatales y privadas), hasta para uso personal. El uso de esta tecnología no solamente radica en la cantidad de usuarios que poseen un teléfono o un celular sino en la calidad de servicio que ofrecen a un costo bajo. Todos estos organismos en todo momento necesitan el servicio telefónico que esté disponible, sea accesible sin interrupciones o congestionamiento y sea fácil de usar manteniendo la seguridad y escalabilidad en el tiempo. Debido a los grandes avances tecnológicos constantes, y siendo parte de esta innovación se implementa un sistema de telefonía de voz sobre el protocolo de internet, solución adecuada para la problemática que tiene la Carrera de Ingeniería en Sistemas Computacionales y networking, siendo un gran impulso para la mejora de la Institución.

1.4. Objetivo general y específicos

Objetivo General

Diseñar e implementar un sistema basado en el algoritmo de hormigas, para el manejo dinámico del encaminamiento de flujos de datos a nivel de la capa de aplicación, ante cambios repentinos en la topología lógica de la red de comunicaciones.

Objetivos Específicos

a) Implementar un protocolo de comunicaciones, que permita detectar los cambios en la topología lógica de comunicaciones a tiempo y de manera dinámica.
b) Diseñar e implementar un algoritmo, en la programación Java, basado en la colonia de hormigas, para reconstruir el encaminamiento de manera dinámica.

c) Determinar que el tiempo de convergencia de los algoritmos depende del tamaño de la red.
d) Determinar cómo el ancho de banda de los enlaces influye en determinar la ruta adecuada para el algoritmo de hormigas.

1.5.Limitaciones del estudio

El estudio se desarrolló utilizando herramientas de programación de la capa de aplicación, programación Java, y en una plataforma de código abierto, sistema operativo Linux, estas herramientas son ampliamente utilizadas a nivel mundial y se desarrolló en el Laboratorio de Informática de la Facultad de Ingeniería Eléctrica y Electrónica de la Universidad Nacional de Ingeniería (LIFIEE).
El alcance de este estudio está enmarcado por las siguientes limitaciones:

- El diseño del algoritmo de optimización, basado en colonia de hormigas, fue desarrollado sobre los servicios de la capa de aplicación, que han sido migrados desde redes tradicionales basadas en la conmutación de circuitos, y ahora son implementados en redes IP (telefonía IP).
- El algoritmo se ha implementado, para su estudio y aplicación, utilizando lenguajes y herramientas de código abierto (Linux, Java), en una red lógica de comunicaciones.
- En cuanto a la plataforma se utilizó aplicaciones de código abierto, es decir los nodos de la red y el servidor utilizan el sistema Operativo Linux. Estos servidores están instalados en cada computador personal (PC) así como en un entorno virtualizado para garantizar un despliegue más rápido y confiable durante las pruebas.
- En cuanto al hardware, éste se dimensionó de acuerdo a las necesidades de la implementación, para las simulaciones se utilizan computadoras personales (PC) y laptops en las cuales se ejecuten los servidores como máquinas virtuales, por lo tanto, es suficiente el hardware convencional que se emplea actualmente.

CAPÍTULO II

MARCO TEÓRICO

2.1. Bases teóricas relacionadas con el tema

En la actualidad los algoritmos de enrutamiento deben tener la capacidad de enfrentar problemas en las redes modernas, como las condiciones de tráfico, la estructura de la red, y los recursos de la red (los cuales con el paso del tiempo son limitados y están en constante cambio). Esto es característico en las redes inalámbricas y sensoriales, en las cuales la movilidad de los nodos y los fallos de los dispositivos producen cambios constantes en la topología de la red. Para estas redes, los algoritmos dinámicos de enrutamiento son el único enfoque factible. De hecho, la falta de adaptabilidad de los algoritmos de enrutamiento ante los cambios frecuentes de: La topología de la red, de las capacidades de los nodos, de los modelos de tráfico, de la carga, de la disponibilidad de energía, entre otros, reduce el rendimiento en las redes.

Parámetros de los Algoritmos de enrutamiento

1). Barbancho, y otros (2010). Definen los siguientes conceptos:

- Protocolo vector distancia: Busca el camino más corto determinando la dirección y la distancia a cualquier nodo de la red. Estos algoritmos de enrutamiento basados en vectores pasan copias periódicas de una tabla de enrutamiento de un router a otro y acumulan vectores distancia. Las métricas usadas habitualmente por los routers son:
- Número de saltos: Número de routers por los que pasa un paquete.
- Pulsos: Retraso en un enlace de datos usando pulsos de reloj del procesador.
- Coste: Valor arbitrario, basado generalmente en el ancho de banda, el coste económico y otra medida.
- Ancho de banda: Capacidad de datos de un enlace.

- Carga: Cantidad de actividad existente en un recurso de la red, como un router o un enlace.
- Fiabilidad: La tasa de errores de bits de cada enlace de la red.
- MTU (Maximun Transmission Unit): La Unidad Máxima de Transmisión, es la longitud de la trama en octetos que puede ser aceptada por los enlaces de la ruta.
- Protocolos de estado de enlace: Crean tablas de enrutamiento basándose en una base de datos de la topología. Esta base de datos se elabora a partir de paquetes de estado de enlace que se pasan entre todos los routers para describir el estado de una red.

2). El Algoritmo de Bellman-Ford, es un algoritmo de cálculo de ruta basado en el camino más corto. En el caso de Bellman-Ford, el cálculo de la ruta óptima se reduce a calcular el camino más corto entre un nodo de origen y un nodo de destino. Por lo tanto, en este caso no se suele hablar de coste asociado a un enlace, ya que normalmente la métrica que se asocia a un enlace es uno. Al final, el camino más corto tendrá como coste la suma de todas las métricas de cada uno de los enlaces que, por lo general, no será otra que el número de enlaces por los que ha pasado. Gil, Pomares, & Candelas (2010).

3). Algoritmo de Dijkstra (1959-2002), está diseñado para encontrar las rutas más cortas entre el nodo de origen y cada uno de los nodos de la red. Este algoritmo es de tipo “greedy” porque en cada iteración elige la mejor opción de las posibles con la esperanza de encontrar así la mejor solución global. Una característica de este algoritmo es la utilización de etiquetas en cada nodo cuya función es indicar en cada iteración del algoritmo la distancia de origen a dicho nodo. En cada iteración una de las etiquetas será “permanente”, es decir, indicará la distancia mínima final del nodo inicial a dicho nodo. Alonso (2008).

4). Metaheurística de optimización de colonia de hormigas: Según, Rodrigo (2011): “La Metaheurística de optimización de colonia de hormigas es un método probabilístico basado en el comportamiento natural de las colonias de hormigas

reales. En la misma manera en que las hormigas van en búsqueda de una trayectoria desde su hormiguero hasta el alimento, el algoritmo de colonia de hormigas usa una colonia de hormigas artificiales en búsqueda de la mejor solución a un problema de optimización.

El factor clave de operación de estos algoritmos se fundamenta en la comunicación indirecta entre unos agentes de software, simulando el comportamiento de las hormigas. Esa forma de comunicación se conoce como "estimergia" en la cual las hormigas realizan modificaciones del ambiente obrando de manera cooperativa. Esto se lleva a cabo por una hormiga en el camino de vuelta al hormiguero luego de encontrar alimento, dejando un rastro de una pequeña cantidad de feromona en su trayectoria capaz de ser detectada por las demás hormigas. Las hormigas que consigan el rastro de feromona depositado por la hormiga anterior seguirán ese mismo camino para ir hasta la comida y volver, dejando a su vez un rastro nuevo de feromona por la misma trayectoria, aumentando la probabilidad de que las siguientes hormigas sigan el mismo camino hasta el alimento.

Varias hormigas pueden dejar su rastro de feromona por sus trayectorias individuales, esto podría ser un problema debido a que cada camino tendría la misma probabilidad, pero las feromonas se evaporan con el tiempo, los caminos más largos tendrán niveles de feromona menores a los caminos más cortos permitiendo entonces que las hormigas tomen el camino más corto al alimento." (págs. 35-36).

2.2. Definición de términos usados.

1) Red Próxima Generación (Next Generation Networking o NGN en inglés): **S**e refiere a la evolución de la infraestructura actual de redes de telecomunicación y acceso telefónico con el objetivo de lograr la convergencia tecnológica de los nuevos servicios multimedia (voz, datos, video...) en los

próximos 5-10 años. La idea principal que se esconde debajo de este tipo de redes es el transporte de paquetes encapsulados de información a través de Internet. Estas nuevas redes serán construidas a partir del protocolo IP, siendo el término "All-IP" comúnmente utilizado para describir dicha evolución.

2) Algoritmo de las hormigas: Técnica probabilística utilizada para solucionar problemas de cómputo, este algoritmo está inspirado en el comportamiento que presentan las hormigas para encontrar las trayectorias desde la colonia hasta el alimento.

3) Inteligencia Artificial (IA): Capacidad de razonar de un agente no vivo. Se utilizó este término en 1956 como "La ciencia e ingenio de hacer máquinas inteligentes", especialmente programas de cómputo inteligentes.

4) Metaheurística: Método heurístico para resolver un tipo de problema computacional general, usando los parámetros dados por el usuario sobre unos procedimientos genéricos y abstractos de una manera que se espera eficiente. Se puede decir, que es llegar a una conclusión más avanzada en lo futuro de lo que se está haciendo en el presente con los mismos datos que se tiene, es como adelantarse al futuro. Eso se logra con algún tipo de programa o pensamiento definido, y a eso se le llama heurística.

5) Inteligencia del enjambre (Swarm Intelligence): Rama de la Inteligencia artificial que se basa en el comportamiento colectivo de sistemas descentralizados y auto-organizados. Estos sistemas están constituidos típicamente de agentes simples que interactúan entre ellos y con su ambiente. Los agentes siguen reglas simples y, aunque no existe una estructura de control que dictamine el comportamiento de cada uno de ellos, las interacciones locales entre los agentes conducen a la emergencia de un comportamiento global complejo. Por ejemplo; las colonias de hormigas, el alineamiento de las aves en vuelo, el comportamiento de rebaños, el crecimiento bacteriano y el comportamiento de cardúmenes.

6) VoIP (Voice Over Internet Protocol): Término que significa "voz sobre un protocolo de internet". Básicamente VoIP es un método por el cual tomando señales de audio analógicas del tipo de las que se escuchan cuando uno habla por teléfono se las transforma en datos digitales que pueden ser transmitidos a través de Internet hacia una dirección IP determinada.

7) Ant Colony Optimization (ACO): En ciencias de la computación y en las operaciones de investigación, el algoritmo de optimización colonia de hormigas es una técnica probabilística para solucionar problemas computacionales que permitan buscar los mejores caminos o rutas en grafos, aplicable al enrutamiento.

8) Optimización de enjambres de partículas (PSO): Método de optimización heurístico que fue descrito por Kennedy & Eberhart (1995), este método nace del comportamiento de los enjambres de insectos en la naturaleza. En concreto, el enjambre que se pone de ejemplo para explicar este método es uno de abejas, ya que las abejas a la hora de buscar polen buscan la región del espacio en la que existe más densidad de flores, ya que es ahí donde encontraran mayor cantidad. Este método ha sido llevado al campo de la computación en forma de algoritmo y se emplea en la actualidad en la optimización de distintos tipos de sistemas.

9) Peso o costo del enlace: El costo de una interfaz es inversamente proporcional al ancho de banda de la interfaz. Por lo tanto, cuanto mayor es el ancho de banda, menor es el costo. Cuanto más sobrecarga y retraso, mayor es el costo. Por lo tanto, una línea Ethernet de 10 Mb/s tiene un costo mayor que una línea Ethernet de 100 Mb/s.

Para el protocolo OSPF.

- La fórmula para calcular el costo es:
  Costo = BW de referencia / BW de la interfaz
- El ancho de banda de referencia predeterminado es 10^8 (100 000 000); por lo tanto, la fórmula es:
  Costo = 100 000 000 bps / BW de la interfaz en bps

Para el protocolo EIGRP

- Utiliza estos valores escalados para determinar la métrica total hacia la red:

  M = ([K1*BW + (K2*BW)/(256 - TR) + K3*RT]*[K5/(CO + K4)])*256

  M : Métrica
  BW : Ancho de Banda
  TR : Tráfico
  CO : Confiabilidad
  RT : Retardo

- Estos valores K deben utilizarse después de una planificación minuciosa. Los valores K no coincidentes impiden que se cree una relación de vecinos, lo que puede hacer que su red no pueda converger. Si el K5 = 0, la fórmula reduce a

  *M = ([k1 * BW + (k2 * BW)/(256 - TR) + k3 * RT]) * 256.*

- Los valores predeterminados para K son: *K1 = 1, K2 = 0, K3 = 1, K4 = 0, K5 = 0.*

2.3. Hipótesis

Hipótesis General

Mediante el algoritmo de colonia de hormigas se determina la ruta óptima en el encaminamiento de flujo de datos a nivel de la capa de aplicación, ante los posibles cambios en la topología lógica de la red de comunicaciones.

Hipótesis Específicas:

a) Las detecciones de los cambios en la topología lógica de la red de manera dinámica se obtienen mediante el protocolo de comunicaciones en el momento oportuno.

b) La reconstrucción dinámica de la topología lógica de la red ante posibles cambios, debido a fallas en los enlaces, se obtiene implementando el algoritmo basado en colonia de hormigas.
c) El tiempo de convergencia de los algoritmos depende del tamaño físico y lógico de la red.
d) La ruta óptima se obtiene determinando el ancho de banda de los enlaces.

2.4. Variables

Variable independiente o explicativa (x)

X: Diseño e implementación del algoritmo de hormigas.

Indicadores:

X1: Cambios en la topología lógica de la red en tiempo oportuno y de manera dinámica.
X2: Fallas en los enlaces.
X3: Tamaño de la red.
X4: Ancho de banda de los enlaces.

Variable dependiente (Y)

Y: Búsqueda de la ruta óptima para el encaminamiento del flujo de datos.

Indicadores:

Y1: Encaminamiento adecuado del flujo de datos.
Y2: Topología lógica de la red reconstruida dinámicamente.
Y3: Tiempo de convergencia de los algoritmos.
Y4: Ruta óptima seleccionada adecuadamente.

CAPÍTULO III

METODOLOGÍA DE LA INVESTIGACIÓN

3.1.Diseño de investigación

El tipo de investigación es descriptivo y explicativo, según la clasificación de Hernández, Fernández , & Baptista (2010), ya que se diseñó e implementó, tanto el algoritmo de comunicaciones para detectar el cambio en la topología lógica de la red y para la reconstrucción de una nueva topología; así como el algoritmo basado en la colonia de hormigas (OCH) para la búsqueda de la ruta óptima para el flujo de datos, que implica un desarrollo descriptivo paso a paso, y la explicación de funcionamiento de dichos protocolos. El diseño de los algoritmos es cuasi experimental ya que su desarrollo y evaluación son realizados en un laboratorio que sirve para medir los diferentes parámetros.

3.2.Diseño del protocolo de comunicaciones

En la red conmutación utilizada en la investigación, los nodos y los enlaces se consideran de acuerdo con una realidad topográfica y además a la cantidad de tráfico que se cursa, algunos de ellos son nodos finales donde se puede instalar a los usuarios y nodos intermedios que pueden estar funcionando como nodos de tránsito o pueden ser nodos que tengan ambas funcionalidades. La topología física es del tipo mixta, una combinación de malla y estrella que permite agregar nuevos nodos dependiendo de la ubicación geográfica o del tráfico que se presenta en la red, el tráfico es transportado de un nodo inicial un nodo final a través de la ruta más adecuada, como se muestra en la Figura 1. Al aplicar el algoritmo propuesto, se pueden elegir las rutas más adecuadas y sobre todo se puede reestructurar las nuevas rutas ante algún fallo de un enlace o fallo de un nodo.

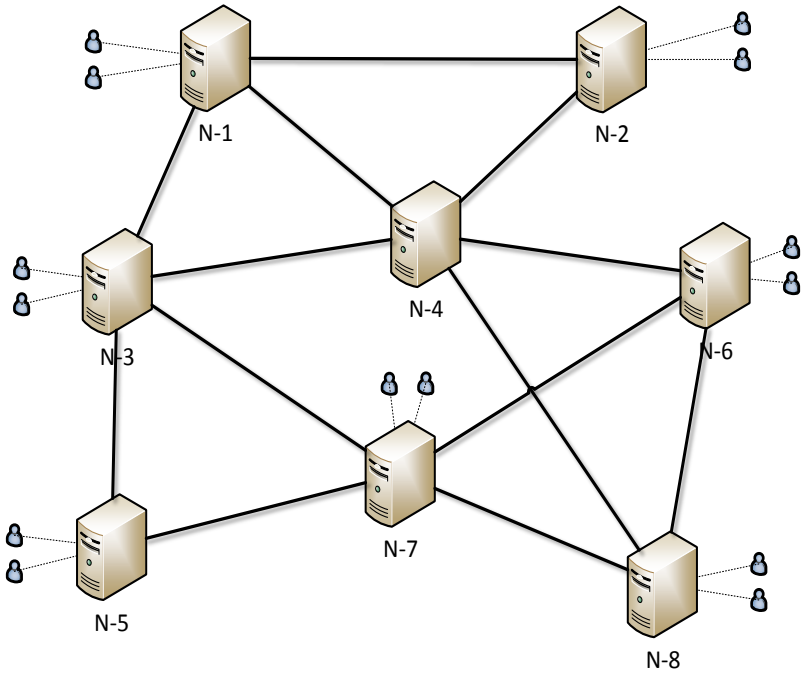


Figura 1 Topología física de la red

Fuente: Elaboración propia

Los nodos están supervisados por un servidor que se comunica con cada nodo a través de puertos UDP para solicitar información de cada nodo, la comunicación entre los nodos y el servidor es a través de un protocolo de comunicaciones y la topología lógica de señalización (control) sería la que se muestra en la Figura 2.

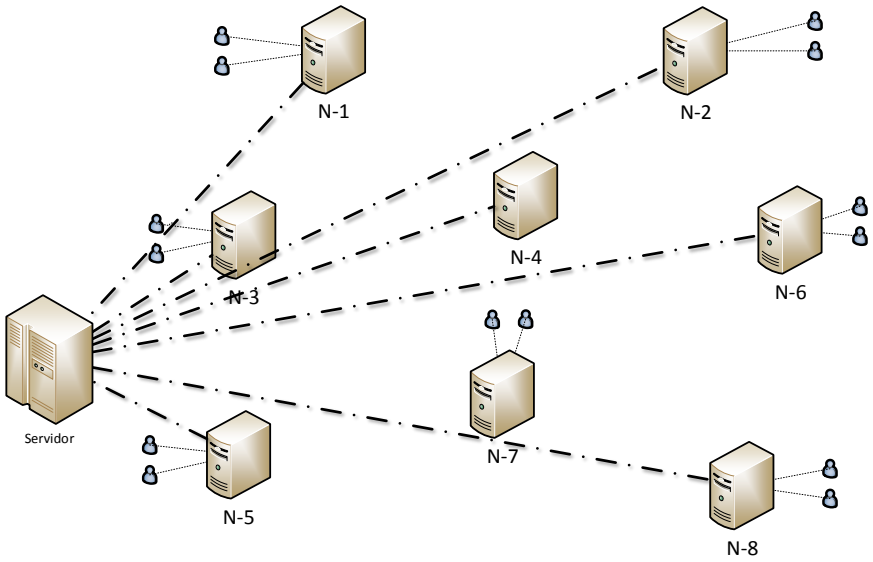


Figura 2. Servidor y la red de señalización.

Fuente: Elaboración propia

Para establecer el control de la comunicación entre el servidor y los nodos se ha realizado un protocolo de comunicaciones, la coordinación desde el servidor con los nodos se realiza mediante la asignación de puertos UDP y números IP, la asignación de los puertos utilizados se muestra en la Tabla 1.

Tabla 1. Nodos y sus parámetros para la comunicación con el servidor.

Fuente: Elaboración propia

ASIGNACIÓN DE PUERTOS UDP Y No IP A LOS NODOS		
Nodo	**Puerto UDP**	**No IP**
N1	5910	200.20.20.11
N2	5920	200.20.20.12
N3	5930	200.20.20.13
N4	5940	200.20.20.14
N5	5950	200.20.20.15
N6	5960	200.20.20.16
N7	5970	200.20.20.17
N8	5980	200.20.20.18

Además de estos datos es necesario ingresar la topología inicial de la red de comunicaciones con los siguientes parámetros (ver Tabla 2).

Tabla 2. Parámetros de la topología de la red.

Fuente: Elaboración propia

CUATRO PARÁMETROS DE LA TOPOLOGÍA	
No	**Parámetros**
1	Número de nodos
2	Número IP del nodo
3	Número de puerto de conexión
4	Número de Enlaces

3.3.Programas del protocolo

El protocolo de comunicaciones se ha implementado usando la programación java, para el servidor se ha desarrollado e implementado el programa "*servidor1.java"* (anexo C), este programa permite que el servidor se comunique con cada uno de los nodos de la topología de la red. Para los nodos se ha desarrollado e implementado el programa "*nodov1.java"* (Anexo D), este programa se instala en todos los nodos de la red. Toda comunicación entre el servidor y todos los nodos se realizan mediante estos programas.

Comunicación entre Servidor-Nodo

En el servidor se instalan los programas "*servidorv1.java*" y en los nodos el programa "*nodov1.java"*, el servidor genera un socket de comunicación (IP, Protocolo y Puerto) y lo envía en forma simultánea a cada nodo, luego cada nodo procesa estos datos y responde al servidor mediante un segmento de datos, cada segmento contiene los datos necesarios para que el servidor construya la topología de la red y pueda generar un archivo inu.txt actualizado, esta comunicación se encuentra abierta esperando un posible cambio del nodo o de un enlace, con la cual nuevamente el servidor actualiza el archivo *inu.txt*.

En la Figura 3, se muestra el resultado de ejecutar el programa en el nodo 1 de una topología básica de 4 nodos, un servidor (No) y tres nodos (N1, N2 y N3). El nodo N1 se comunica con el servidor (No) mediante el socket (No IP, puerto UDP y protocolo) y remite la siguiente información: Esperando conexión, No IP 200.20.20.11, número de conexión 3, puertos UDP abiertos 2510, 2640 y 2633, tiempo de espera limite 5 s, cerrando comunicación. Esta información se repite cada 5 segundos, tiempo que se puede ajustar de acuerdo con las necesidades de comunicación. En la Figura 4, se muestra el resultado de ejecutar el programa en el servidor nodo No, este programa nos reporta la siguiente información que ha sido enviada por los nodos: Fecha de activación, Numero de nodos conectados 4, No IP de cada nodo. Además, construye las siguientes matrices: Matriz de adyacencia de los nodos, Matriz de puertos para la comunicación entre los nodos y la matriz de pesos. Así mismo confirma la conexión con cada uno de los nodos. Ejemplo de la matriz de pesos.

Matriz de pesos =	∞	∞	∞	∞
	∞	∞	6	4
	∞	6	∞	2
	∞	4	2	∞

Resultado de ejecutar el programa en cada nodo/Estado de espera
[root@localhost hormigasnv1]#
[root@localhost hormigasnv1]# java nodov1
.....
esperando conexion .....
inicio de sesión
nodo: 200.20.20.11
nconecciones del nodo:1
numero de conexiones del nodo: 3.0
4
puerto abierto:2510
4
puerto abierto:2640
4
puerto abierto:2633
tiempo de espera limite 5 seg
cerrando conexión
.....
esperando conexión .....
inicio de sesión
nodo: 200.20.20.11
conexiones del nodo:1
numero de conexiones del nodo: 3.0
4
puerto abierto:2510
4
puerto abierto:2640
4
puerto abierto:2633
tiempo de espera limite 5 seg
cerrando conexión
.....
esperando conexion .....
inicio de sesión
nodo: 200.20.20.11
nconecciones del nodo:1
numero de conexiones del nodo: 3.0
4
puerto abierto:2510
4
puerto abierto:2640
4
puerto abierto:2633
tiempo de espera limite 5 seg
cerrando conexión
.....
esperando conexion .....

Figura 3. Resultado de instalar el programa en cada nodo.

Fuente: Elaboración propia

Programa Activado En el servidor / espera de conexión
[root@localhost interfaz]# java servidorv1 <inu.txt
.....
Mon Mar 09 16:20:30 PET 2015
.....
inicio de sesión el servidor
numero de nodos a conectar 4
*tiempor de reconocimiento de topologia: 18 miliseg
nodo ip 0 : 200.20.20.10
nodo ip 1 : 200.20.20.11
nodo ip 2 : 200.20.20.12
nodo ip 3 : 200.20.20.13
********** Matriz de adyacencia **********
0 1 1 1
1 0 1 1
1 1 0 1
1 1 1 0
********** Matriz de pesos **********
∞ ∞ ∞ ∞
∞ ∞ 6 4
∞ 6 ∞ 2
∞ 4 2 ∞
********** Matriz de puertos **********
cone 2510 2520 2530
2510 cone 2640 2633
2520 2640 cone 2643
2530 2633 2643 cone
esperando la respuesta del servidor
Respuesta: llego
Respuesta: llego
Respuesta: llego
Respuesta: llego
Respuesta: llego
El tiempo en establecer conexión con el nodo 2: 6 miliseg
Respuesta: llego
Respuesta: llego
El tiempo en establecer conexión con el nodo 3: 11 miliseg
Respuesta: llego
Respuesta: llego
El tiempo en establecer conexión con el nodo 1: 16 miliseg
[root@localhost interfaz]#

Figura 4. Resultado de instalar el programa en el servidor.

Fuente: Elaboración propia

Al activar el programa en el servidor se establecen las conexiones y se muestra el número IP asignado a cada nodo, la matriz de adyacencia de los nodos, matriz de pesos o costos entre los nodos, la matriz de puertos para la conexión entre nodos y la confirmación de conexión con los nodos, además se indica el tiempo de establecimiento de conexión entre el servidor y cada uno de los nodos.

Algoritmo de Hormigas

El algoritmo de hormigas se implementó en programación Java y se denomina *"detonador.java"* (anexo E), este programa se instala en el servidor y como datos de entrada requiere de la topología de la red que está en el archivo *inu.txt*. Una vez obtenida la topología de la Red con sus respectivos pesos en cada enlace, el algoritmo de hormigas busca las rutas más adecuadas en función de los pesos de cada enlace. La ruta elegida está en base al menor peso.

El servidor tiene "conocimiento" de toda la información de la red y los Pesos de cada enlace. A su vez, busca la ruta óptima para la emisión de los datos utilizando el algoritmo de colonia de hormigas propuesto. Para ello, ubica a todas las hormigas en un nodo en particular llamado "nodo fuente" o "nodo inicial", que en este caso es la estación base. A partir de allí, las hormigas inician su recorrido hasta que retornan al nodo inicial.

Matemáticamente, la red se traduce en un grafo completamente conexo, donde todos los vértices o nodos están conectados por un ruta o trayectoria, es decir para cualquier par de vértices, existe al menos un ruta o trayectoria. Un grafo conexo se define como $G_N = (N, A)$ donde *"N"* es el número de nodos o vértices y *"A"* es el conjunto de los enlaces o arcos. Para cada enlace o arco $(i, j) \in A$, se le asigna un valor E_{ij} llamado peso que está en función del ancho de banda y el tráfico correspondiente, E_{ij} representa el peso entre el nodo i y el nodo j. Se desea encontrar una ruta (enlace) de menor peso desde un nodo inicial hasta un nodo final, que recorra cada uno de los nodos del grafo G.

Si la ruta tiene mayor ancho de banda el peso es menor (inversamente proporcional) y si el peso o costo es menor la ruta será elegida por el algoritmo. Si el tráfico en el enlace es alto, implica que la ruta es de mayor peso o costo (directamente proporcional), por lo tanto, esta ruta será descartada por el algoritmo.

A continuación, se muestra un grafo conexo de cinco nodos $G_N = (N, A) = G_N = (5,6)$, mostrado en la Figura 5, para este caso N=5, el cual es el conjunto de nodos o vértices, N = {1, 2 ,3, 4, 5}. El número de enlaces A es igual a 6, el cual es el conjunto de conexiones entre los nodos A = {(1,2), (1,5), (2, 3), (2, 4), (3, 4), (4,5)}. Los pesos de conexión entre los nodos (i,j) están definidos por el conjunto $E_{i,j}$ = {($E_{1,2}$), ($E_{1,5}$), ($E_{2,3}$), ($E_{2,4}$), ($E_{3,4}$), ($E_{4,5}$)} = {3, 6, 8, 10, 1, 3}.

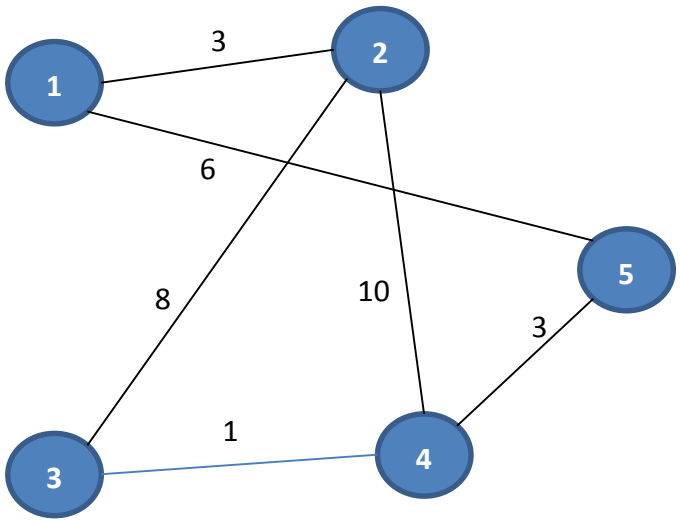


Figura 5. Grafo G(N, A) de 5 Nodos y 6 conexiones

Fuente: Elaboración propia

En la fase inicial se asignan, mediante un archivo de texto, ciertos valores necesarios para la implementación, los cuales representan a los parámetros del algoritmo y se muestran en la Tabla 3.

Tabla 3 Parámetros para la implementación del algoritmo de hormigas

Fuente: Elaboración propia

PARÁMETROS UTILIZADOS EN EL DISEÑO DEL ALGORITMO DE HORMIGAS	
Parámetro	**Descripción del parámetro**
n	Número de nodos de la red.
m	Número de hormigas usadas para la construcción de rutas.
nn	Cantidad de vecinos más cercanos que se calcula para cada nodo de la red.
α	Importancia de la tabla de feromonas en la selección de caminos.
β	Importancia de la heurística en la selección de caminos.
ρ	Tasa de evaporación global de feromona.
ξ	Tasa de evaporación local de feromona.

Adicionalmente se incorpora la topología de la red mediante el archivo de texto, es decir, la ubicación espacial de cada nodo representada como coordenadas (x, y) en un plano cartesiano, relacionadas con las posiciones reales de los nodos en el área geográfica. Con las coordenadas de los nodos se calculan la matriz de distancia y la matriz de vecinos más cercanos. A su vez se inicializan las matrices de rastros de feromonas y los valores de los pesos del nodo con valores predeterminados. De esta manera, la red de nodos está formada por un conjunto de "n" nodos en donde cada nodo se encuentra ubicado en un punto cartesiano (x, y) y contiene un valor de peso de la ruta e. La representación de la red en el algoritmo contiene tres matrices principales las cuales son: Los nodos, las distancias entre los nodos y la lista de vecinos más cercanos, ver Figura 6.

```
//Representación del nodo
  inicio
    real n              // identificación del nodo
    real(x,y)           // distancia del nodo x al nodo y
  fin
```

Figura 6. Estructura de datos para la representación de cada nodo en la red.

Fuente: Elaboración propia

En la Figura 7, se observa una matriz de distancia *dist* de tamaño n x n, ya que, es necesario conocer las distancias entre todos los nodos de la red. Los índices de la matriz coinciden con el identificador del nodo, es decir, el valor de *dist[2][3]* almacena la distancia entre el nodo 2 y el nodo 3.

```
//Representación del problema
Inicio
    Real dist[n][n]              // matriz de distancia entre nodos
    Entero lista nn[n][nn]// matriz de vecinos más cercanos de tamaño nn
    Estructura nodo[n]    // nodos de la red
fin
```

Figura 7. Estructuras de la matriz distancia.

Fuente: Elaboración propia

La matriz de vecinos más cercanos *lista_nn*, de tamaño nxn, contiene el identificador de todos los nodos vecinos de un nodo en particular, en un rango de distancia dado. Sea *dist[i][j]* la lista de las distancias del nodo *i* a todos los nodos *j*, con *j =1,….n* donde *i≠j*, la lista de vecinos más cercanos *lista_nn[i][z]* de un nodo N*i* con *z=1,…,nn,* contiene los primeros *nn* valores obtenidos al ordenar de mayor a menor las distancias de la lista *dist[i][j]*. La lista *nodo* de tamaño n, almacena todos los nodos de la red con la estructura también se muestra en la Figura 7.

Distancia entre los nodos

La distancia que existe entre el nodo emisor y el nodo receptor se determina mediante las coordenadas (*x*, *y*) de los nodos, la distancia $d(i,j)$ del nodo Ni al nodo Nj, se expresa por la distancia Euclidiana entre los puntos $(x_i, y_i)\ y\ (x_j, y_j)$, calculada según la fórmula. Rodrigo (2011).

$$d(i,j) = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \quad \text{……...……..…..} \ (1)$$

Generalmente, las distancias entre los nodos se almacenan en memoria como simples valores enteros. En este caso, se almacenan como valores reales debido a que las distancias entre los nodos pueden ser relativamente pequeñas o contener valores decimales dependiendo de la unidad de medida empleada.

Representación de las Hormigas

Cada hormiga es un agente computacional que, partiendo del nodo emisor, se encarga de construir una ruta entre todos los nodos, aplicando una regla de actualización de feromona en cada arco (i,j) que recorre. Para ello, debe almacenar la ruta parcial que ha construido, determinar el vecino de cada nodo, calcular y almacenar los valores de las soluciones que se generan, para cada hormiga, por lo tanto, es necesario almacenar en la lista los identificadores de los nodos que han sido visitados por la hormiga, ordenados en el arreglo según su aparición, ver Figura 8. La estructura de datos visitados se inicializa con cero (0) y se marca con el valor uno (1) el índice que coincide con el identificador del nodo que ya ha sido visitado. Esto se realiza con el fin de facilitar el proceso de búsqueda de la hormiga en la construcción de la ruta. La lista de *hormigas_hormiga* tiene tamaño m, dado en el archivo de inicialización de parámetros del algoritmo.

```
//Representación de las Hormigas
Estructura estructura_hormiga
inicio
entero ruta [n+1]          //memoria de las hormigas para almacenar las rutas parciales
caracter visitado [n]      //nodos visitados
fin
estructura_hormiga hormiga[m] //estructura de tipo estructura_hormiga
```

Figura 8. Estructura de los datos, rutas parciales y nodos visitados.

Fuente: Elaboración propia.

Rastros de Feromona

Para cada conexión (i, j) se le asigna un valor τ_{ij} que representa el rastro de feromona asociado con dicha conexión. Por ello, se almacenan los rastros de feromona en la matriz "*feromona"* de tamaño n^2, la cual es simétrica debido a que se asume que $\tau_{ij} = \tau_{ji}, \forall(i,j)$ (ver Figura 9).

```
//Representación de la matriz de feromona y heurística
Inicio
    real feromona n                    // matriz de feromonas
    real red_probabilidades [n][n]// total de información de feromona
    y arreglo de distancias
Fin
```

Figura 9. Matriz de feromona y el total de información heurística.

Fuente: Elaboración propia.

Información total de feromona y Heurística

Según Rodrigo Said, Henríquez Hernández (2011). En el proceso de construcción de la ruta, una hormiga en un sensor "*i"* elige al siguiente sensor "*j"* con una probabilidad de p_{ij}, según indica la fórmula metaheurística. Debido a que esta información es usada para cada hormiga, es necesario el uso de la matriz simétrica "*total"*, de tamaño n^2 para almacenar los valores resultantes de $[\tau_{ij}]^{\alpha}[\eta_{ij}]^{\beta}$.

Algoritmo de enrutamiento basado en Colonia de Hormigas

En esta parte se describen los pasos de implementación del Algoritmo de Colonia de Hormigas aplicado al problema de la Red de nodos, en el cual, la construcción de caminos es un aspecto clave para la búsqueda de soluciones. La ruta para el envío de la data se construye aplicando el siguiente proceso: Primero se envía un paquete hormiga desde el nodo fuente. Luego se usan los valores de feromonas y heurísticas para construir de manera probabilística una ruta, agregando nodos que no se han visitado aún, fusionando data de cada nodo hasta que todos los nodos sean visitados para así culminar el recorrido hacia la estación base o nodo inicial. Cada vez que la hormiga usa un arco para moverse de un nodo a otro, actualiza los rastros de feromona, reduciéndola para incrementar la exploración de caminos alternos. Otro paso importante es la actualización global de los rastros de feromona, en el cual se

lleva a cabo el depósito y evaporación de feromona en los arcos de la ruta obtenida por la mejor hormiga.

El diagrama de flujo del algoritmo de Colonia de Hormigas, mostrado en la Figura 10, presenta tres principales tareas. La primera consiste en el proceso de llevar a cabo la tarea de las hormigas para construir rutas de comunicación entre nodos, incluida la actualización local de feromona. Luego se determina la construcción de caminos eligiendo la mejor hormiga de la iteración. Finalmente, el algoritmo debe gestionar la actualización y evaporación de los rastros de feromona llevado a cabo por la mejor hormiga.

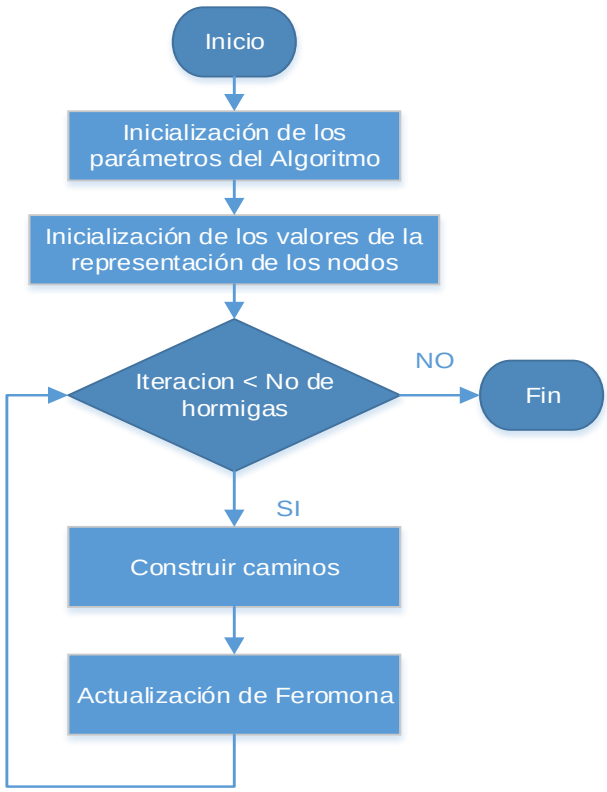


Figura 10. Diagrama de flujo del Algoritmo de Colonia de Hormigas.

Fuente: Elaboración propia.

Adicionalmente y fuera del ciclo principal, se lleva a cabo la inicialización de parámetros y estructuras de datos del algoritmo y la información de la red con los parámetros de la simulación detallada al inicio del capítulo.

Proceso del Algoritmo de Colonia de Hormigas

Como se mencionó anteriormente, las principales tareas del algoritmo de Colonia de Hormigas consisten en construir rutas de comunicación entre los nodos y gestionar la actualización de los rastros de feromona (ver Figura 11). Este proceso se repite mientras no se alcance el máximo de iteraciones.

```
Procedimiento ColoniadeHormigaparaRSI
        Inicialización
        Iteraciones ← 1
        Mientras (iteraciones <max_iteraciones) haga:
        ConstruirCaminos
        ActualizacionFeromona
        Fin del Mientras
Fin
```

Figura 11. Proceso del algoritmo de colonia de hormigas.

Fuente: Elaboración propia.

Construcción de los caminos

En esta fase, inicialmente se limpia la memoria de los caminos visitados por las hormigas. Luego, se ubican a las hormigas en el nodo inicial. Cada hormiga comienza una ruta desde dicho nodo, recorre todos los nodos y culmina su camino de regreso al nodo fuente. El seudocódigo del procedimiento se muestra en la Figura 12.

En cada paso, una hormiga situada en el nodo r, selecciona el siguiente nodo (Vecino Siguiente) usando la regla probabilística propuesta en la metaheurística, (Rodrigo Said, Henríquez Hernández: 2011).

$$P_{ij}^{k} = \begin{cases} \dfrac{[\tau_{ij}]^{\alpha}[\eta_{ij}]^{\beta}}{\sum_{l \in N_i^k}[\tau_{il}]^{\alpha}[\eta_{il}]^{\beta}} sij \in N_i^k \\ 0,\ en\ cualquier\ otro\ caso \end{cases} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (2)$$

Donde:

k: Es la hormiga situada en el nodo *i*.

P_y^k: Es la probabilidad en la cual la hormiga k elige para moverse desde un nodo i aun nodo j.

τ_{ij}: Es la abreviación de la tabla "feromona" donde se almacena la cantidad de rastros de feromona en la conexión que hay entre los nodos *i, j*.

$\eta_{ij} = 1/d_j$: Es una función heurística propuesta en la investigación.

$\alpha\ y\ \beta$: Parámetros que controlan la importancia relativa de los rastros de feromona versus la visibilidad.

N_i^k: Vecindad factible o vecinos más cercanos de una hormiga *k* cuando se encuentra en un nodo *i*, es decir el conjunto de nodos que una hormiga *k* no ha visitado aun (la probabilidad de elegir un sensor fuera de N_i^k es 0).

Así, la probabilidad de elegir un arco particular (i,j) se incrementa con el valor de información heurística η_{ij} y el valor de feromona asociado τ_{ij}. La probabilidad de selección es un balance entre visibilidad y la intensidad actual del rastro de feromona.

La visibilidad permite que los nodos con menor peso sean elegidos con mayor probabilidad, mientras que los rastros de feromona indican que ha habido mayor tráfico por esa ruta, lo cual se convierte en una mejor opción.

Procedimiento ConstruirCaminos
desde k= 1 **hasta** m
desde i=1 **hasta** n
hormiga[k].visitado[i]← falso
fin-desde
fin-desde
paso ←1
desde$k$ =1 **hasta** m
r ←estacionbase
hormiga[k].ruta[pasa]← r
hormiga[k].visitado[r]← verdadero
fin-desde
mientras (paso<n)
paso ← paso+1
desde k =1 **hasta** m
SeleccionaVecinoSiguiente($k, paso$
ActualizaciónLocalFeromona($k, paso$
Fin-desde
Fin-mientras
paso ←n
desde k= 1 **hasta** m
hormiga[k].ruta[n+1]← hormiga[k].ruta[r]
ActualizaciónLocalFeromona($k, paso$
Fin-desde

Figura 12. Proceso para la construcción de las rutas.

Fuente: Elaboración propia.

Actualización local de los rastros de feromona

Cuando una hormiga se mueve al siguiente nodo, se realiza la actualización local de rastros de feromona, propuesto por Dorigo & Stützle (2004). En el procedimiento *ActualizaciónLocalFeromona (k, paso)*, la hormiga calcula la cantidad de feromona que depositará en su recorrido de la siguiente manera:

$$\tau_{ij} \leftarrow (1-\xi)\tau_{ij} + \xi\tau_0 \quad \text{..... (3)}$$

El parámetro ξ, $0<\xi<1$, es la tasa de evaporación de feromona. Según los experimentos realizados en el trabajo de Dorigo & Stützle (2004), el valor de ξ sugerido es de 0,1. τ_0 , cuyo valor sugerido es de 0.25.

La implementación de la actualización local de feromonas es mostrada en la Figura 13, el cual es invocado por el procedimiento *ConstruirCaminos* mostrado en la Figura 12. En este procedimiento, cada vez que una hormiga usa un arco *(i,j)*, reduce su valor de feromona, lo cual hace que su ruta sea menos deseable por las siguientes hormigas, provocando así un comportamiento emergente de exploración de rutas.

```
Procedimiento ActualizacionLocalFeromona(k,paso)
j ←hormiga[k].ruta[paso]
h← hormiga[k].ruta[paso-1]
feromona[j][h]=(1-ξ)*feromona[j][h]+ξ*τ0
feromona[h][j]←feromona[j][h]
total[j][h]← exp feromona[j][h],α)*exp(1/d,β)
total[h][j]←total[j]h]
fin – procedimiento
```

Figura 13. Proceso para la actualización local de feromona.

Fuente: Elaboración propia.

Actualización global de los rastros de feromona

Luego de la actualización de los pesos, se selecciona la mejor hormiga de la iteración actual, denominada *mejor_hormiga_iteración*, la cual es la que posee el menor peso en su ruta. Si el peso de dicha hormiga es menor que cualquiera de las obtenidas en las iteraciones anteriores, entonces se designa como la mejor hormiga de todas las iteraciones y se le denomina *mj*. A esta hormiga se le permite agregar feromona al final de cada iteración, la cual está dada por la siguiente ecuación:

$$\tau_{ij} \leftarrow (1-\rho)\tau_{ij} + \rho\Delta, \forall (i,j) \in T^{mj} \quad \text{..... (4)}$$

Dónde: $\rho = 0.5$ y $\Delta = 0.4$

El parámetro *p* es la tasa de evaporación de feromona y debe estar entre 0.0 y 1.0 tal como se presenta en Dorigo & Stützle (2004). Este parámetro se usa para evitar la acumulación ilimitada de feromona y permite que el algoritmo pueda buscar nuevas soluciones explorando rutas no visitadas por las hormigas.

La implementación del procedimiento para la actualización global de feromonas se muestra en la Figura 14. Su función es reforzar la feromona en el camino conseguido por la mejor hormiga para aumentar la probabilidad de escoger esa ruta por otras hormigas. En líneas generales, los procedimientos mostrados anteriormente conforman la parte fundamental del algoritmo.

```
procedimiento ActualizacionGlobalFeromona(k)
 Δτ= 0.4
 para i =1 hasta n
      j ← hormiga[k].ruta[paso]
      h ← hormiga[k].ruta[paso + 1]
      feromona[j][h] = (1— p)* feromona[ j][h] + p * Δτ
      feromona[h][j] ←feromona[j ][h ]
      total[j][h] ←exp(feromona[j][h],α)*exp(1/ distancias,β)
      total[h][j] ← total[j][h]
 fin-para
fin –procedimiento
```

Figura 14. Proceso para la actualización global de feromona.

Fuente: Elaboración propia.

3.4.Población y muestra

La población está orientada al entorno empresarial que cuente con redes IP con un promedio de 300 computadoras, estas redes pueden estar conectadas mediante cable o de forma inalámbrica. Para la muestra se utilizaron 40 computadoras y 60 máquinas virtuales en VMware (programas que emulan un computador), en el

laboratorio LIFIEE de la Facultad de Ingeniería Eléctrica y Electrónica de la Universidad Nacional de Ingeniería. La muestra fue analizada en cinco topologías con la finalidad evaluar el desempeño del algoritmo desde una topología simple a una topología compleja. Para este caso se describen las estructuras de cinco topologías.

Topología 1: Para la topología 1 se consideró 7 Nodos y un servidor (8 nodos en total), además se ha asignado un peso para cada enlace como se muestra en la Figura 15. En cada computador del laboratorio se instaló una máquina virtual, la red IP utilizada fue la 200.20.20.0 con máscara 255.255.255.0, se asignó un peso a cada enlace y además se especificaron los puertos UDP para el experimento. El servidor se conectó con cada uno de los nodos, el peso asignado fue muy alto (999), por lo que el algoritmo de hormigas lo considera inaccesible.

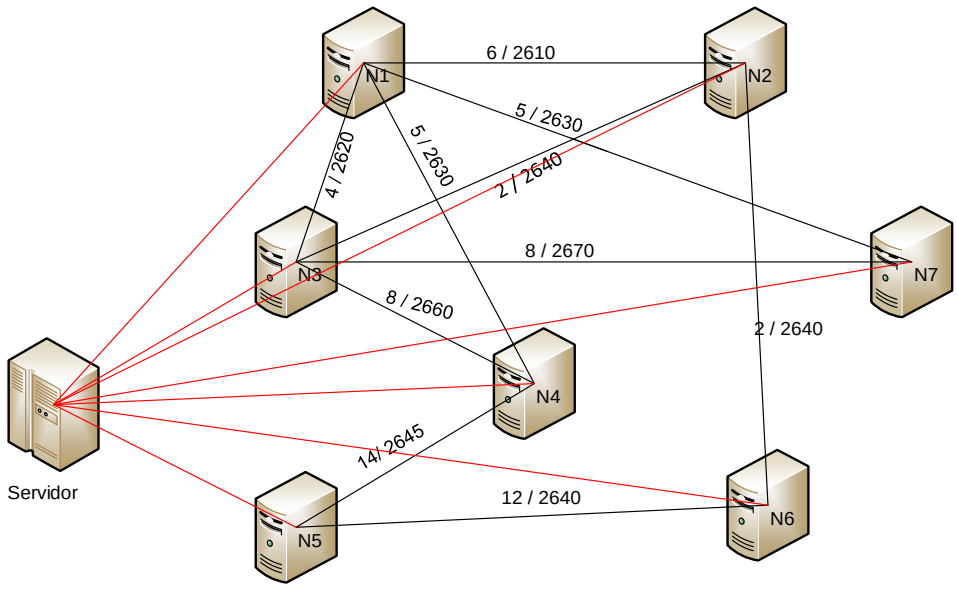


Figura 15. Topología 1 de la Red - Laboratorio LIFIEE-UNI.

Fuente: Elaboración propia.

En la Figura 16, se detalla la información de la topología 1 que está en el archivo inu.txt, esta tabla muestra el número de nodos totales 8 (7 nodos y 1 servidor), el número de enlaces 18 (7 enlaces con el servidor, 10 entre nodos y 1 de cierre con

valores fijados a cero) se tiene la dirección IP de cada nodo (máquina), también se muestra el enlace, el peso del enlace y el puerto UDP. Por ejemplo (4 5 14 2645), este número indica la conexión del nodo 4 con el nodo 5, el peso 14 y el puerto UDP 2645.

8 18 4 0
200.20.20.10 200.20.20.18 200.20.20.17 200.20.20.16 200.20.20.15 200.20.20.14 200.20.20.13 200.20.20.12
0 1 999 2510
0 2 999 2520
0 3 999 2530
0 4 999 2540
0 5 999 2520
0 6 999 2530
0 7 999 2540
1 2 6 2610
1 3 4 2620
4 5 14 2645
3 7 8 2670
1 4 5 2630
5 6 12 2640
2 3 2 2640
3 4 8 2660
1 7 5 2630
2 6 2 2640
0 0 0 0

Figura 16. Archivo inu.txt, que contiene la topología de la red.
Fuente: Elaboración propia.

Topología 2: Consta de 19 Nodos y un servidor, cada PC del laboratorio se ha instalado una máquina virtual, la red IP utilizada fue la 200.20.20.0 con mascara 255.255.255.0, se asignaron los pesos a los enlaces y además los puertos UDP para el experimento. Cabe indicar que el servidor se conecta con cada uno de los nodos, siendo el peso muy alto (999) de tal manera que el algoritmo de hormigas lo considera inaccesible y en la Figura 17, se muestra la conexión con líneas discontinuas.

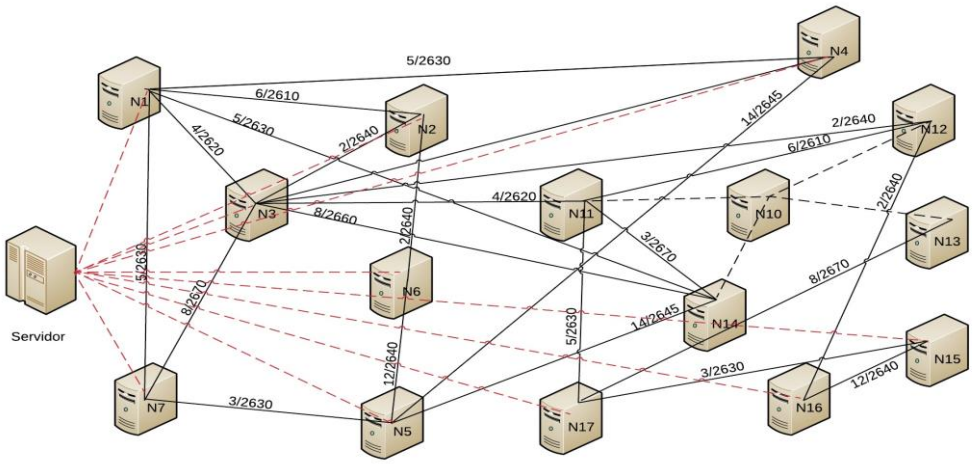


Figura 17. Topología 2 de la Red - Laboratorio LIFIEE-UNI.
Fuente: Elaboración propia.

En el Anexo A1 se tiene la topología 2, muestra el número de nodos totales 20 (19 nodos y 1 servidor), el número de enlaces 47 (19 con el servidor, 27 entre nodos y uno de cierre), luego se establece la dirección IP de cada nodo (máquina), también se detalla el enlace, el peso del enlace y el puerto UDP. Por ejemplo (5 6 12 2640), este número indica la conexión del nodo N5 con el nodo N6, el peso 12 y el puerto UDP 2640.

Topología 3: Consta de 40 Nodos (1 servidor y 39 Nodos), además se han asignado los pesos para cada uno de los enlaces como se muestra en la Figura 18. En cada computador del laboratorio se ha instalado una máquina virtual, la red IP utilizada es la 200.20.20.0 con mascara 255.255.255.0, se asignaron los pesos a los enlaces y los puertos UDP para el experimento. Cabe indicar que el servidor está conectado con cada uno de los nodos, pero el peso asignado es muy alto (999) de tal manera que el algoritmo de hormigas lo considera inaccesible, por tanto, se muestra la conexión con línea discontinua. Esta topología se ha implementado con el apoyo de máquinas virtuales VMware, de tal manera que no se necesitó muchas computadoras.

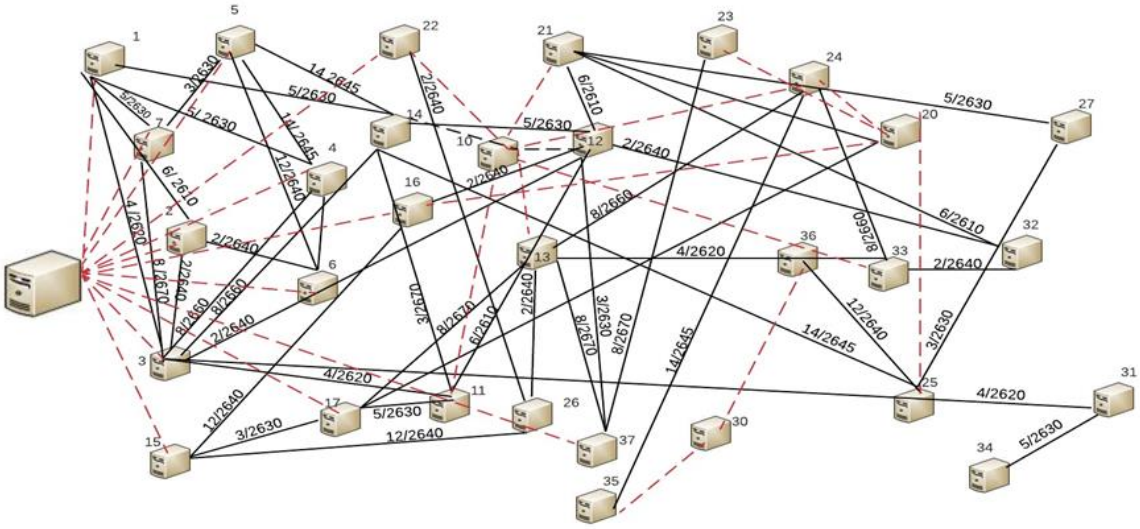


Figura 18. Topología 3 de la Red - Laboratorio LIFIEE-UNI.
Fuente: Elaboración propia.

En el Anexo A2 se detalla la información de la topología 3, en la cual se muestra el número de nodos totales 40 (39 nodos y 1 servidor), el número de enlaces 103 (39 enlaces con el servidor, 63 entre nodos y 1 de cierre), luego se asigna la dirección IP de cada nodo (máquina). Esta topología se ha implementado con el apoyo de máquinas virtuales VMware, de tal manera que no se han requerido muchos computadores.

Topología 4: En la Figura 19, se muestra una parte de este tipo de topología de red implementada, por razones de capacidad, esta topología ha sido implementada utilizando Máquinas Virtuales por cada computador en el laboratorio LIFIEE-UNI con 60 nodos (1 servidor y 59 nodos) y 81 enlaces, como se muestra en el Anexo A3, luego se asigna la dirección IP de cada nodo (máquina).

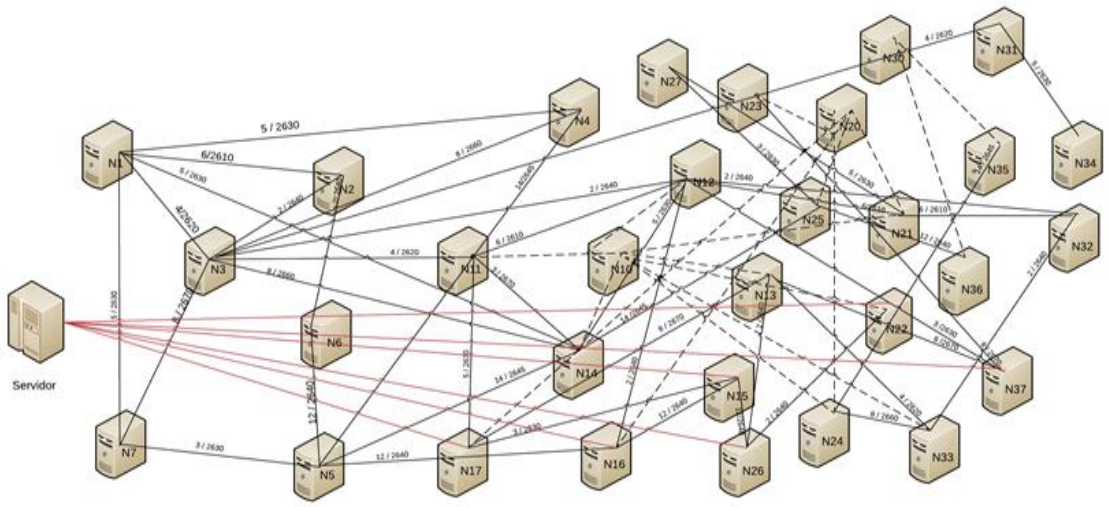


Figura 19. Topología 4 de la Red - Laboratorio LIFIEE-UNI.
Fuente: Elaboración propia.

Topología 5: Se ha implementado utilizando 3 a 4 máquinas virtuales por cada PC en el laboratorio LIFIEE-UNI. En esta topología solo se muestra una parte de la red implementada por razones de capacidad, debido a la gran cantidad de nodos (1 servidor y 99 nodos) y 91 enlaces. (ver Figura 20 y Anexo A4).

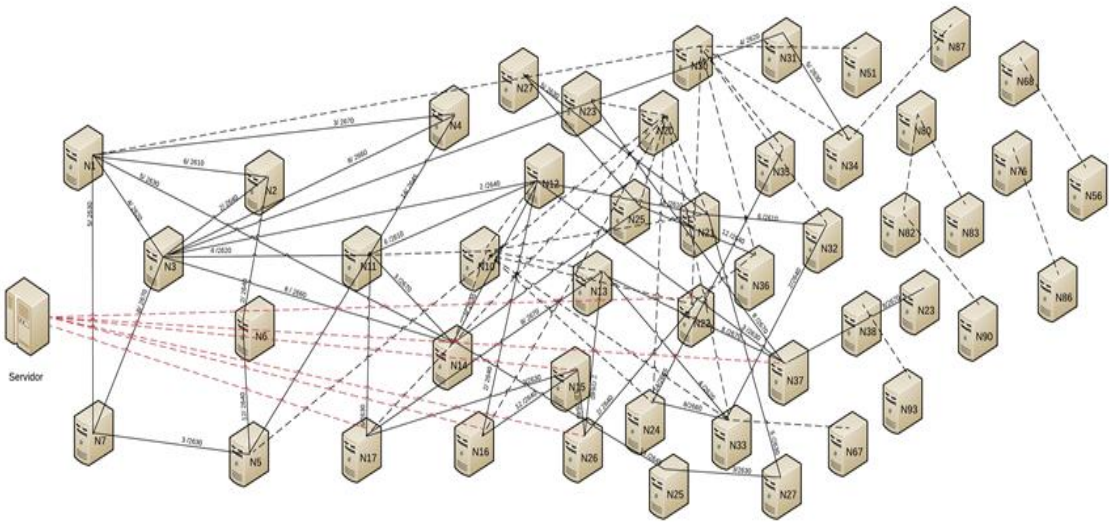


Figura 20. Topología 5 de la Red - Laboratorio LIFIEE-UNI
Fuente: Elaboración propia.

3.5.Técnicas e instrumentos

La técnica utilizada consiste en la observación experimental, porque procesa datos en condiciones controladas, particularmente porque es posible manipular la o las variables. La observación experimental es una técnica de investigación científica; se utilizó como instrumento la hoja o ficha de registro de datos. La técnica y los instrumentos utilizados para evaluar el experimento conforman una topología de red implementada en el laboratorio del LIFIEE-UNI. Los programas se han desarrollado usando el lenguaje JAVA, en máquinas virtuales instaladas en cada computador del laboratorio y la adquisición de los datos para su respectivo análisis se han obtenido mediante la herramienta estadística de Excel (ver Tablas 4 y 5).

Tabla 4. Instrumento para la estimación del tiempo de convergencia

Fuente: Elaboración propia.

Tiempo medio de convergencia		
Nodos	**Iteraciones**	**t-medio (mili-seg)**

Tabla 5. Instrumento para el análisis de rutas

Fuente: Elaboración propia.

Rutas desde el nodo Nx con destino al nodo Ny / No de saltos, nodo de tránsito y peso del enlace										
Saltos	**Ruta 1**		**Ruta 2**		**Ruta 3**		**. . .**		**Ruta n**	
	Nodo	**Peso**	**Nodo**	**Peso**	**Nodo**	**Peso**	**Nodo**	**Peso**	**Nodo**	**Peso**
1										
2										
3										
Peso total										

3.6.Procedimientos de recolección de datos

La recolección de datos se ha realizado basándose en una topología de red, con los parámetros de ancho de banda relacionados con sus respectivos pesos, al instarse los programas en los diferentes equipos o máquinas virtuales, se ejecuta el algoritmo de hormigas para determinar la ruta más adecuada desde un nodo inicial a los demás nodos finales, obteniéndose los resultados óptimos de las rutas seleccionadas por este algoritmo editado en un archivo de texto, con esta información el servidor trasmite a cada nodo la ruta a seguir para permitir el flujo de los datos y las topologías de las redes analizadas se describen en las Figuras 21 y 22.

TOPOLOGÍA 1: Se muestra en la Figura 21.

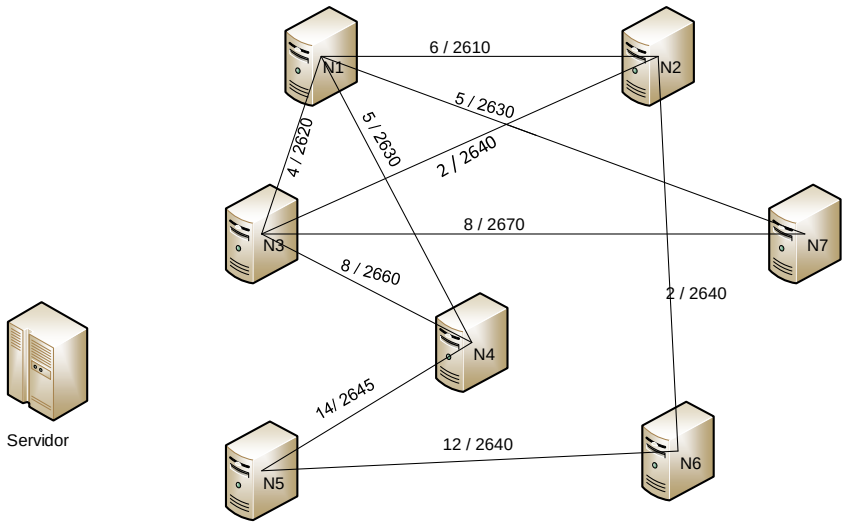


Figura 21. Topología 1, con 8 nodos y 10 enlaces.

Fuente: Elaboración propia.

Al desactivarse el enlace, entre el nodo N1 y el nodo N3, el protocolo de comunicaciones detecta el cambio de la topología de la red y transmite al servidor, actualizándose el archivo de la topología lógica de la red y el algoritmo de hormigas se ejecuta para encontrar las nuevas rutas del flujo de datos (ver Figura 22).

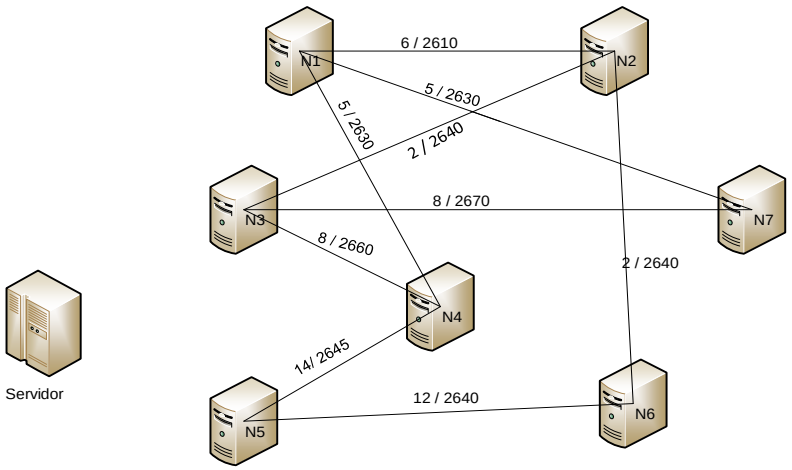


Figura 22. Nueva Topología 1, sin el enlace entre los nodos N1 y el nodo N3.

Fuente: Elaboración propia.

En la Figura 23, se muestra el resultado de esta acción, el protocolo de comunicaciones actualiza el archivo que contiene la topología lógica de la red, se observa que ya no está el enlace entre los nodos N1 y N3, también se observa el cambio del número de enlaces de 18 a 17, la cantidad de nodos no ha variado.

8 17 4 0
200.20.20.10 200.20.20.18 200.20.20.17 200.20.20.16 200.20.20.15 200.20.20.14 200.20.20.13 200.20.20.12
0 1 999 2510
0 2 999 2520
0 3 999 2530
0 4 999 2540
0 5 999 2520
0 6 999 2530
0 7 999 2540
1 2 6 2610
4 5 14 2645
3 7 8 2670
1 4 5 2630
5 6 12 2640
2 3 2 2640
3 4 8 2660
1 7 5 2630
2 6 2 2640
0 0 0 0

Figura 23 Archivo inu.txt, sin el enlace entre los nodos N1 y N3.

Fuente: Elaboración propia.

TOPOLOGÍA 2: Se muestra la Figura 24.

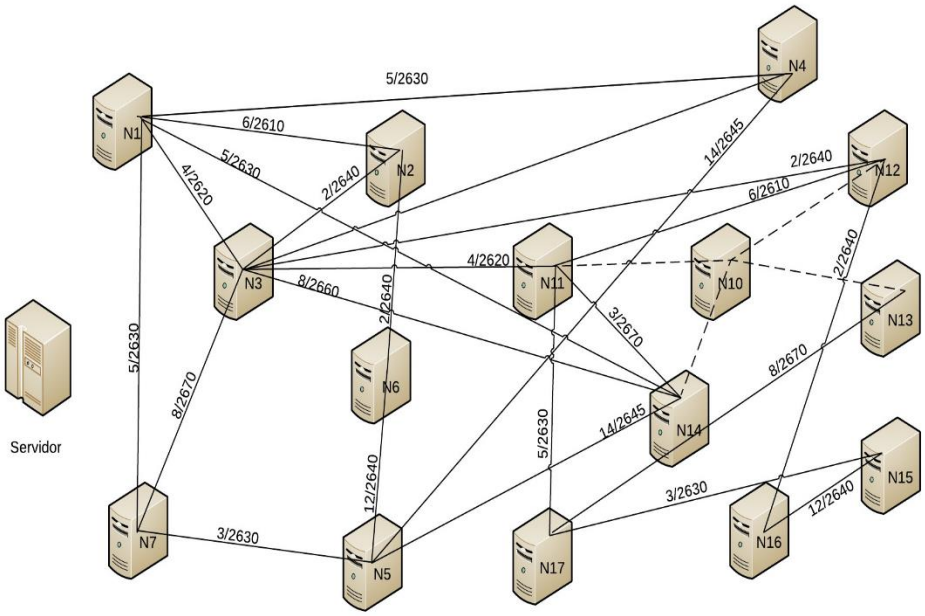


Figura 24: Topología 2, con 19 nodos y los pesos de los enlaces.

Fuente: Elaboración propia.

Primer caso: Al desactivarse el enlace, entre el nodo N3 y el nodo N4, el protocolo de comunicaciones detecta el cambio de la topología de la red y transmite al servidor, actualizándose el archivo de la topología lógica de la red y el algoritmo de hormigas se ejecuta para encontrar las nuevas rutas del flujo de datos (ver Figura 25).

En el Anexo B1 se muestra el resultado de esta acción, el protocolo de comunicaciones actualiza el archivo que contiene la topología lógica de la red, se observa que ya no está el enlace entre los nodos N3 y N4, también se observa el cambio del número de enlaces de 47 a 46, la cantidad de nodos no ha variado.

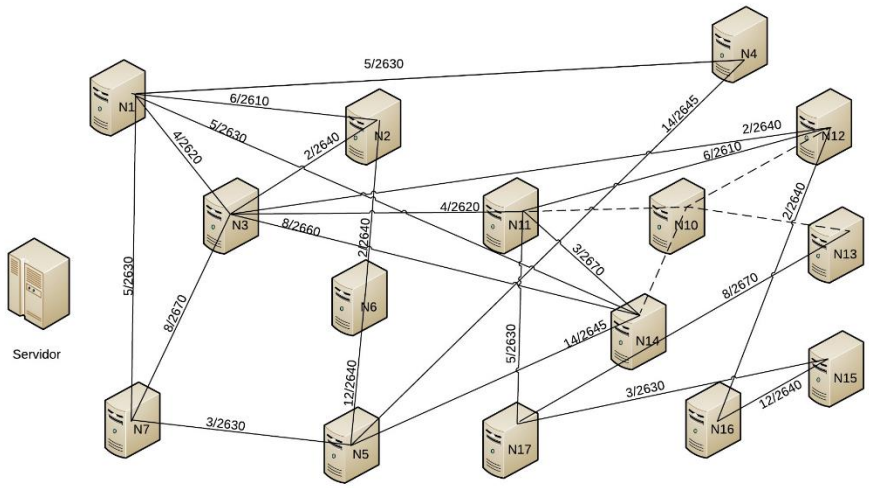
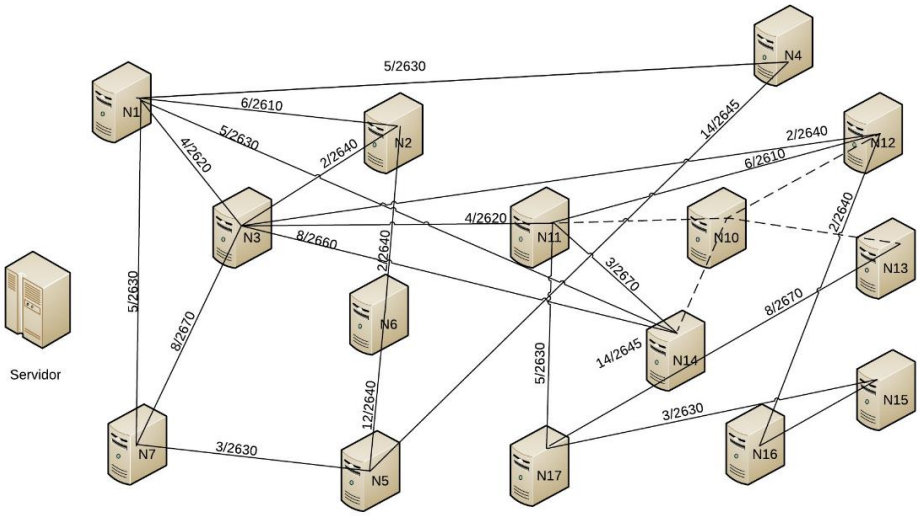


Figura 25: Nueva Topología 2, sin el enlace entre los nodos N3 y N4.

Fuente: Elaboración propia.

Segundo Caso: Al desactivarse el enlace, entre el nodo N14 y el nodo N5, el protocolo de comunicaciones detecta el cambio de la topología de la red y transmite al servidor, actualizándose el archivo de la topología lógica de la red y el algoritmo de hormigas se ejecuta para encontrar las nuevas rutas del flujo de datos y la nueva topología se muestra en a Figura 26.



.

Figura 26: Nueva Topología 2, sin el enlace entre los nodos N14 y N5

Fuente: Elaboración propia.

En el Anexo B2 se muestra el resultado de esta acción, el protocolo de comunicaciones actualiza el archivo que contiene la topología lógica de la red, se observa que ya no está el enlace entre los nodos N14 y N5, también se observa el cambio del número de enlaces de 46 a 45, la cantidad de nodos no ha variado

Tercer caso: Al desactivarse el enlace, entre el nodo N1 y el nodo N3, el protocolo de comunicaciones detecta el cambio de la topología de la red y transmite al servidor, actualizándose el archivo de la topología lógica de la red y el algoritmo de hormigas se ejecuta para encontrar las nuevas rutas del flujo de datos y la nueva topología se muestra en la Figura 27.

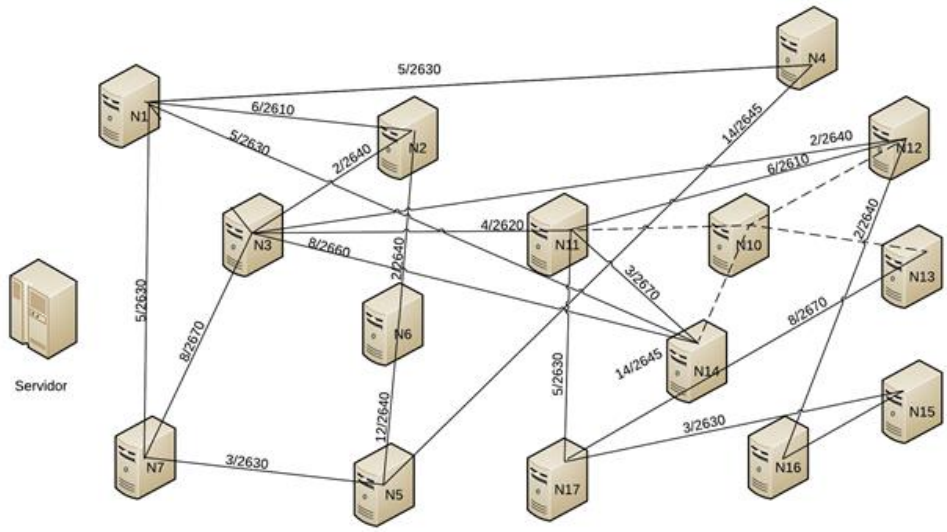


Figura 27. Nueva Topología 2, sin el enlace entre los nodos N1 y N3.
Fuente: Elaboración propia.

En el Anexo B3 se muestra el resultado de esta acción, el protocolo de comunicaciones actualiza el archivo que contiene la topología lógica de la red, se observa que ya no está el enlace entre los nodos N1 y N3, también se observa el cambio del número de enlaces de 45 a 44, la cantidad de nodos no ha variado

TOPOLOGÍA 3: Se muestra en la Figura 28.

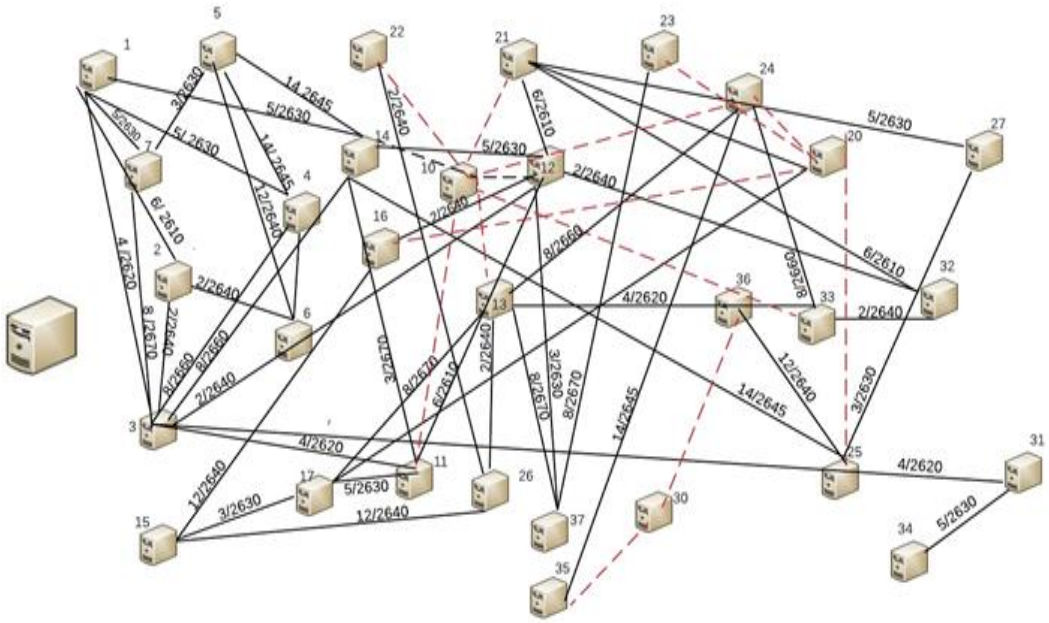


Figura 28. Topología 3, con 39 nodos y los pesos de los enlaces.

Fuente: Elaboración propia.

Al desactivar el enlace entre los nodos N3 y N14, y el enlace entre los nodos N12 y N16, el protocolo de comunicaciones detecta los cambios de la topología de la red, en base a los datos que transmite cada nodo hacia el servidor, se actualiza el archivo de la topología lógica de la red y se ejecuta el algoritmo de hormigas para encontrar las nuevas rutas para el flujo de datos (ver Figura 29). En la Figura 30 se muestra que el enlace del nodo N12 y el nodo N16 no está. En el Anexo B4 se muestra el resultado de esta acción, el protocolo de comunicaciones actualiza el archivo que contiene la nueva topología de la red, se observa además que ya no están los enlaces entre los nodos N3 y N14, y los nodos N12 y N16, también se observa el cambio de 103 enlaces a 101 enlaces, la cantidad de nodos no ha variado.

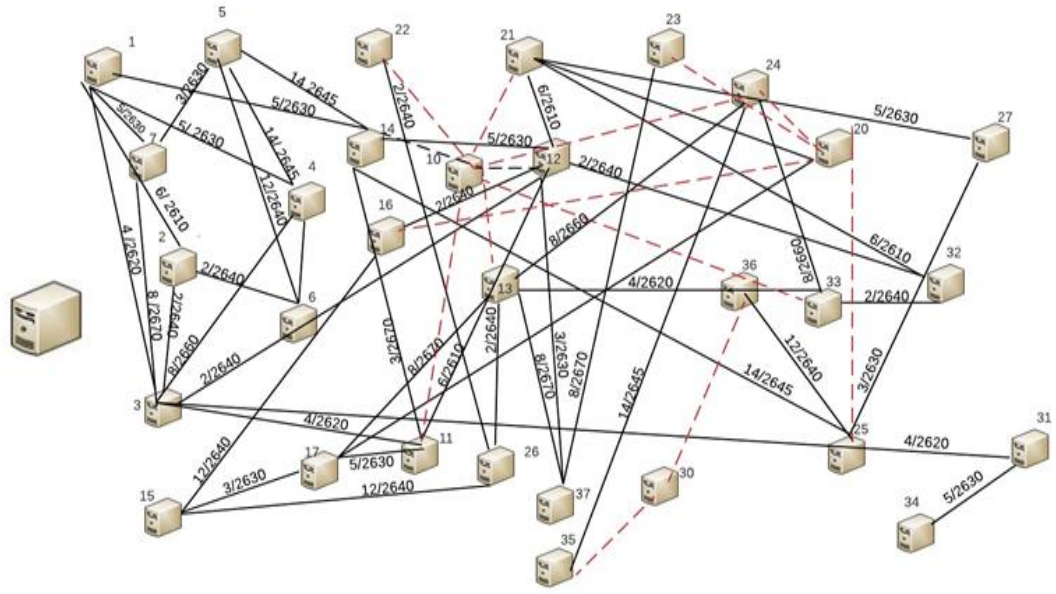


Figura 29: Nueva Topología 3, sin el enlace entre los nodos N3 y N14.

Fuente: Elaboración propia.

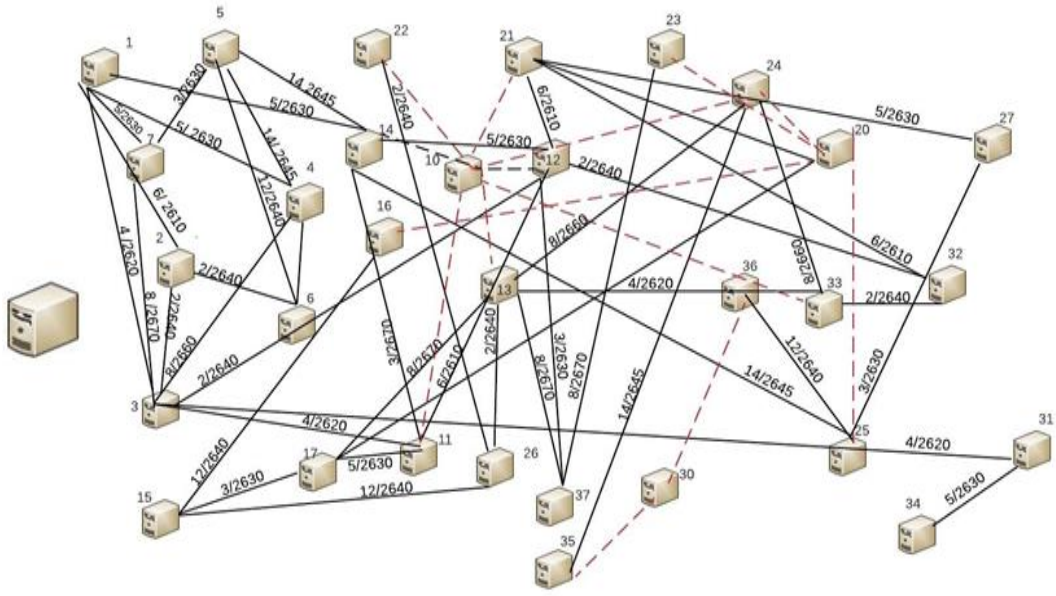


Figura 30. Nueva Topología 3, sin el enlace entre los nodos N12 y N16.

Fuente: Elaboración propia.

Topología 4: Se muestra en la Figura 31

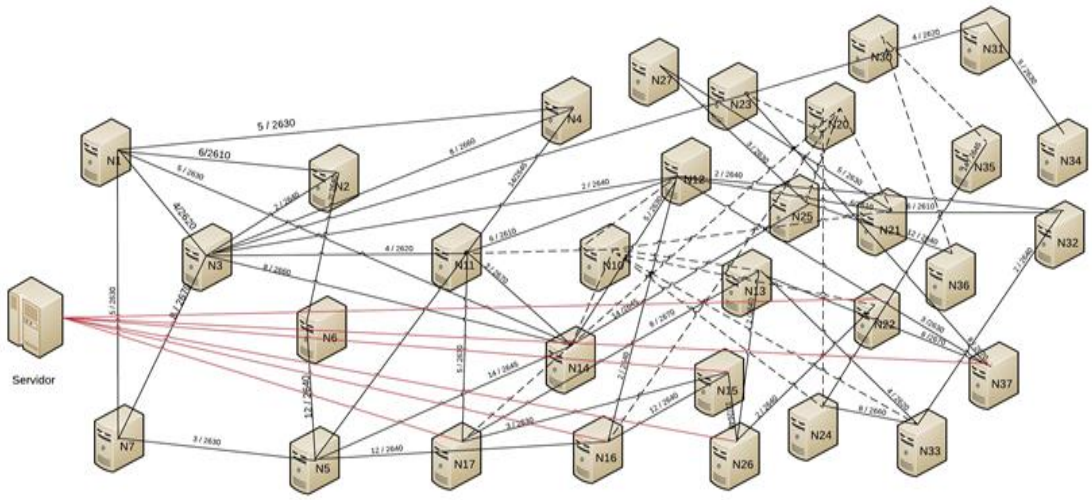


Figura 31. Topología 4, con 59 nodos y los pesos de los enlaces.

Fuente: Elaboración propia.

Al desactivar el enlace entre los nodos N5 y N6, y el enlace entre los nodos N11 y N12, el protocolo de comunicaciones detecta los cambios de la topología de la red en base a los datos que transmite cada nodo al servidor y se actualiza el archivo de la topología de la red, ejecutándose el algoritmo de hormigas para encontrar las nuevas rutas para el flujo de datos (ver Figura 32 y Figura 33). En el Anexo B5 se muestra el resultado de esta acción, el protocolo de comunicaciones recibe la información correspondiente de cada nodo y actualiza el archivo que contiene la nueva topología de red, se observa que ya no está el enlace entre el nodo N5 y el nodo N6, ni el enlace entre el nodo N11 y nodo N12, también se observa el cambio de 81 enlaces a 79 enlaces y que la cantidad de nodos no ha variado.

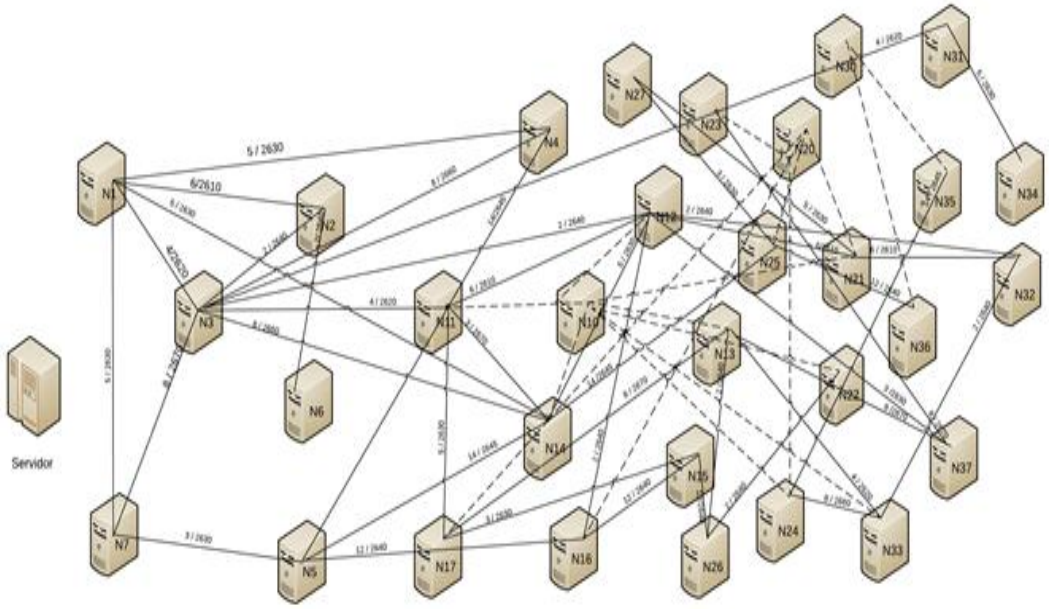


Figura 32. Nueva Topología 4, sin los enlaces entre los nodos N5 y N6.

Fuente: Elaboración propia.

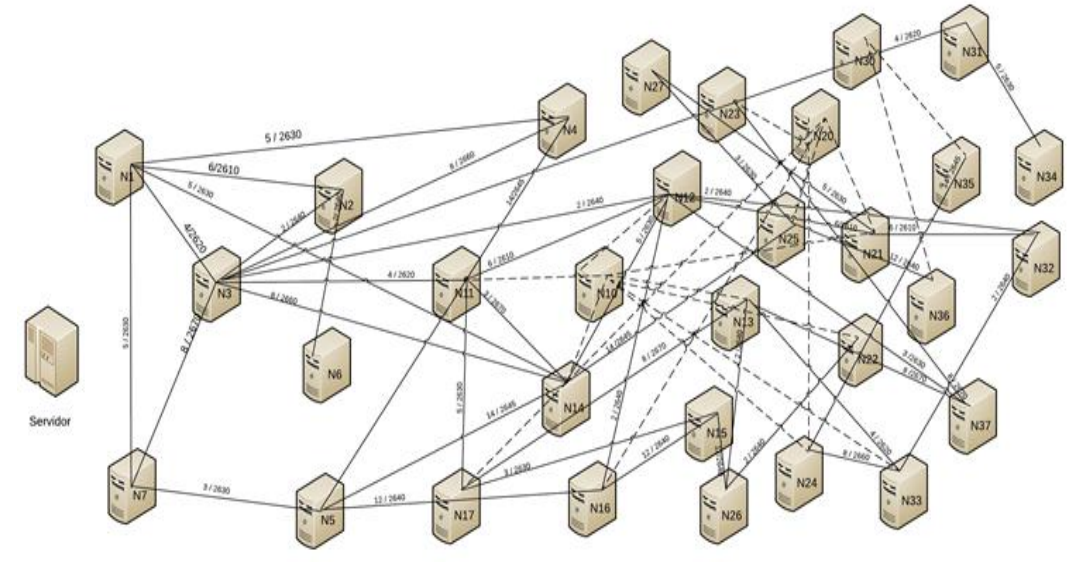


Figura 33. Nueva Topología 4, sin los enlaces entre los nodos N11 y N12.

Fuente: Elaboración propia.

Topología 5: Se muestra en la Figura 34

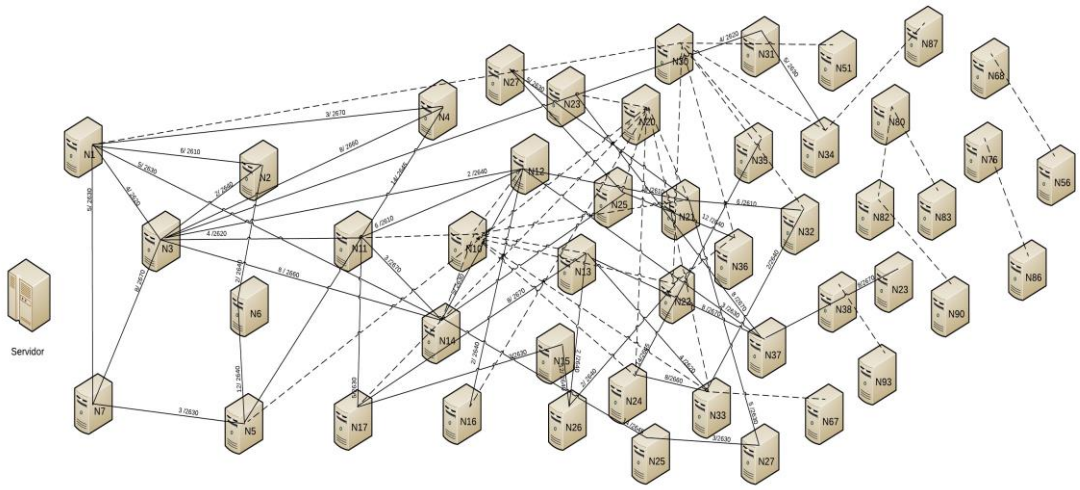


Figura 34. Topología 5, con 99 nodos y los pesos de los enlaces.

Fuente: Elaboración propia.

Al desactivarse el enlace entre el nodo N14 y el nodo N5, el protocolo de comunicaciones detecta los cambios en la topología de la red en base a los datos que remite cada nodo hacia el servidor, se actualiza el archivo de la topología de la red y se ejecuta el algoritmo de hormigas para que encuentre las nuevas rutas para el flujo de datos. En el Anexo B6 se muestra el resultado de esta acción, el protocolo de comunicaciones actualiza el archivo que contiene la topología de la red, se observa que ya no están los enlaces entre los nodos N14 y N5. Esta misma acción se repite al desactivarse el enlace entre los nodos N15 y N16, así como el cambio de 91 a 89 enlaces y la cantidad de nodos no ha variado (ver Figuras 35 y 36).

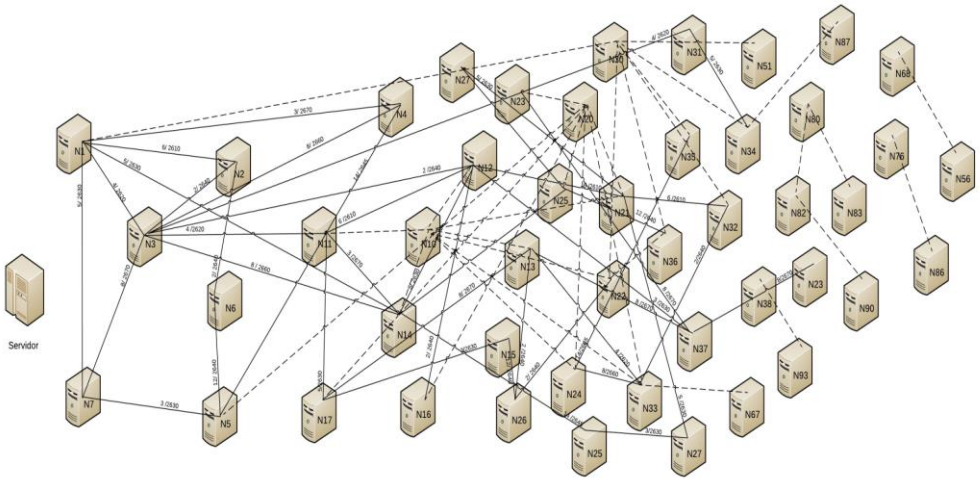


Figura 35. Nueva Topología 5, sin el enlace entre los nodos N14 y N5

Fuente: Elaboración propia.

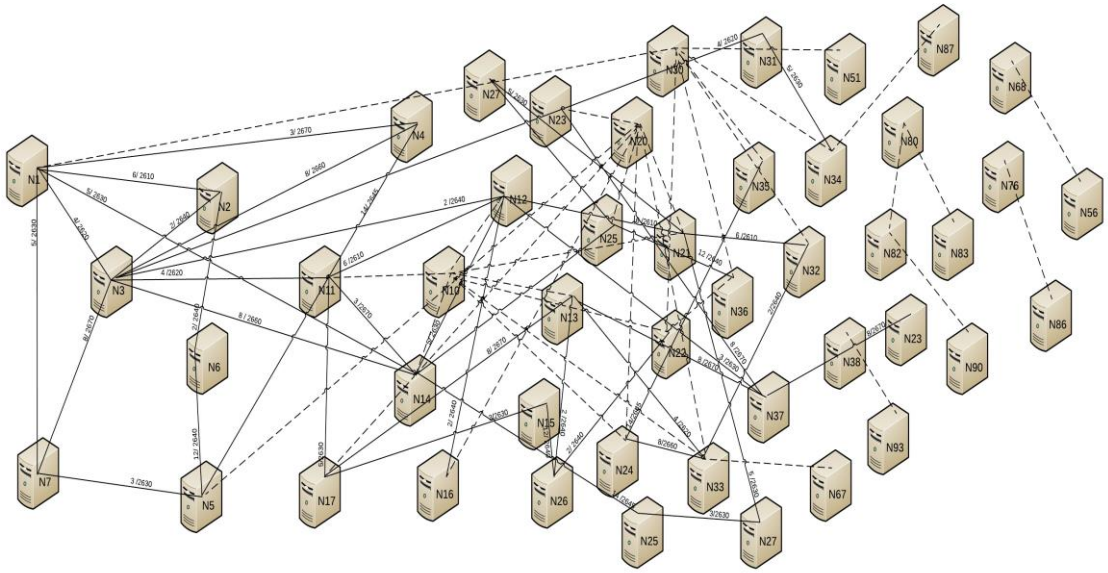


Figura 36. Nueva Topología 5, sin los enlaces entre los nodos N15 y N16.

Fuente: Elaboración propia.

CAPÍTULO IV

RESULTADOS Y ANÁLISIS DE LOS RESULTADOS

4.1.Resultados.

En esta sección se presentan los resultados experimentales obtenidos mediante el algoritmo propuesto. Cada vez que se ejecuta el algoritmo, se realiza la inicialización de las variables y los parámetros requeridos en ciertos valores, los cuales se muestran en la Tabla 6. El número de vecinos más cercanos "*nn"* se ha fijado en un valor de 15, ya que comúnmente el valor de "*nn"* está entre los valores de 15 y 40 Dorigo y Stützle (2004). Debido a que el valor de "*nn"* permite disminuir la complejidad al momento de elegir el siguiente nodo, el valor se fija al mínimo para aumentar la velocidad de ejecución del algoritmo. El valor de α se ha fijado en 0,85 y β a 0,75 para aumentar el peso en la tabla de feromonas de las hormigas que utiliza el método heurístico en la construcción de las rutas (Dorigo et., 1992; 1996). Adicionalmente el valor de ρ debe estar entre 0,0 y 1,0; el valor elegido es de ρ= 0,5 tal como es sugerido en los experimentos de Dorigo (1992, 1996). Por otra parte, el valor de ξ es de 0,1. Para todas las topologías se ha considerado 50 hormigas (m=50).

Tabla 6. Valores de los parámetros del algoritmo OCH

Fuente: Elaboración propia.

Parámetros y sus valores iniciales	
Parámetro	**Valor**
Número de vecinos más cercanos *nn*	15
Influencia relativa de la información heurística α	0,85
Influencia relativa de los rastros de feromona β	0,75
Factor de evaporación de feromona global ρ	0,50
Factor de evaporación de feromona local ξ	0,10

Los nodos se colocaron de forma aleatoria en un espacio bidimensional. En la primera fase de la experimentación se usaron 08 nodos. La disposición de los nodos se muestra en la siguiente Figura 37.

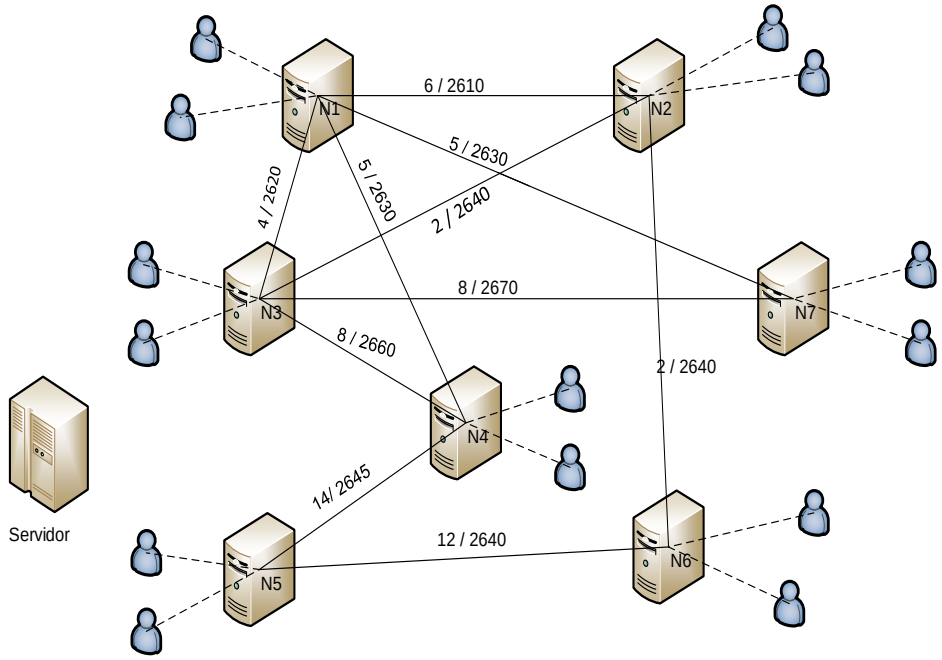


Figura 37. Topología de la red analizada.

Fuente: Elaboración propia.

4.2.Resultado del tiempo de convergencia

Topología 1: En la Tabla 7, se muestran los resultados de las mediciones del tiempo medio de convergencia del algoritmo, para una topología de 8 nodos y la cantidad de iteraciones. Se han realizado pruebas de 11, 21, 31, 41 y 51 iteraciones.

Tabla 7. Topología 1, tiempo promedio de convergencia.

Fuente: Elaboración propia.

No de Iteraciones y el tiempo de convergencia		
Nodos	**Iteraciones**	**t-Medio (mseg)**
8	11	9,18181818
8	21	9,04761905
8	31	10,19354840
8	41	8,26829268
8	51	9,25490196
	Suma total	45,946180270
	Promedio	9,189236050

Topología 2: En Tabla 8, se muestran los resultados de la medición del tiempo medio de convergencia del algoritmo, para una topología de 20 nodos y la cantidad de iteraciones. Se han realizado pruebas de 11, 21, 31, 41 y 51 iteraciones.

Tabla 8. Topología 2, tiempo promedio de convergencia.

Fuente: Elaboración propia.

No de Iteraciones y el tiempo de convergencia		
Nodos	**Iteraciones**	**t-Medio (mseg)**
20	11	12.90909
20	21	9.80952
20	31	9.54839
20	41	11.17073
20	51	8.05882
	Suma total	51.49655
	Promedio	10.29931

Topología 3: En la Tabla 9, se muestran los resultados de la medición del tiempo medio de convergencia del algoritmo, para una topología de 40 nodos y la cantidad de iteraciones. Se han realizado pruebas de 11, 21, 31, 41 y 51 iteraciones.

Tabla 9. Topología 3, tiempo promedio de convergencia.

Fuente: Elaboración propia.

No de Iteraciones y el tiempo de convergencia		
Nodos	**Iteraciones**	**t-Medio (mseg)**
40	11	18.09091
40	21	18.61905
40	31	14.51613
40	41	14.80488
40	51	16.70588
	Suma total	82.73685
	Promedio	16.54737

Topología 4: En la Tabla 10, se muestran los resultados de la medición del tiempo medio de convergencia del algoritmo, para una topología de 60 nodos y la cantidad de iteraciones. Se han realizado pruebas de 11, 21, 31, 41 y 51 iteraciones.

Tabla 10. Topología 4, tiempo promedio de convergencia.

Fuente: Elaboración propia.

No de Iteraciones y el tiempo de convergencia		
Nodos	**Iteraciones**	**t-Medio (mseg)**
60	11	25.00000
60	21	22.57143
60	31	24.09677
60	41	25.60976
60	51	38.49020
	Suma total	135.76815
	Promedio	27.15363

Topología 5: En la Tabla 11, se muestran los resultados de la medición del tiempo medio de convergencia del algoritmo, para una topología de 100 nodos y la cantidad de iteraciones. Se han realizado pruebas de 11, 21, 31, 41 y 51 iteraciones.

Tabla 11. Topología 5, tiempo promedio de convergencia.

Fuente: Elaboración propia.

No de Iteraciones y el tiempo de convergencia		
Nodos	**Iteraciones**	**t-Medio (mseg)**
100	11	29.00000
100	21	29.50000
100	31	29.50000
100	41	31.70000
100	51	32.10000
	Suma total	149.22219
	Promedio	29.84444

4.3. Resultado del tiempo de convergencia vs el número de nodos

En la Tabla 12 se muestra el resumen de los resultados del número de nodos y los valores del tiempo medio de convergencia del algoritmo de hormigas.

Tabla 12. Topologías y el tiempo promedio de convergencia.

Fuente: Elaboración propia.

Topologías y tiempo promedio de convergencia	
Nodos	**t-medio (mseg)**
8	9.18923605
20	10.29931000
40	16.54737000
60	27.15363000
100	29.84444000

4.4. Resultados de la elección rutas en diferentes topologías.

Topología 1

- ***Conexión entre el nodo N1 y el nodo N5.***

  En esta topología se muestran las diferentes rutas existentes entre el nodo N1 y el nodo N5. En la tabla 13 se muestra la existencia de 5 rutas entre estos dos nodos, además se indica el peso o costo de cada ruta, y el costo total por cada ruta. Ejemplo: La primera ruta para conectar el nodo N1 con el nodo N5, se establece de la siguiente manera: La conexión del nodo N1 al nodo N3 con peso 4, la conexión del nodo N3 al nodo N4 con peso 8 y la conexión del nodo N4 al nodo N5 con peso 14, luego se suman los pesos de esta ruta del nodo N1 al nodo N5 y se obtiene el peso total de la ruta igual a 26.

Tabla 13. Rutas desde el Nodo N1 al nodo N5.

Fuente: Elaboración propia.

Análisis de las rutas del nodo N1 al nodo N5, saltos, nodos y pesos										
Saltos	**Ruta 1**		**Ruta 2**		**Ruta 3**		**Ruta 4**		**Ruta 5**	
	Nodo	**Peso**	**Nodo**	**Peso**	**Nodo**	**Peso**	**Nodo**	**Peso**	**Nodo**	**Peso**
0	1		1		1		1		1	
1	3	4	2	6	7	5	2	6	4	5
2	4	8	3	2	3	8	6	2	5	14
3	5	14	4	8	4	8	5	12		
4			5	14	5	14				
Peso total		26		30		35		20		19

- ***Conexión entre el nodo N2 y el nodo N4.***

  En este caso se muestran las diferentes rutas existentes entre el nodo N2 y el nodo N4, así como los pesos por cada enlace y el peso total de la ruta, en total se establecen 5 rutas para la conexión entre estos dos nodos (ver tabla 14).

Tabla 14. Rutas desde el nodo N2 al nodo N4.

Fuente: Elaboración propia.

Análisis de las Rutas del nodo N2 con destino al nodo N4, saltos, nodos y pesos										
Saltos	**Ruta 1**		**Ruta 2**		**Ruta 3**		**Ruta 4**		**Ruta 5**	
	Nodo	**Peso**	**Nodo**	**Peso**	**Nodo**	**Peso**	**Nodo**	**Peso**	**Nodo**	**Peso**
0	2		2		2		2		2	
1	6	2	1	6	1	6	3	2	1	6
2	5	12	7	5	3	4	4	8	4	5
3	4	14	3	8	4	8				
4			4	8						
Peso total		28		27		18		10		11

- ***Conexión entre el nodo N3 y el nodo N6.***

  En este caso se muestran las diferentes rutas existentes entre el nodo N3 y el nodo N6. Se establecen 6 rutas para la conexión entre estos dos nodos (ver tabla 15).

Tabla 15. Peso de cada ruta desde el nodo N3 al nodo N6.

Fuente: Elaboración propia.

Análisis de las Rutas del nodo N3 con destino al nodo N6, saltos, nodos y pesos												
Saltos	**Ruta 1**		**Ruta 2**		**Ruta 3**		**Ruta 4**		**Ruta 5**		**Ruta 6**	
	Nodo	**Peso**	**Nodo**	**Peso**	**Nodo**	**Peso**	**Nodo**	**Peso**	**Nodo**	**Peso**	**Nodo**	**Peso**
0	3		3		3		3		3		3	
1	4	8	1	4	2	2	1	4	7	8	7	8
2	5	14	2	6	6	2	4	5	1	5	1	5
3	6	12	6	2			5	14	4	5	2	6
4							6	12	5	14	6	2
5									6	12		
Peso total		34		12		4		35		44		21

- ***Conexión entre el nodo N1 y el nodo N6.***

  En este caso se muestran las diferentes rutas existentes entre el nodo N1 y el nodo N6. Para este caso se establecen 5 rutas para la conexión entre estos nodos (ver tabla 16).

Tabla 16. Peso de cada ruta desde el nodo N1 al nodo N6.

Fuente: Elaboración propia.

Análisis de las Rutas del nodo N1 con destino al nodo N6, saltos, nodos y pesos										
Saltos	**Ruta 1**		**Ruta 2**		**Ruta 3**		**Ruta 4**		**Ruta 5**	
	Nodo	**Peso**	**Nodo**	**Peso**	**Nodo**	**Peso**	**Nodo**	**Peso**	**Nodo**	**Peso**
0	1		1		1		1		1	
1	2	6	4	5	3	4	7	5	2	6
2	6	2	5	14	4	8	3	8	3	2
3			6	12	5	14	4	8	4	8
4					6	12	5	14	5	14
5							6	12	6	12
Peso total		8		31		38		47		42

Topología 2

- ***Conexión entre el nodo N1 y el nodo N5***

  En esta topología de 20 nodos se muestran las diferentes rutas existentes para la conexión entre el nodo N1 y el nodo N5. Para este caso se establecen 5 rutas (ver tabla 17).

Tabla 17. Peso de cada ruta desde el nodo N1 al nodo N5.

Fuente: Elaboración propia.

Análisis de las Rutas del nodo N1 con destino al nodo N5, saltos, nodos y pesos										
Saltos	**Ruta 1**		**Ruta 2**		**Ruta 3**		**Ruta 4**		**Ruta 5**	
	Nodo	**Peso**	**Nodo**	**Peso**	**Nodo**	**Peso**	**Nodo**	**Peso**	**Nodo**	**Peso**
0	1		1		1		1		1	
1	7	5	14	5	3	4	2	6	4	5
2	5	3	5	14	14	8	6	2	5	14
3					5	14	5	12		
Peso total		8		19		26		20		19

- ***Conexión entre el nodo N2 y el nodo N7***

  En este caso se analizan las diferentes rutas existentes entre el nodo N2 y el nodo N7. Para este caso se establecen 7 rutas para la conexión entre estos nodos (ver Tabla 18).

Tabla 18. Peso de cada ruta desde el nodo N2 al nodo N7.

Fuente: Elaboración propia.

Análisis de las Rutas del nodo N2 con destino al nodo N7, saltos, nodos y pesos														
Saltos	**Ruta 1**		**Ruta 2**		**Ruta 3**		**Ruta 4**		**Ruta 5**		**Ruta 6**		**Ruta 7**	
	Nodo	**Peso**	**Nodo**	**Peso**	**Nodo**	**Peso**	**Nodo**	**Peso**	**Nodo**	**Peso**	**Nodo**	**Peso**	**Nodo**	**Peso**
0	2		2		2		2		2		2		2	
1	1	6	6	2	3	2	3	2	1	6	1	6	1	6
2	7	5	5	12	7	8	1	4	3	4	14	5	14	5
3			7	3			7	5	7	8	5	14	3	8
4											7	3	7	8
Peso total		11		17		10		11		18		28		27

- ***Conexión entre el nodo N1 y el nodo N12***

  En este caso se analizan las diferentes rutas existentes entre el nodo N1 y el nodo N12. Como la topología es más grande, entonces en la conexión entre algunos nodos se presentan mayor cantidad de rutas. Para este caso se establecen 10 rutas posibles entre estos dos nodos (ver Tabla 19).

Tabla 19. Peso de cada ruta desde el nodo N1 al nodo N12.

Fuente: Elaboración propia.

Análisis de las Rutas del nodo N1 con destino al nodo N12, saltos, nodos y pesos																				
Saltos	**Ruta 1**		**Ruta 2**		**Ruta 3**		**Ruta 4**		**Ruta 5**		**Ruta 6**		**Ruta 7**		**Ruta 8**		**Ruta 9**		**Ruta 10**	
	Nodo	**Peso**	**Nodo**	**Peso**	**Nodo**	**Peso**	**Nodo**	**Peso**	**Nodo**	**Peso**	**Nodo**	**Peso**	**Nodo**	**Peso**	**Nodo**	**Peso**	**Nodo**	**Peso**	**Nodo**	**Peso**
0	1		1		1		1		1		1		1		1		1		1	
1	3	4	3	4	7	5	3	4	2	6	7	5	14	5	4	5	4	5	4	5
2	11	4	12	2	3	8	14	8	6	2	5	3	11	3	5	14	5	14	5	14
3	12	6			11	4	11	3	5	12	14	14	12	6	14	14	6	12	6	12
4					12	6	12	6	14	14	11	5			11	3	2	2	2	2
5									11	3	12	6			12	6	3	2	3	2
6									12	6							12	2	11	4
7																			12	6
Peso total		14		6		23		21		43		33		14		42		37		45

- ***Conexión entre el nodo N2 y el nodo N14***

  En este caso se analizan las diferentes rutas existentes entre el nodo N2 y el nodo N14. Para la conexión entre estos nodos se presentan nueve rutas posibles (ver tabla 20).

Tabla 20. Peso de cada ruta desde el Nodo N2 al nodo N14.

Fuente: Elaboración propia.

Análisis de las Rutas del nodo N2 con destino al nodo N14, saltos, nodos y pesos																		
Saltos	Ruta 1		Ruta 2		Ruta 3		Ruta 4		Ruta 5		Ruta 6		Ruta 7		Ruta 8		Ruta 9	
	Nodo	Peso	Nodo	Peso	Nodo	Peso	Nodo	Peso	Nodo	Peso	Nodo	Peso	Nodo	Peso	Nodo	Peso	Nodo	Peso
0	2		2		2		2		2		2		2		2		2	
1	3	2	6	2	3	2	3	2	1	6	1	6	1	6	3	2	1	6
2	14	2	5	12	11	4	4	8	3	4	14	5	7	5	7	8	4	5
3			14	14	14	3	5	14	14	8			5	3	5	3	5	14
4							14	14					14	14	14	14	14	14
Peso total		4		28		9		38		18		11		28		27		39

4.5.Análisis del tiempo de convergencia del algoritmo de hormigas

Topología 1:

De los datos obtenidos del tiempo de convergencia para 8 nodos y número de iteraciones 10, 20, 30,40 y 50, se obtiene la gráfica de la Figura 38.

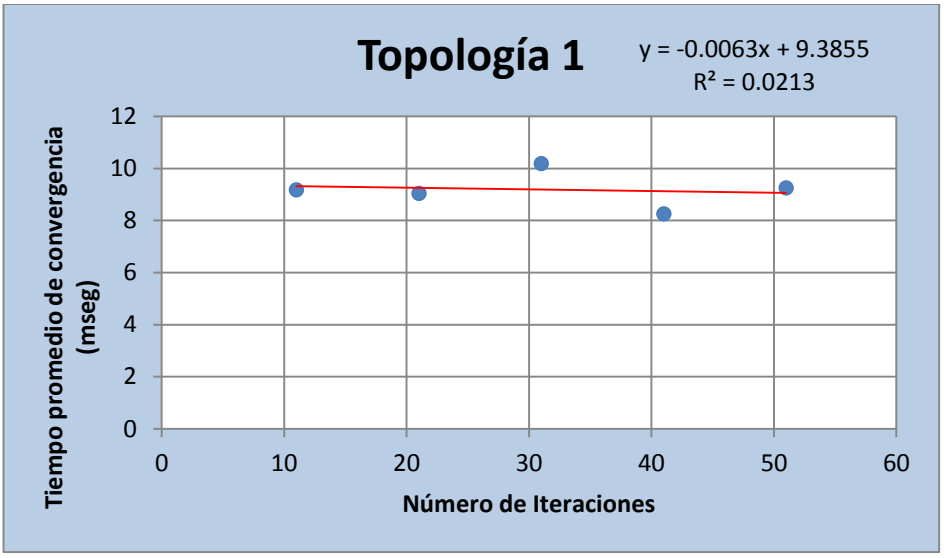


Figura 38. Tendencia lineal de tiempo de convergencia de topología 1.

Fuente: Elaboración propia.

Analizando el coeficiente de correlación o del valor de R elevado al cuadrado se muestra que el valor de R es muy pequeño, por lo tanto, se concluye que no existe una relación directa entre estos parámetros.

Topología 2:

De los datos obtenidos del tiempo de convergencia para 20 nodos y el número de iteraciones 11, 21, 31, 41 y 51, se obtiene la gráfica de la Figura 39.

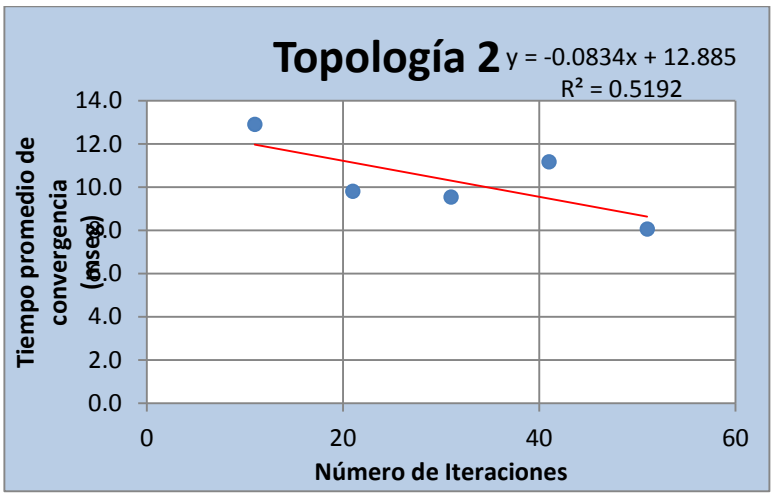


Figura 39. Tendencia lineal de tiempo de convergencia de topología 2.

Fuente: Elaboración propia.

Analizando el coeficiente de correlación o del valor de Re elevado al cuadrado se muestra que el valor de R es el valor promedio y puede indicar una tendencia errónea, por lo tanto, no existe una relación directa entre estos parámetros.

Topología 3:

De los datos obtenidos del tiempo de convergencia para 40 nodos y número de iteraciones 11, 21, 31, 41 y 51. Se obtiene la gráfica de la Figura 40.

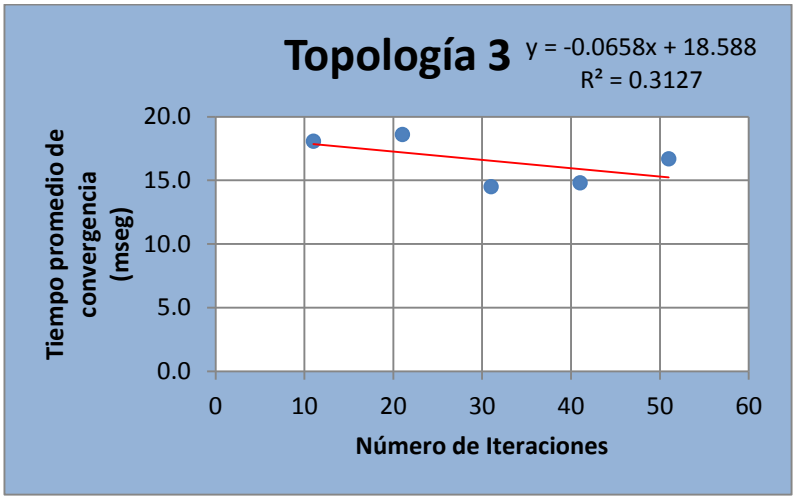


Figura 40. Tendencia lineal de tiempo de convergencia de topología 3.

Fuente: Elaboración propia.

Analizando el coeficiente de correlación o del valor de R elevado al cuadrado se muestra que el valor de R es muy pequeño, por lo tanto, no existe una relación directa entre estos dos parámetros.

Topología 4:

De los datos obtenidos del tiempo de convergencia para 60 nodos y número de iteraciones 11, 21, 31, 41 y 51, se obtiene la gráfica de la Figura 41.

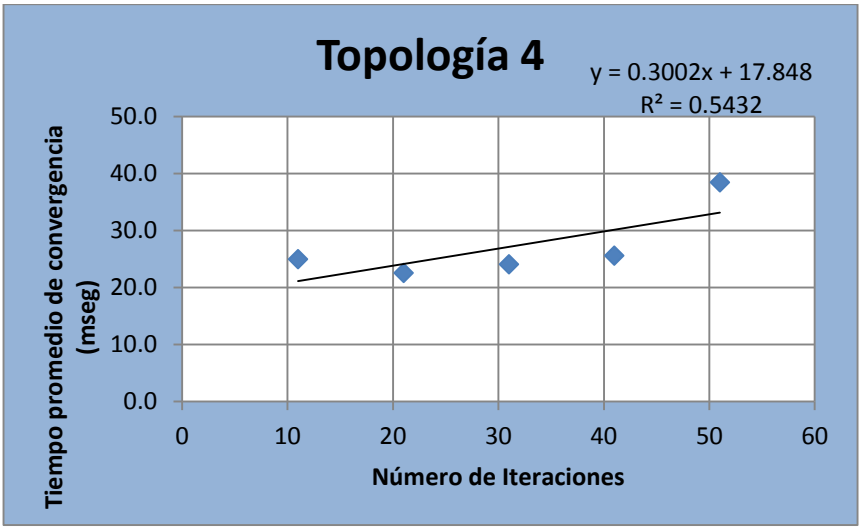


Figura 41: Tendencia lineal de tiempo de convergencia de topología 4.

Fuente: Elaboración propia.

Analizando el coeficiente de correlación o del valor de R elevado al cuadrado se muestra que el valor de R es el valor promedio y puede indicar una tendencia errónea, por lo tanto, no existe una relación directa entre estos parámetros.

Topología 5:

De los datos obtenidos del tiempo de convergencia para 100 nodos y número de iteraciones 11, 21, 31, 41 y 51, se obtiene la gráfica de la Figura 42.

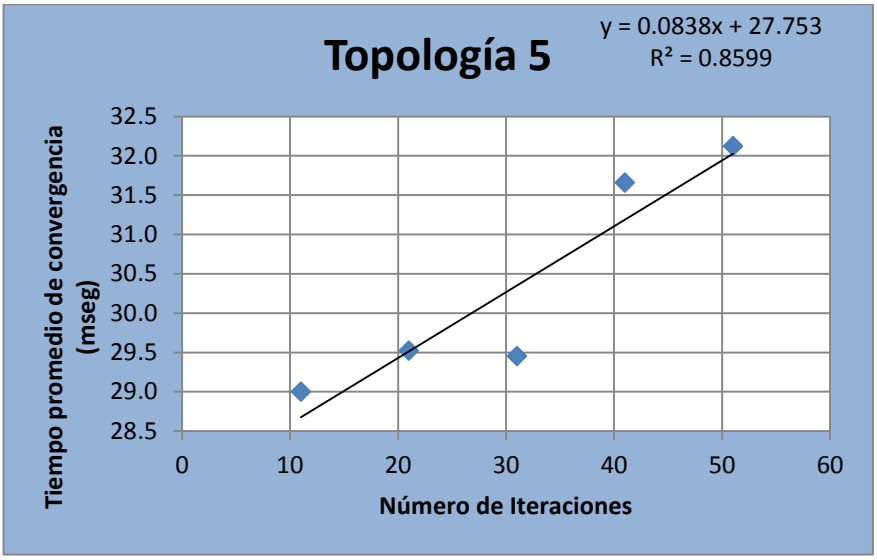


Figura 42. Tendencia lineal de tiempo de convergencia de topología 5.

Fuente: Elaboración propia.

Analizando el coeficiente de correlación o del valor de R elevado al cuadrado se muestra que el valor de R es muy pequeño, por lo tanto, no existe una relación directa entre estos parámetros.

4.6. Análisis del tiempo de convergencia vs el número de nodos

Utilizando la herramienta de Microsoft Excel para realizar estimaciones estadísticas, se ha realizado un análisis de los datos del tiempo de convergencia promedio y el número de nodos (tabla 12), se obtiene el valor del coeficiente de correlación igual a 0.9055, el cual indica que existe una relación directa entre el número de nodo, y el tiempo de convergencia promedio (Ver Tablas 21 y 22, y Figuras 43 y 44).

Tabla 21. Resultados del Excel que muestra la regresión de los datos.
Fuente: Elaboración propia.

Estadísticas de la regresión	
Parámetro	Valor
Coeficiente de correlación múltiple	0,95160452
Coeficiente de determinación R^2	0,90555117
R^2 ajustado	0,87406823
Error típico	128,773,816
Observaciones	5

Tabla 22. Detalle de los cálculos de regresión de los datos.

Fuente: Elaboración propia.

ANÁLISIS DE VARIANZA

	Grados de libertad	*Suma de cuadrados*	*Promedio de los cuadrados*	*F*	*Valor crítico de F*
Regresión	1	4769,71913	4769,71913	28,7632315	0,01268712
Residuos	3	497,480869	165,826956		
Total	4	5267,2			

	Coeficientes	*Error típico*	*Estadístico t*	*Probabilidad*	*Inferior 95%*	*Superior 95%*	*Inferior 95.0%*	*Superior 95.0%*
Intercepción	-22,0077812	13,8591855	-1,58795631	0,21050023	-66,113895	22,0983326	-66,113895	22,0983326
Variable X 1	3,63349909	0,67749519	5,36313635	0,01268712	1,47740702	5,78959117	1,47740702	5,78959117

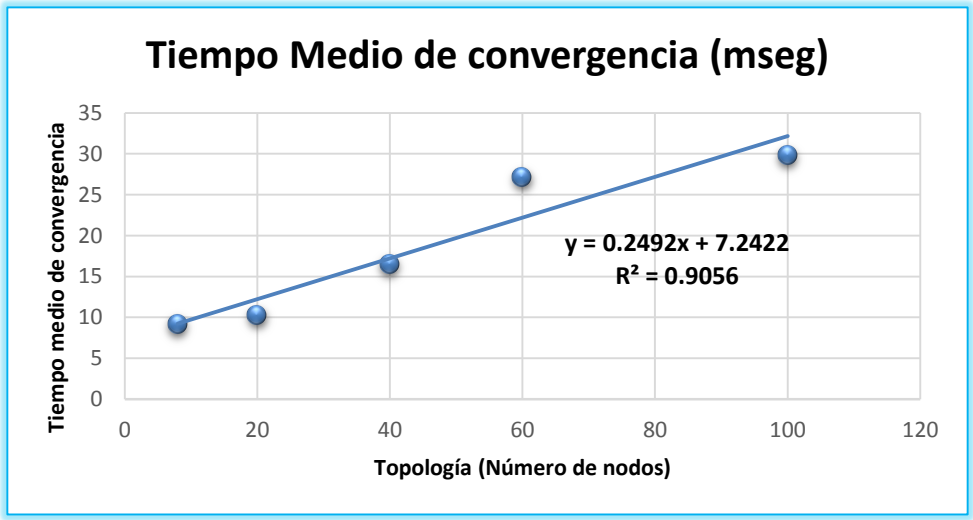


Figura 43. Número de Nodos y tiempo medio de convergencia.

Fuente: Elaboración propia.

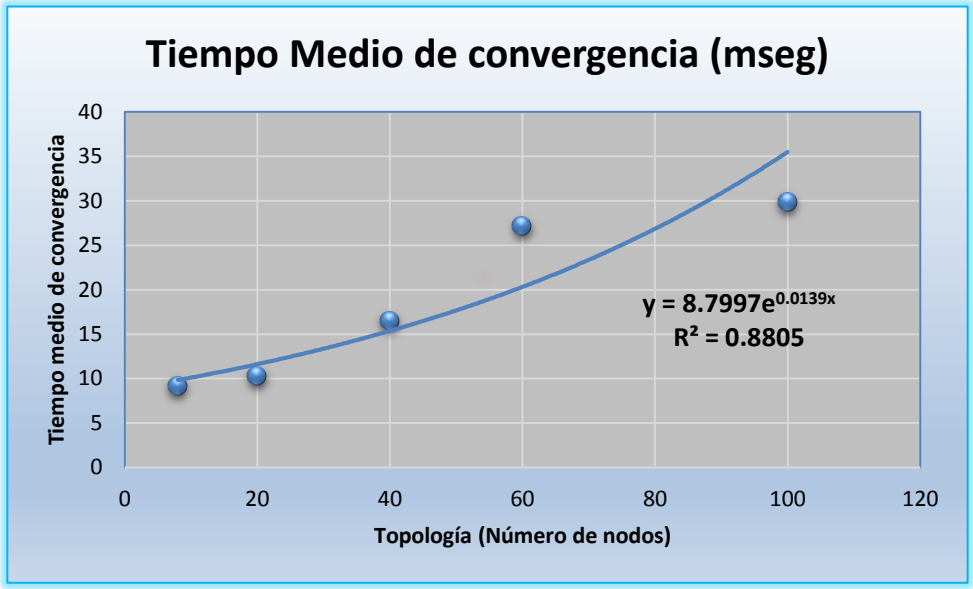


Figura 44. Número de Nodos y tiempo de convergencia.

Fuente: Elaboración propia.

4.7. Análisis de la elección de rutas en las diferentes topologías.

Topología 1.

- ***Conexión entre el nodo N1 y el nodo N5.***

En esta topología se analizan las diferentes rutas existentes entre el nodo N1 y el nodo N5, se observa los valores de sus respectivos pesos y la cantidad de saltos que utiliza de extremo a extremo. La Tabla 23, muestra el número de saltos y los pesos de cada una de las cinco rutas y la Figura 45 muestra el resultado de las diferentes rutas que se obtienen entre el nodo N1 y el nodo N5, como se aprecia, la ruta obtenida por el algoritmo es la ruta de menor peso. Se demuestra que mediante el algoritmo se selecciona la ruta óptima (ruta 5), que es la que tiene menor peso (19).

Tabla 23. Número de rutas entre el nodo N1 y el nodo N5.
Fuente: Elaboración propia.

Rutas del nodo N1 al nodo N5 y sus pesos					
Saltos	**Ruta 1**	**Ruta 2**	**Ruta 3**	**Ruta 4**	**Ruta 5**
0	4	6	5	6	5
1	8	2	8	2	14
2	14	8	8	12	
3		14	14	20	
Peso Total	26	30	35	29	19

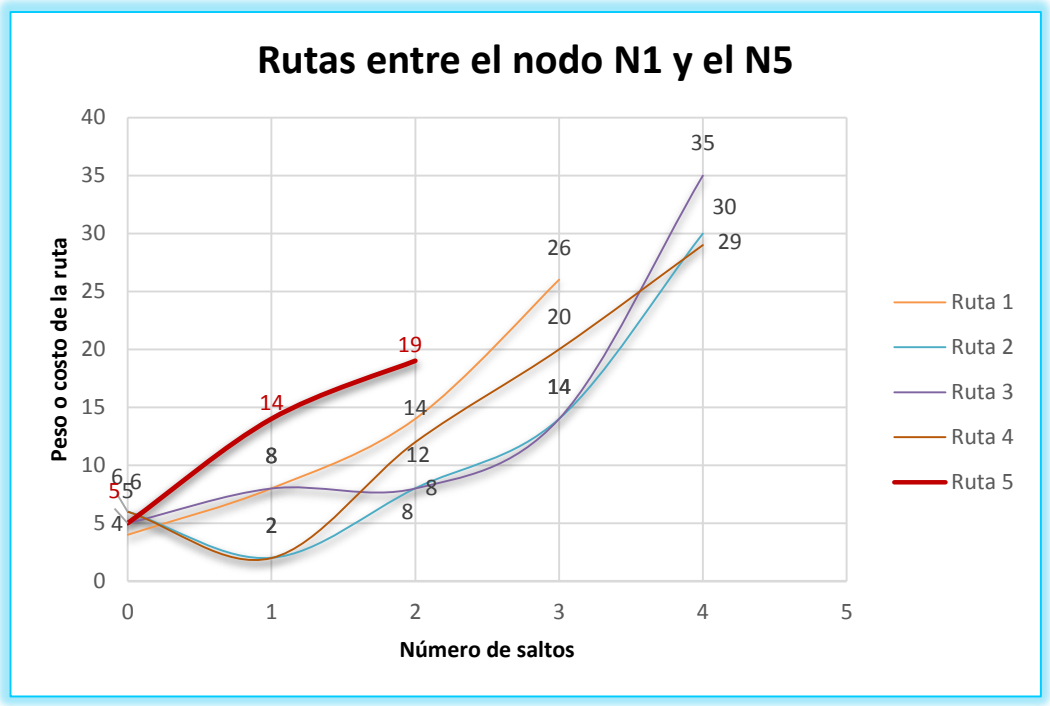


Figura 45. Ruta óptima entre los nodos N1 y N5.
Fuente: Elaboración propia.

- ***Conexión entre el nodo N2 y el nodo N4.***

En esta topología se analizan las diferentes rutas existentes entre el nodo N2 y el nodo N4, se observa los valores de sus respectivos pesos y la cantidad de saltos que utiliza de extremo a extremo. La Tabla 24, muestra el número de saltos y los pesos de cada una de las cinco rutas, la Figura 46 muestra las cinco rutas establecidas entre el nodo N2 y el nodo N4, como se aprecian, la ruta obtenida por el algoritmo es la

ruta de menor peso. Se demuestra que mediante el algoritmo se selecciona la ruta óptima que es la que tiene peso de 10

Tabla 24. Número de rutas entre el nodo N2 y el nodo N4.

Fuente: Elaboración propia.

Rutas del nodo N2 al nodo N4 y sus pesos					
Saltos	**Ruta 1**	**Ruta 2**	**Ruta 3**	**Ruta 4**	**Ruta 5**
0	2	6	6	2	6
1	12	5	4	8	5
2	14	8	8		
3		8			
Peso total	28	27	18	10	11

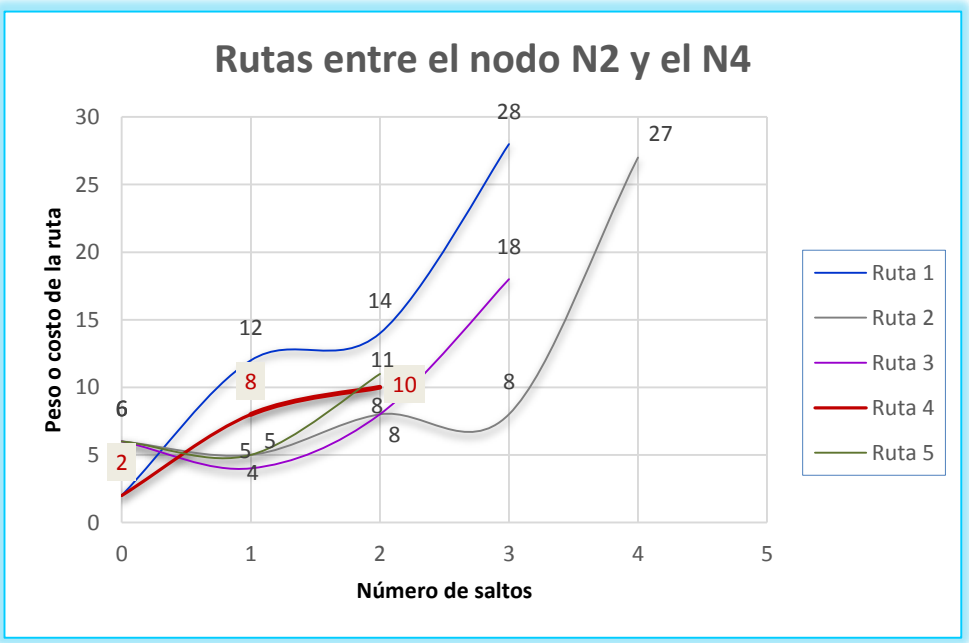


Figura 46. Ruta óptima entre los nodos N2 y N4.

Fuente: Elaboración propia.

- ***Conexión entre el nodo N3 y el nodo N6.***

En esta topología se analizan las diferentes rutas existentes entre el nodo N3 y el nodo N6, se observa los valores de sus respectivos pesos y la cantidad de saltos que utiliza de extremo a extremo. La Tabla 25, muestra el número de saltos y los pesos de cada una de las seis rutas y la Figura 47 muestra las seis posibles rutas que se

obtienen entre el nodo N3 y el nodo N6, como se puede apreciar la ruta obtenida por el algoritmo es la ruta de menor peso. Lo que demuestra que mediante el algoritmo se selecciona la ruta óptima con peso de 4.

Tabla 25. Número de rutas entre el nodo N3 y el nodo N6.

Fuente: Elaboración propia.

Rutas del nodo N3 al nodo N6 y sus pesos						
Saltos	**Ruta 1**	**Ruta 2**	**Ruta 3**	**Ruta 4**	**Ruta 5**	**Ruta 6**
1	8	4	2	4	8	8
2	14	6	2	5	5	5
3	12	2		14	5	6
4				12	14	2
5					12	
Peso total	34	12	4	35	44	21

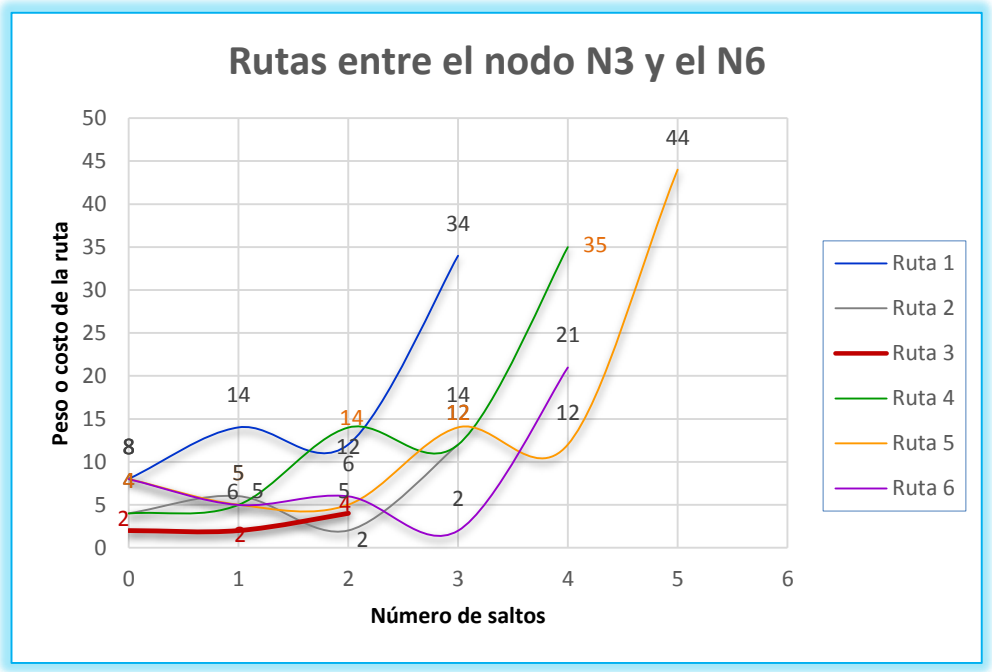


Figura 47. Ruta óptima entre los nodos N3 y N6.

Fuente: Elaboración propia.

- ***Conexión entre el nodo 1 y el nodo 6***

En esta topología se analizan las diferentes rutas existentes entre el nodo N1 y el nodo N6, se observa los valores de sus respectivos pesos y la cantidad de saltos que utiliza de extremo a extremo. La Tabla 26, muestra el número de saltos y los pesos de cada una de las cinco rutas y la Figura 48 muestra las cinco posibles rutas que se obtienen entre el nodo N1 y el nodo N6, como se aprecia la ruta obtenida por el algoritmo es la ruta de menor peso. Lo que demuestra que mediante el algoritmo se selecciona la ruta óptima que tiene el mínimo peso de 8.

Tabla 26. Número de rutas entre el nodo N1 y el nodo N6.

Fuente: Elaboración propia.

Rutas del nodo N1 al nodo N6 y sus pesos					
Saltos	**Ruta 1**	**Ruta 2**	**Ruta 3**	**Ruta 4**	**Ruta 5**
1	6	5	4	5	6
2	2	14	8	8	2
3		12	14	8	8
4			12	14	14
5				12	12
Peso total	8	31	38	47	42

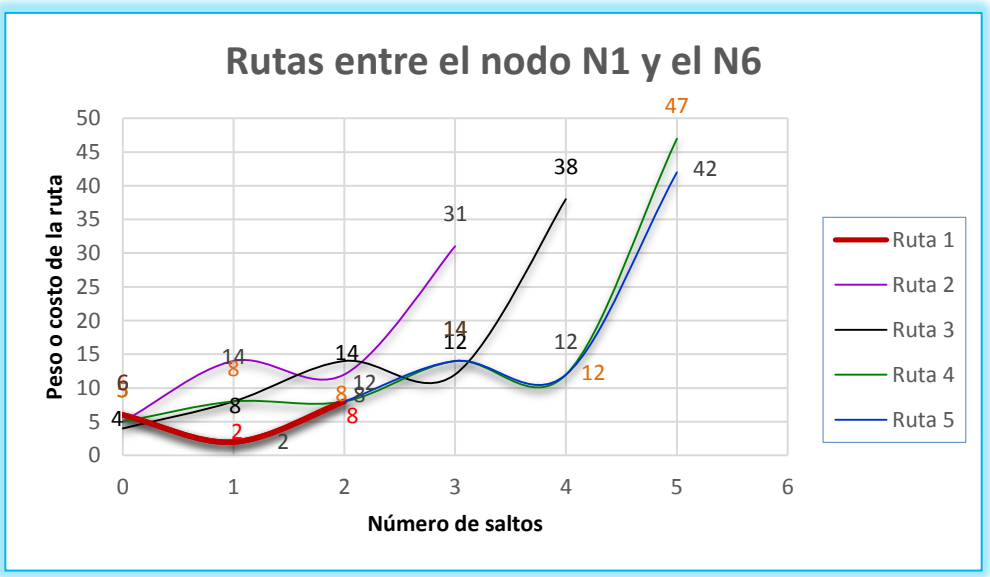


Figura 48. Ruta óptima entre los nodos N1 y N6.

Fuente: Elaboración propia.

Topología 2:

- ***Conexión entre el nodo N1 y el nodo N5***

En esta topología se analizan las diferentes rutas existentes entre el nodo N1 y el nodo N5, se observa los valores de sus respectivos pesos y la cantidad de saltos que utiliza de extremo a extremo. La Tabla 27, muestra el número de saltos y los pesos de cada una de las cinco rutas y la Figura 49 muestra las cinco rutas que se obtienen entre el nodo N1 y el nodo N5, como se aprecia la ruta seleccionada por el algoritmo es la ruta de menor peso. Lo que demuestra que mediante el algoritmo se selecciona la ruta óptima que tiene peso de 8.

Tabla 27. Número de rutas entre los nodos N1 y N5

Fuente: Elaboración propia.

Rutas del nodo N1 al nodo N5 y sus pesos					
Saltos	**Ruta 1**	**Ruta 2**	**Ruta 3**	**Ruta 4**	**Ruta 5**
1	5	5	4	6	5
2	3	14	8	2	14
3			14	12	
Peso total	8	19	26	20	19

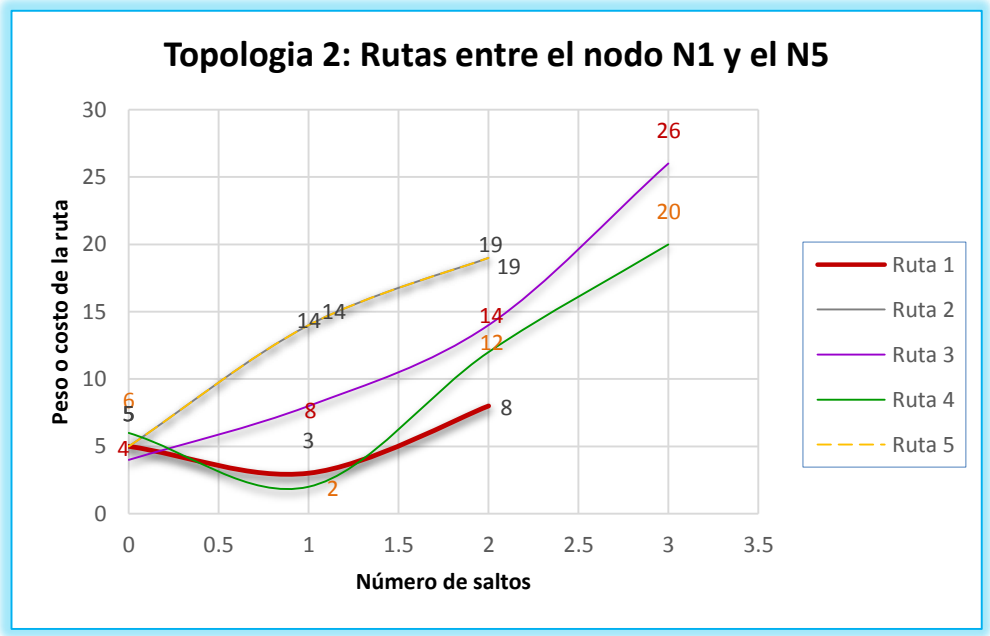


Figura 49. Ruta óptima entre los nodos N1 y N5.

Fuente: Elaboración propia.

- ***Conexión entre el nodo N2 y el nodo N7***

En esta topología se analizan las diferentes rutas existentes entre el nodo N2 y el nodo N7, se observa los valores de sus respectivos pesos y la cantidad de saltos que utiliza de extremo a extremo. La Tabla 28, muestra el número de saltos y los pesos de cada una de las cinco rutas y la Figura 50 muestra las siete rutas que se tiene entre el nodo N2 y el nodo N7, como se aprecia la ruta obtenida por el algoritmo es la ruta de menor peso. Lo que demuestra que mediante el algoritmo se selecciona la ruta óptima que es la que tiene peso de 10.

Tabla 28. Número de rutas entre los nodos N2 y N7

Fuente: Elaboración propia.

Rutas del nodo N2 al nodo N7 y sus pesos							
Saltos	**Ruta 1**	**Ruta 2**	**Ruta 3**	**Ruta 4**	**Ruta 5**	**Ruta 6**	**Ruta 7**
1	6	2	2	2	6	6	6
2	5	12	8	4	4	5	5
3		3		5	8	14	8
4						3	8
Peso total	11	17	10	11	18	28	27

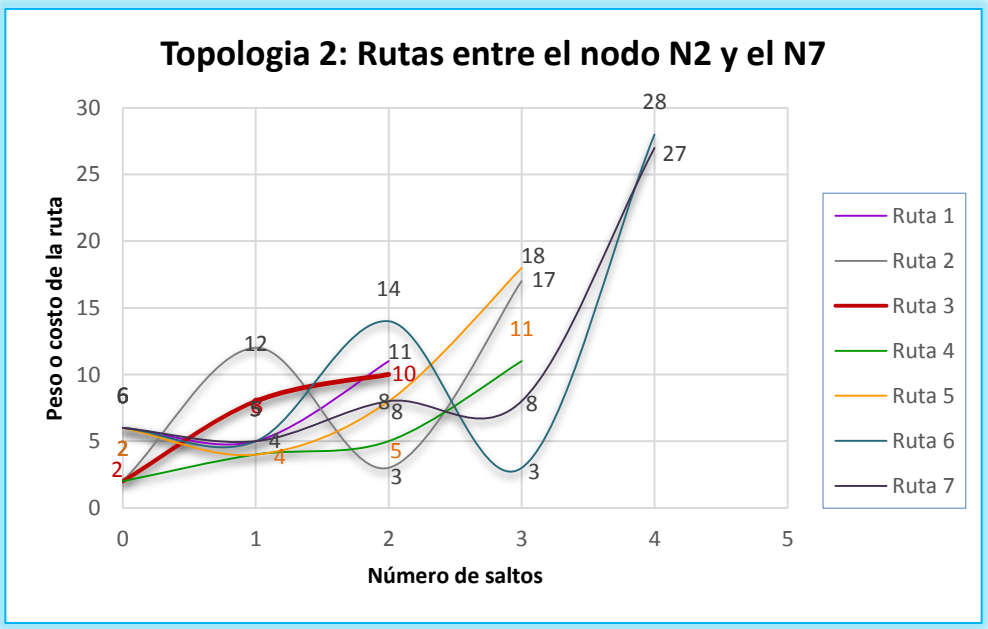


Figura 50: Ruta óptima entre los nodos N2 y N7.

Fuente: Elaboración propia.

- ***Conexión entre el nodo N1 y el nodo N12***

En esta topología se analizan las diferentes rutas existentes entre el nodo N1 y el nodo N12, se observa los valores de sus respectivos pesos y la cantidad de saltos que utiliza de extremo a extremo. La Tabla 29, muestra el número de saltos y los pesos de cada una de las cinco rutas y la Figura 51 muestra las diez rutas que se tiene entre el nodo N1 y el nodo N12, como se aprecia la ruta seleccionada por el algoritmo es la ruta óptima que es la de menor peso de 6.

Tabla 29: Número de rutas entre los nodos N1 y N12.

Fuente: Elaboración propia.

Rutas del nodo N1 al nodo N12 y sus pesos										
Saltos	**Ruta 1**	**Ruta 2**	**Ruta 3**	**Ruta 4**	**Ruta 5**	**Ruta 6**	**Ruta 7**	**Ruta 8**	**Ruta 9**	**Ruta 10**
1	4	4	5	4	6	5	5	5	5	5
2	4	2	8	8	2	3	3	14	14	14
3	6		4	3	12	14	6	14	12	12
4			6	6	14	5		3	2	2
5					3	6		6	2	2
6					6				2	4
7										6
Peso total	14	6	23	21	43	33	14	42	37	45

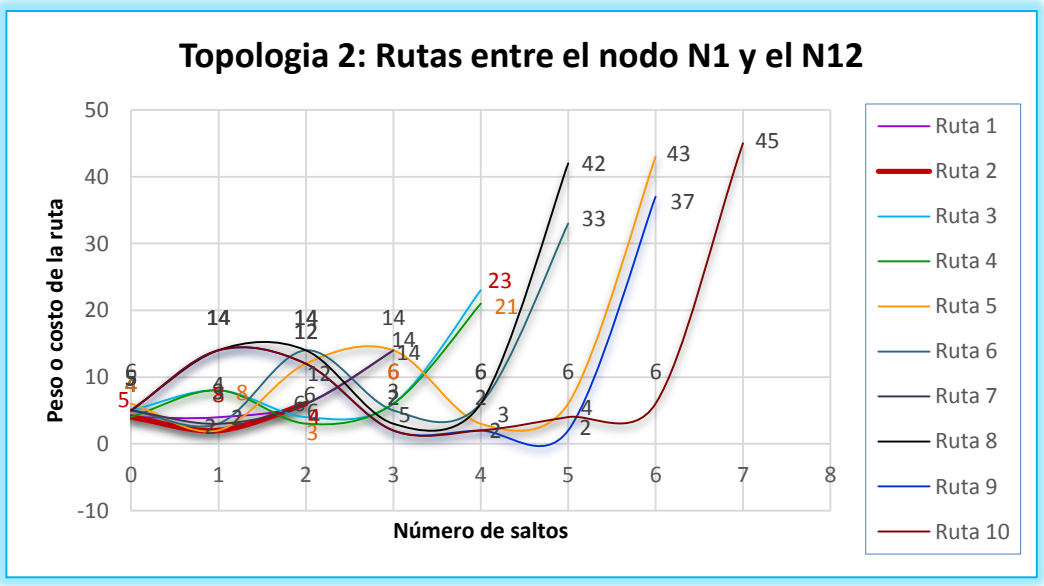


Figura 51. Ruta óptima entre los nodos N1 y N12.

Fuente: Elaboración propia.

- ***Conexión entre el nodo N2 y el nodo N14***

En esta topología se analizan las diferentes rutas existentes entre el nodo N2 y el nodo N14, se observa los valores de sus respectivos pesos y la cantidad de saltos que utiliza de extremo a extremo. La Tabla 30, muestra el número de saltos y los pesos de cada una de las nueve rutas y la Figura 52 muestra las nueve rutas que se obtienen entre el nodo N2 y el nodo N14, como se aprecia la ruta seleccionada obtenida mediante el algoritmo es la ruta óptima con menor peso de 4.

Tabla 30: Número de rutas entre los nodos N2 y N14.

Fuente: Elaboración propia.

Rutas del nodo N2 al nodo N14 y sus pesos									
Saltos	**Ruta 1**	**Ruta 2**	**Ruta 3**	**Ruta 4**	**Ruta 5**	**Ruta 6**	**Ruta 7**	**Ruta 8**	**Ruta 9**
1	2	2	2	2	6	6	6	2	6
2	2	12	4	8	4	5	5	8	5
3		14	3	14	8		3	3	14
4				14			14	14	14
Peso total	4	28	9	38	18	11	28	27	39

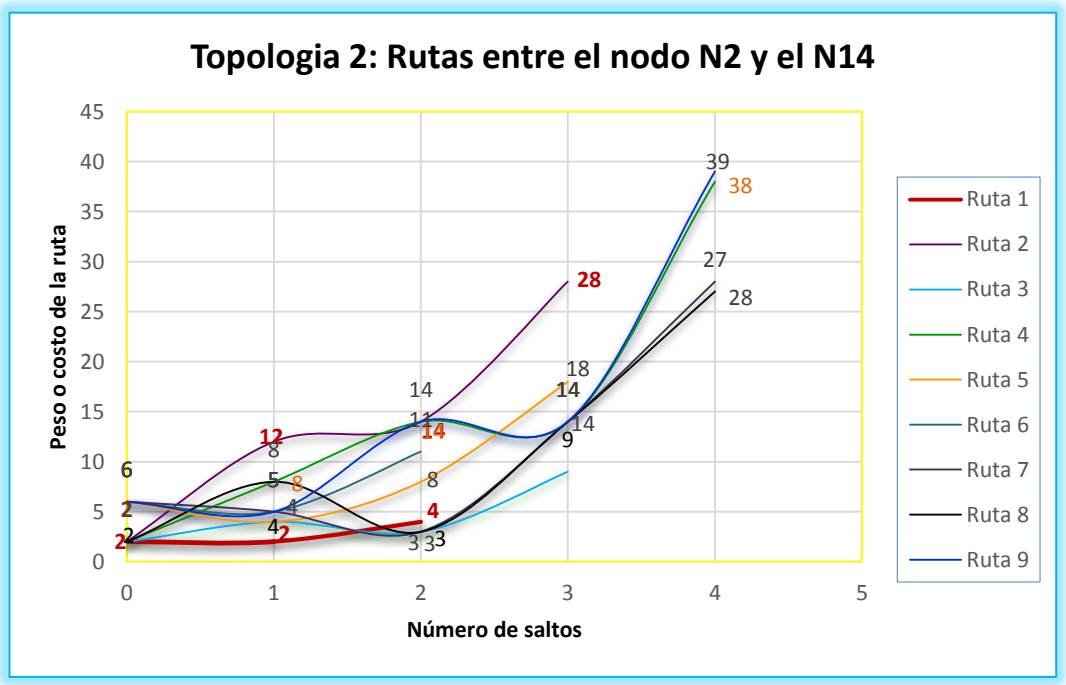


Figura 52: Ruta óptima entre los nodos N2 y N14.

Fuente: Elaboración propia.

CAPÍTULO V

CONCLUSIONES Y RECOMENDACIONES

4.1.Conclusiones.

1. Los cambios en la topología de la red debido a las fallas de los enlaces fueron detectado a tiempo y de manera dinámica mediante el protocolo de comunicaciones, diseñado e implementado para este propósito.
2. La reconstrucción dinámica de la topología de la red es realizada por el protocolo de comunicaciones, diseñado e implementado para este propósito, en base a la información suministrada por los nodos.
3. El tiempo de convergencia del algoritmo de hormigas es directamente proporcional a la complejidad de la red, mientas más nodos y enlaces tiene la red requiere mayor tiempo de convergencia.
4. El costo de las rutas es la sumatoria de todos los costos de los enlaces, a mayor ancho de banda menor costo y a mayor tráfico el costo es mayor.
5. La ruta óptima seleccionada por el algoritmo de hormigas siempre es el de menor costo.

4.2.Recomendaciones

1. El propósito del presente trabajo de investigación consistió en desarrollar un algoritmo basado en la Metaheurística de Colonia de Hormigas para conseguir rutas de bajo costo o bajo peso en redes con diferente número de nodos. Por tanto, la implementación de este algoritmo junto con un protocolo de comunicaciones se puede utilizar como método de manejo de encaminamiento dinámico, en los cuales se tomen en cuenta nuevas variables, tales como el tiempo de vida útil de la red (nodo, Enlace), tráfico cursado por los enlaces, ancho de banda del enlace, latencia, etc.
2. Como trabajo futuro, el algoritmo basado en colonia de hormigas se podría ampliar y combinar con el servicio de telefonía en *Asterisk* de tal manera que

el encaminamiento o Trunking sea dinámico y evitar la programación de las rutas por comando, como se hace con el protocolo IAX (Inter-Asterisk eXchange protocol).

3. Se recomienda realizar nuevas investigaciones y plantear proyectos de investigación para aplicar este algoritmo en otras redes a nivel de la capa de aplicación como son las redes de video, redes de voz y video, redes de servicios unificados, etc.

REFERENCIAS BIBLIOGRÁFICAS

Aguilar, J., & Labrador, M. (Diciembre de 2007). Un algoritmo de enrutamiento distribuido para redes de comunicación basado en sistemas de hormigas. IEEE LATIN AMERICA TRANSACTIONS, 5(8), 616. Obtenido de http://www.ewh.ieee.org/reg/9/etrans/ieee/issues/vol05/vol5issue8Dec.2007/5TLA8_11Aguilar.pdf

Almeida, R. (2015). Implementar una central telefónica IP basada en tecnología open source en la carrera de ingeniería en pen source en la carrera de ingeniería en. (Tesis de Ingeniero de sistemas computacionales), Universidad de Guayaquil, Guayaquil, Ecuador. Obtenido de http://repositorio.ug.edu.ec/bitstream/redug/9946/1/PTG-191-Almeida%20Arboleda%20Ronald%20Edmundo.pdf

Alonso, J. (2008). Flujo en Redes y Gestión de Proyectos. Teoría y Ejercicios Resueltos. La coruña, España: Gesbiblo, S. L. Obtenido de https://books.google.com.pe/books?id=UQpy6PGbo9MC&printsec=frontcover&dq=Flujo+en+Redes+y+Gesti%C3%B3n+de+Proyectos.+Teor%C3%ADa+y+Ejercicios+Resueltos.+Netbiblo&hl=es&sa=X&ved=0ahUKEwjJorSZ7M_WAhVJFZAKHbYaB18Q6AEIJTAA#v=onepage&q=Flujo%20en%20Redes%20y%2

Barbancho, J., Benjumea, J., Rivera, O., Romero, M., Ropero, J., Sánchez, G., & Sivianes, F. (2010). Redes locales. Madrid, España: Paraninfo. Obtenido de https://books.google.es/books?id=duk7k-YoYwEC&printsec=frontcover&hl=es&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false

Caballero, J. (2007). Implementación de una Red (VoIP) a través de software libre enel desarrollo de una pequeña central telefónica. (Tesis de Licenciado en ciencias de la Computación), Universidad Autónoma de Yucatán, Mérida, Yucatán, México. Obtenido de https://es.scribd.com/document/49427671/Implementacion-V0iP

Corbalán, C. (Noviembre de 2006). Sistemas inteligentes aplicados a redes de datos. La plata, Argentina: Universidad Nacional de La Plata. Obtenido de http://postgrado.info.unlp.edu.ar/Carreras/Especializaciones/Redes_y_Seguridad/Trabajos_Finales/Corbalan.pdf

Dominguez, C. (2011). Algoritmos bioinspirados para el encaminamiento de datos en redes inalámbricas de sensores. (Tesis de Maestro en ciencias de la computación), Instituto Politécnico Nacional-Centro de investigación en computación, Distrito Federal, Mexico. Obtenido de http://www.repositoriodigital.ipn.mx/bitstream/123456789/9230/1/Dominguez_Tesis.pdf

Dorigo, M., & Stützle, T. (2004). Ant Colony Optimization. Cambridge, Massaachusetts, USA: Massachusetts Institute of Technology. Obtenido de https://pdfs.semanticscholar.org/7c72/393febe25ef5ce2f5614a75a69e1ed0d9857.pdf

Gainza, C. M. (Marzo de 2008). Voz sobre protocolos IP. Buenos Aires, Argentina. Obtenido de https://docuri.com/download/n4voip_59a8d495f581719e12acfd90_pdf

Gil, P., Pomares, J., & Candelas, F. (2010). Redes de transmisión de datos. (publicaciones@ua.es, Ed.) Alicante, España: Universidad de Alicante. Obtenido de https://books.google.es/books?id=On6y2SEaWyMC&pg=PA189&dq=Gil,+A;+Pomares+J;+Candelas,+F+(2010).+Redes+y+Transmisi%C3%B3n+de+Datos.+Publicaciones+Universidad+de+Alicante.&hl=es&sa=X&ved=0ahUKEwjNtun36M_WAhWBgZAKHUjGDlcQ6wEIJjAA#v=onepage&q=Gil%2C%20A%3B%2

Hernández, R., Fernández , C., & Baptista, M. (2010). Metodología de la investigación. CDMX, Mexico: McGRAW-HILL / INTERAMERICANA EDITORES, S.A. DE C.V. Obtenido de https://www.esup.edu.pe/descargas/dep_investigacion/Metodologia%20de%20la%20investigaci%C3%B3n%205ta%20Edici%C3%B3n.pdf

Jiménez, A. (Febrero de 2012). Determinación de grupos de datos binarios mediante el comportamiento de hormigas. Revista digital Matemática, Educación e Internet, 12(1). Obtenido de http://tecdigital.tec.ac.cr/revistamatematica/ARTICULOS_V12_N1_2011/A_Jimenez_V12N1_2011/index.html

Kennedy, J., & Eberhart, R. (1995). Particle swarm optimization. IEEE Int. Conf on Neuronal Network, 1. Obtenido de http://www.cs.us.es/~fsancho/?e=70

López, D. (2011). Implementación de protocolos de señalización VoIP sobre un entorno de red IPv6. (Tesis de Licenciado en Informática), Universidad de La Sierra Juárez, Ixtlán de Juárez, Oaxaca, Mexico. Obtenido de https://es.scribd.com/document/121937582/Protocol-o

Muñoz, M., López, J., & Caicedo, E. (Agosto de 2008). Inteligencia de enjambres: sociedades para la solución de. Revista Ingeniería E Investigación, 28(2), 119. Obtenido de http://www.scielo.org.co/pdf/iei/v28n2/v28n2a15.pdf

Pinto, D., Estigarribia, H., & Barán , B. (2005). Enrutamiento Multicast Multiobjetivos basado en Colonia de Hormigas. (U. N. Asunción, Ed.) Universidad Nacional de Asunción, 1-10. Obtenido de http://www.cnc.una.py/publicaciones/4_109.pdf

Quintana, D. (2007). Diseño e implementación de una red de telefonía IP con software libre en la RAAP. (Tesis de Ingeniero de las telecomunicaciones), Pontificia Universidad Católica del Perú, Lima, Perú. Obtenido de http://tesis.pucp.edu.pe/repositorio/bitstream/handle/123456789/205/QUINTANA_DIEGO_DISENO_RED_TELEFONIA_IP_RAAP.pdf?sequence=2&isAllowed=y

Rodrigo, H. (2011). Algorítmo de colonia de Hormigas para el enrutamiento en redes de sensores inalámbricos. (Tesis de Magister Scientiarum en ciencias de la computación), Universidad Centroccidental Lisandro Alvarado, Barquisimeto, Lara, Venezuela.

Rupérez, D. (2009). Protocolo de encaminamiento ACO híbrido para redes móviles Ad Hoc. (Trabajo de investigación en maestría), Universidad Complutense de Madrid, Madrid, España. Obtenido de http://docplayer.es/7888675-Proyecto-fin-de-master-en-sistemas-inteligentes-protocolo-de-encaminamiento-aco-hibrido-para-redes-moviles-ad-hoc.html

Santamaría, C., & Tejada, V. (2010). Prueba de Concepto de Un Sistema de Telefonía VoIP Utilizando Redes Inalámbricas de Tipo Mesh (Tesis para Ingeniería Informática). Panamá. (Tesis de Licenciado en Ingeniería Informática), Universidad de Panamá, Transístmica, Panamá. Obtenido de https://wiki.freifunk.net/images/1/1c/Thesis_tejada_santamaria.pdf

ANEXO A. Archivos de las topologías.

En este anexo se tiene todos los archivos de las topologías que se usaron en el desarrollo del trabajo.

A1. Archivo inu.txt, topología 2 original.

20 47 4 0
200.20.20.10 200.20.20.11 200.20.20.12 200.20.20.13 200.20.20.14 200.20.20.15 200.20.20.16 200.20.20.17 200.20.20.18 200.20.20.19 200.20.20.20 200.20.20.21 200.20.20.22 200.20.20.23 200.20.20.24 200.20.20.25 200.20.20.26 200.20.20.27 200.20.20.28 200.20.20.29
0 1 999 2510
0 2 999 2520
0 3 999 2530
0 4 999 2540
0 5 999 2520
0 6 999 2530
0 7 999 2540
0 8 999 2550
0 9 999 2560
0 10 999 2570
0 11 999 2580
0 12 999 2590
0 13 999 2600
0 14 999 2610
0 15 999 2620
0 16 999 2630
0 17 999 2640
0 18 999 2650
0 19 999 2660
1 2 6 2610
1 3 4 2620
4 5 14 2645
3 7 8 2670
1 4 5 2630
5 6 12 2640
2 3 2 2640
3 4 8 2660
1 7 5 2630
2 6 2 2640
5 7 3 2630
10 11 999 2510
10 12 999 2520
10 13 999 2530
10 14 999 2540
11 12 6 2610
11 3 4 2620
14 5 14 2645
13 17 8 2670
1 14 5 2630
15 16 12 2640

12 3 2 2640
3 14 8 2660
11 17 5 2630
12 16 2 2640
15 17 3 2630
11 14 3 2670
0 0 0 0

A2. Archivo inu.txt, topología 3 original.

40 103 4 0
200.20.20.10 200.20.20.11 200.20.20.12 200.20.20.13 200.20.20.14 200.20.20.15 200.20.20.16 200.20.20.17 200.20.20.18 200.20.20.19 200.20.20.20 200.20.20.21 200.20.20.22 200.20.20.23 200.20.20.24 200.20.20.25 200.20.20.26 200.20.20.27 200.20.20.28 200.20.20.29 200.20.20.30 200.20.20.31 200.20.20.32 200.20.20.33 200.20.20.34 200.20.20.35 200.20.20.36 200.20.20.37 200.20.20.38 200.20.20.39 200.20.20.40 200.20.20.41 200.20.20.42 200.20.20.43 200.20.20.44 200.20.20.45 200.20.20.46 200.20.20.47 200.20.20.48 200.20.20.49
0 1 999 2510
0 2 999 2520
0 3 999 2530
0 4 999 2540
0 5 999 2520
0 6 999 2530
0 7 999 2540
0 8 999 2550
0 9 999 2560
0 10 999 2570
0 11 999 2580
0 12 999 2590
0 13 999 2600
0 14 999 2610
0 15 999 2620
0 16 999 2630
0 17 999 2640
0 18 999 2650
0 19 999 2660
0 20 999 2670
0 21 999 2680
0 22 999 2690
0 23 999 2700
0 24 999 2710
0 25 999 2720
0 26 999 2730
0 27 999 2740
0 28 999 2750
0 29 999 2760
0 30 999 2770
0 31 999 2780
0 32 999 2790
0 33 999 2800
0 34 999 2810

0 35 999 2820
0 36 999 2830
0 37 999 2840
0 38 999 2850
0 39 999 2860
1 2 6 2610
1 3 4 2620
4 5 14 2645
3 7 8 2670
1 4 5 2630
5 6 12 2640
2 3 2 2640
3 4 8 2660
1 7 5 2630
2 6 2 2640
5 7 3 2630
1 4 3 2670
10 11 999 2510
10 12 999 2520
10 13 999 2530
10 14 999 2540
11 12 6 2610
11 3 4 2620
14 5 14 2645
13 17 8 2670
1 14 5 2630
15 16 12 2640
12 3 2 2640
3 14 8 2660
11 17 5 2630
12 16 2 2640
15 17 3 2630
11 14 3 2670
20 21 999 2510
20 23 999 2530
20 24 999 2540
30 35 999 2520
30 36 999 2530
21 32 6 2610
13 33 4 2620
24 35 14 2645
23 37 8 2670
31 34 5 2630
25 36 12 2640
32 33 2 2640
33 24 8 2660
21 27 5 2630
22 26 2 2640
25 27 3 2630
31 34 3 2670
10 21 999 2510
10 22 999 2520
10 33 999 2530
10 24 999 2540

20 25 999 2520
20 16 999 2530
20 17 999 2540
21 12 6 2610
31 3 4 2620
14 25 14 2645
13 37 8 2670
12 14 5 2630
15 26 12 2640
12 32 2 2640
33 24 8 2660
11 17 5 2630
13 26 2 2640
12 37 3 2630
0 0 0 0

A3. Archivo inu.txt, topología 4 original.

60 81 4 0
200.20.20.10 200.20.20.11 200.20.20.12 200.20.20.13 200.20.20.14 200.20.20.15 200.20.20.16 200.20.20.17 200.20.20.18 200.20.20.19 200.20.20.20 200.20.20.21 200.20.20.22 200.20.20.23 200.20.20.24 200.20.20.25 200.20.20.26 200.20.20.27 200.20.20.28 200.20.20.29 200.20.20.30 200.20.20.31 200.20.20.32 200.20.20.33 200.20.20.34 200.20.20.35 200.20.20.36 200.20.20.37 200.20.20.38 200.20.20.39 200.20.20.40 200.20.20.41 200.20.20.42 200.20.20.43 200.20.20.44 200.20.20.45 200.20.20.46 200.20.20.47 200.20.20.48 200.20.20.49 200.20.20.50 200.20.20.51 200.20.20.52 200.20.20.53 200.20.20.54 200.20.20.55 200.20.20.56 200.20.20.57 200.20.20.58 200.20.20.59 200.20.20.60 200.20.20.61 200.20.20.62 200.20.20.63 200.20.20.64 200.20.20.65 200.20.20.66 200.20.20.67 200.20.20.68 200.20.20.69
30 1 999 2510
30 32 999 2520
30 33 999 2530
30 34 999 2540
30 21 999 2510
30 22 999 2520
20 33 999 2530
30 34 999 2540
20 5 999 2520
0 26 999 2530
0 37 999 2540
1 2 6 2610
1 3 4 2620
4 5 14 2645
3 7 8 2670
1 4 5 2630
5 6 12 2640
2 3 2 2640
3 4 8 2660
1 7 5 2630
2 6 2 2640
5 7 3 2630

1 4 3 2670
10 11 999 2510
10 12 999 2520
10 13 999 2530
10 14 999 2540
0 15 999 2520
0 16 999 2530
0 17 999 2540
11 12 6 2610
11 3 4 2620
14 5 14 2645
13 17 8 2670
1 14 5 2630
15 16 12 2640
12 3 2 2640
3 14 8 2660
11 17 5 2630
12 16 2 2640
15 17 3 2630
11 14 3 2670
20 21 999 2510
0 22 999 2520
20 23 999 2530
20 24 999 2540
30 35 999 2520
30 36 999 2530
0 37 999 2540
21 32 6 2610
13 33 4 2620
24 35 14 2645
23 37 8 2670
31 34 5 2630
25 36 12 2640
32 33 2 2640
33 24 8 2660
21 27 5 2630
22 26 2 2640
25 27 3 2630
31 34 3 2670
10 21 999 2510
10 22 999 2520
10 33 999 2530
10 24 999 2540
20 25 999 2520
20 16 999 2530
20 17 999 2540
21 12 6 2610
31 3 4 2620
14 25 14 2645
13 37 8 2670
12 14 5 2630
15 26 12 2640
12 32 2 2640
33 24 8 2660

11 17 5 2630
13 26 2 2640
12 37 3 2630
3 14 3 2670
0 0 0 0

A4. Archivo inu.txt, topología 5 original.

100 91 4 0
200.20.20.10 200.20.20.11 200.20.20.12 200.20.20.13 200.20.20.14 200.20.20.15 200.20.20.16 200.20.20.17 200.20.20.18 200.20.20.19 200.20.20.20 200.20.20.21 200.20.20.22 200.20.20.23 200.20.20.24 200.20.20.25 200.20.20.26 200.20.20.27 200.20.20.28 200.20.20.29 200.20.20.30 200.20.20.31 200.20.20.32 200.20.20.33 200.20.20.34 200.20.20.35 200.20.20.36 200.20.20.37 200.20.20.38 200.20.20.39 200.20.20.40 200.20.20.41 200.20.20.42 200.20.20.43 200.20.20.44 200.20.20.45 200.20.20.46 200.20.20.47 200.20.20.48 200.20.20.49 200.20.20.50 200.20.20.51 200.20.20.52 200.20.20.53 200.20.20.54 200.20.20.55 200.20.20.56 200.20.20.57 200.20.20.58 200.20.20.59 200.20.20.60 200.20.20.61 200.20.20.62 200.20.20.63 200.20.20.64 200.20.20.65 200.20.20.66 200.20.20.67 200.20.20.68 200.20.20.69 200.20.20.70 200.20.20.71 200.20.20.72 200.20.20.73 200.20.20.74 200.20.20.75 200.20.20.76 200.20.20.77 200.20.20.78 200.20.20.79 200.20.20.80 200.20.20.81 200.20.20.82 200.20.20.83 200.20.20.84 200.20.20.85 200.20.20.86 200.20.20.87 200.20.20.88 200.20.20.89 200.20.20.90 200.20.20.91 200.20.20.92 200.20.20.93 200.20.20.94 200.20.20.95 200.20.20.96 200.20.20.97 200.20.20.98 200.20.20.99 200.20.20.100 200.20.20.101 200.20.20.102 200.20.20.103 200.20.20.104 200.20.20.105 200.20.20.106 200.20.20.107 200.20.20.108 200.20.20.109
30 51 999 2510
36 22 999 2520
38 93 999 2530
80 83 999 2540
80 82 999 2510
90 82 999 2520
67 33 999 2530
87 34 999 2540
68 56 999 2520
76 86 999 2530
30 1 999 2510
30 32 999 2520
30 33 999 2530
30 34 999 2540
30 21 999 2510
30 22 999 2520
20 33 999 2530
30 34 999 2540
20 5 999 2520
0 26 999 2530
0 37 999 2540
1 2 6 2610
1 3 4 2620
4 5 14 2645
3 7 8 2670
1 4 5 2630

5 6 12 2640
2 3 2 2640
3 4 8 2660
1 7 5 2630
2 6 2 2640
5 7 3 2630
1 4 3 2670
10 11 999 2510
10 12 999 2520
10 13 999 2530
10 14 999 2540
0 15 999 2520
0 16 999 2530
0 17 999 2540
11 12 6 2610
11 3 4 2620
14 5 14 2645
13 17 8 2670
1 14 5 2630
15 16 12 2640
12 3 2 2640
3 14 8 2660
11 17 5 2630
12 16 2 2640
15 17 3 2630
11 14 3 2670
20 21 999 2510
0 22 999 2520
20 23 999 2530
20 24 999 2540
30 35 999 2520
30 36 999 2530
0 37 999 2540
21 32 6 2610
13 33 4 2620
24 35 14 2645
23 37 8 2670
31 34 5 2630
25 36 12 2640
32 33 2 2640
33 24 8 2660
21 27 5 2630
22 26 2 2640
25 27 3 2630
31 34 3 2670
10 21 999 2510
10 22 999 2520
10 33 999 2530
10 24 999 2540
20 25 999 2520
20 16 999 2530
20 17 999 2540
21 12 6 2610
31 3 4 2620

14 25 14 2645
13 37 8 2670
12 14 5 2630
15 26 12 2640
12 32 2 2640
33 24 8 2660
11 17 5 2630
13 26 2 2640
12 37 3 2630
3 14 3 2670
0 0 0 0

ANEXO B. Archivos de las topologías con falla de enlaces.

B1. Topología 2, sin el enlace del nodo N3 al N4.

20 46 4 0
200.20.20.10 200.20.20.11 200.20.20.12 200.20.20.13 200.20.20.14 200.20.20.15 200.20.20.16 200.20.20.17 200.20.20.18 200.20.20.19 200.20.20.20 200.20.20.21 200.20.20.22 200.20.20.23 200.20.20.24 200.20.20.25 200.20.20.26 200.20.20.27 200.20.20.28 200.20.20.29
0 1 999 2510
0 2 999 2520
0 3 999 2530
0 4 999 2540
0 5 999 2520
0 6 999 2530
0 7 999 2540
0 8 999 2550
0 9 999 2560
0 10 999 2570
0 11 999 2580
0 12 999 2590
0 13 999 2600
0 14 999 2610
0 15 999 2620
0 16 999 2630
0 17 999 2640
0 18 999 2650
0 19 999 2660
1 2 6 2610
1 3 4 2620
4 5 14 2645
3 7 8 2670
1 4 5 2630
5 6 12 2640
2 3 2 2640
1 7 5 2630
2 6 2 2640
5 7 3 2630
10 11 999 2510
10 12 999 2520
10 13 999 2530
10 14 999 2540
11 12 6 2610
11 3 4 2620
14 5 14 2645
13 17 8 2670
1 14 5 2630
15 16 12 2640
12 3 2 2640
3 14 8 2660
11 17 5 2630
12 16 2 2640
15 17 3 2630

11 14 3 2670
0 0 0 0

B2. Topología 2, sin el enlace del nodo N14 al N5.

20 45 4 0
200.20.20.10 200.20.20.11 200.20.20.12 200.20.20.13 200.20.20.14 200.20.20.15 200.20.20.16 200.20.20.17 200.20.20.18 200.20.20.19 200.20.20.20 200.20.20.21 200.20.20.22 200.20.20.23 200.20.20.24 200.20.20.25 200.20.20.26 200.20.20.27 200.20.20.28 200.20.20.29
0 1 999 2510
0 2 999 2520
0 3 999 2530
0 4 999 2540
0 5 999 2520
0 6 999 2530
0 7 999 2540
0 8 999 2550
0 9 999 2560
0 10 999 2570
0 11 999 2580
0 12 999 2590
0 13 999 2600
0 14 999 2610
0 15 999 2620
0 16 999 2630
0 17 999 2640
0 18 999 2650
0 19 999 2660
1 2 6 2610
1 3 4 2620
4 5 14 2645
3 7 8 2670
1 4 5 2630
5 6 12 2640
2 3 2 2640
1 7 5 2630
2 6 2 2640
5 7 3 2630
10 11 999 2510
10 12 999 2520
10 13 999 2530
10 14 999 2540
11 12 6 2610
11 3 4 2620
13 17 8 2670
1 14 5 2630
15 16 12 2640
12 3 2 2640
3 14 8 2660
11 17 5 2630

12 16 2 2640
15 17 3 2630
11 14 3 2670
0 0 0 0

B3. Topología 2, sin el enlace entre del nodo N1 al N3.

20 44 4 0
200.20.20.10 200.20.20.11 200.20.20.12 200.20.20.13 200.20.20.14 200.20.20.15 200.20.20.16 200.20.20.17 200.20.20.18 200.20.20.19 200.20.20.20 200.20.20.21 200.20.20.22 200.20.20.23 200.20.20.24 200.20.20.25 200.20.20.26 200.20.20.27 200.20.20.28 200.20.20.29
0 1 999 2510
0 2 999 2520
0 3 999 2530
0 4 999 2540
0 5 999 2520
0 6 999 2530
0 7 999 2540
0 8 999 2550
0 9 999 2560
0 10 999 2570
0 11 999 2580
0 12 999 2590
0 13 999 2600
0 14 999 2610
0 15 999 2620
0 16 999 2630
0 17 999 2640
0 18 999 2650
0 19 999 2660
1 2 6 2610
4 5 14 2645
3 7 8 2670
1 4 5 2630
5 6 12 2640
2 3 2 2640
1 7 5 2630
2 6 2 2640
5 7 3 2630
10 11 999 2510
10 12 999 2520
10 13 999 2530
10 14 999 2540
11 12 6 2610
11 3 4 2620
13 17 8 2670
1 14 5 2630
15 16 12 2640
12 3 2 2640
3 14 8 2660
11 17 5 2630
12 16 2 2640

15 17 3 2630
11 14 3 2670
0 0 0 0

B4. Topología 3, sin el enlace del nodo N3 al N14 y N12 al N16.

40 101 4 0
200.20.20.10 200.20.20.11 200.20.20.12 200.20.20.13 200.20.20.14 200.20.20.15 200.20.20.16 200.20.20.17 200.20.20.18 200.20.20.19 200.20.20.20 200.20.20.21 200.20.20.22 200.20.20.23 200.20.20.24 200.20.20.25 200.20.20.26 200.20.20.27 200.20.20.28 200.20.20.29 200.20.20.30 200.20.20.31 200.20.20.32 200.20.20.33 200.20.20.34 200.20.20.35 200.20.20.36 200.20.20.37 200.20.20.38 200.20.20.39 200.20.20.40 200.20.20.41 200.20.20.42 200.20.20.43 200.20.20.44 200.20.20.45 200.20.20.46 200.20.20.47 200.20.20.48 200.20.20.49
0 1 999 2510
0 2 999 2520
0 3 999 2530
0 4 999 2540
0 5 999 2520
0 6 999 2530
0 7 999 2540
0 8 999 2550
0 9 999 2560
0 10 999 2570
0 11 999 2580
0 12 999 2590
0 13 999 2600
0 14 999 2610
0 15 999 2620
0 16 999 2630
0 17 999 2640
0 18 999 2650
0 19 999 2660
0 20 999 2670
0 21 999 2680
0 22 999 2690
0 23 999 2700
0 24 999 2710
0 25 999 2720
0 26 999 2730
0 27 999 2740
0 28 999 2750
0 29 999 2760
0 30 999 2770
0 31 999 2780
0 32 999 2790
0 33 999 2800
0 34 999 2810
0 35 999 2820
0 36 999 2830

0 37 999 2840
0 38 999 2850
0 39 999 2860
1 2 6 2610
1 3 4 2620
4 5 14 2645
3 7 8 2670
1 4 5 2630
5 6 12 2640
2 3 2 2640
3 4 8 2660
1 7 5 2630
2 6 2 2640
5 7 3 2630
1 4 3 2670
10 11 999 2510
10 12 999 2520
10 13 999 2530
10 14 999 2540
11 12 6 2610
11 3 4 2620
14 5 14 2645
13 17 8 2670
1 14 5 2630
15 16 12 2640
12 3 2 2640
11 17 5 2630
15 17 3 2630
11 14 3 2670
20 21 999 2510
20 23 999 2530
20 24 999 2540
30 35 999 2520
30 36 999 2530
21 32 6 2610
13 33 4 2620
24 35 14 2645
23 37 8 2670
31 34 5 2630
25 36 12 2640
32 33 2 2640
33 24 8 2660
21 27 5 2630
22 26 2 2640
25 27 3 2630
31 34 3 2670
10 21 999 2510
10 22 999 2520
10 33 999 2530
10 24 999 2540
20 25 999 2520
20 16 999 2530
20 17 999 2540
21 12 6 2610

31 3 4 2620
14 25 14 2645
13 37 8 2670
12 14 5 2630
15 26 12 2640
12 32 2 2640
33 24 8 2660
11 17 5 2630
13 26 2 2640
12 37 3 2630
0 0 0 0

B5. Topología 4, sin el enlace del nodo N5 al N6 y de N11 al N12.

60 79 4 0
200.20.20.10 200.20.20.11 200.20.20.12 200.20.20.13 200.20.20.14 200.20.20.15 200.20.20.16 200.20.20.17 200.20.20.18 200.20.20.19 200.20.20.20 200.20.20.21 200.20.20.22 200.20.20.23 200.20.20.24 200.20.20.25 200.20.20.26 200.20.20.27 200.20.20.28 200.20.20.29 200.20.20.30 200.20.20.31 200.20.20.32 200.20.20.33 200.20.20.34 200.20.20.35 200.20.20.36 200.20.20.37 200.20.20.38 200.20.20.39 200.20.20.40 200.20.20.41 200.20.20.42 200.20.20.43 200.20.20.44 200.20.20.45 200.20.20.46 200.20.20.47 200.20.20.48 200.20.20.49 200.20.20.50 200.20.20.51 200.20.20.52 200.20.20.53 200.20.20.54 200.20.20.55 200.20.20.56 200.20.20.57 200.20.20.58 200.20.20.59 200.20.20.60 200.20.20.61 200.20.20.62 200.20.20.63 200.20.20.64 200.20.20.65 200.20.20.66 200.20.20.67 200.20.20.68 200.20.20.69
30 1 999 2510
30 32 999 2520
30 33 999 2530
30 34 999 2540
30 21 999 2510
30 22 999 2520
20 33 999 2530
30 34 999 2540
20 5 999 2520
0 26 999 2530
0 37 999 2540
1 2 6 2610
1 3 4 2620
4 5 14 2645
3 7 8 2670
1 4 5 2630
2 3 2 2640
3 4 8 2660
1 7 5 2630
2 6 2 2640
5 7 3 2630
1 4 3 2670
10 11 999 2510
10 12 999 2520
10 13 999 2530

10 14 999 2540
0 15 999 2520
0 16 999 2530
0 17 999 2540
11 3 4 2620
14 5 14 2645
13 17 8 2670
1 14 5 2630
15 16 12 2640
12 3 2 2640
3 14 8 2660
11 17 5 2630
12 16 2 2640
15 17 3 2630
11 14 3 2670
20 21 999 2510
0 22 999 2520
20 23 999 2530
20 24 999 2540
30 35 999 2520
30 36 999 2530
0 37 999 2540
21 32 6 2610
13 33 4 2620
24 35 14 2645
23 37 8 2670
31 34 5 2630
25 36 12 2640
32 33 2 2640
33 24 8 2660
21 27 5 2630
22 26 2 2640
25 27 3 2630
31 34 3 2670
10 21 999 2510
10 22 999 2520
10 33 999 2530
10 24 999 2540
20 25 999 2520
20 16 999 2530
20 17 999 2540
21 12 6 2610
31 3 4 2620
14 25 14 2645
13 37 8 2670
12 14 5 2630
15 26 12 2640
12 32 2 2640
33 24 8 2660
11 17 5 2630
13 26 2 2640
12 37 3 2630
3 14 3 2670
0 0 0 0

B6. Topología 5, sin el enlace del nodo N15 al N16 y de N14 al N5.

100 89 4 0
200.20.20.10 200.20.20.11 200.20.20.12 200.20.20.13 200.20.20.14 200.20.20.15 200.20.20.16 200.20.20.17 200.20.20.18 200.20.20.19 200.20.20.20 200.20.20.21 200.20.20.22 200.20.20.23 200.20.20.24 200.20.20.25 200.20.20.26 200.20.20.27 200.20.20.28 200.20.20.29 200.20.20.30 200.20.20.31 200.20.20.32 200.20.20.33 200.20.20.34 200.20.20.35 200.20.20.36 200.20.20.37 200.20.20.38 200.20.20.39 200.20.20.40 200.20.20.41 200.20.20.42 200.20.20.43 200.20.20.44 200.20.20.45 200.20.20.46 200.20.20.47 200.20.20.48 200.20.20.49 200.20.20.50 200.20.20.51 200.20.20.52 200.20.20.53 200.20.20.54 200.20.20.55 200.20.20.56 200.20.20.57 200.20.20.58 200.20.20.59 200.20.20.60 200.20.20.61 200.20.20.62 200.20.20.63 200.20.20.64 200.20.20.65 200.20.20.66 200.20.20.67 200.20.20.68 200.20.20.69 200.20.20.70 200.20.20.71 200.20.20.72 200.20.20.73 200.20.20.74 200.20.20.75 200.20.20.76 200.20.20.77 200.20.20.78 200.20.20.79 200.20.20.80 200.20.20.81 200.20.20.82 200.20.20.83 200.20.20.84 200.20.20.85 200.20.20.86 200.20.20.87 200.20.20.88 200.20.20.89 200.20.20.90 200.20.20.91 200.20.20.92 200.20.20.93 200.20.20.94 200.20.20.95 200.20.20.96 200.20.20.97 200.20.20.98 200.20.20.99 200.20.20.100 200.20.20.101 200.20.20.102 200.20.20.103 200.20.20.104 200.20.20.105 200.20.20.106 200.20.20.107 200.20.20.108 200.20.20.109
30 51 999 2510
36 22 999 2520
38 93 999 2530
80 83 999 2540
80 82 999 2510
90 82 999 2520
67 33 999 2530
87 34 999 2540
68 56 999 2520
76 86 999 2530
30 1 999 2510
30 32 999 2520
30 33 999 2530
30 34 999 2540
30 21 999 2510
30 22 999 2520
20 33 999 2530
30 34 999 2540
20 5 999 2520
0 26 999 2530
0 37 999 2540
1 2 6 2610
1 3 4 2620
4 5 14 2645
3 7 8 2670
1 4 5 2630
5 6 12 2640
2 3 2 2640
3 4 8 2660
1 7 5 2630
2 6 2 2640

5 7 3 2630
1 4 3 2670
10 11 999 2510
10 12 999 2520
10 13 999 2530
10 14 999 2540
0 15 999 2520
0 16 999 2530
0 17 999 2540
11 12 6 2610
11 3 4 2620
13 17 8 2670
1 14 5 2630
12 3 2 2640
3 14 8 2660
11 17 5 2630
12 16 2 2640
15 17 3 2630
11 14 3 2670
20 21 999 2510
0 22 999 2520
20 23 999 2530
20 24 999 2540
30 35 999 2520
30 36 999 2530
0 37 999 2540
21 32 6 2610
13 33 4 2620
24 35 14 2645
23 37 8 2670
31 34 5 2630
25 36 12 2640
32 33 2 2640
33 24 8 2660
21 27 5 2630
22 26 2 2640
25 27 3 2630
31 34 3 2670
10 21 999 2510
10 22 999 2520
10 33 999 2530
10 24 999 2540
20 25 999 2520
20 16 999 2530
20 17 999 2540
21 12 6 2610
31 3 4 2620
14 25 14 2645
13 37 8 2670
12 14 5 2630
15 26 12 2640
12 32 2 2640
33 24 8 2660
11 17 5 2630

13 26 2 2640
12 37 3 2630
3 14 3 2670
0 0 0 0

ANEXO C: Programa del protocolo instalado en el servidor.

```
import java.util.*;
import java.net.*;
import java.io.*;
public class servidorv1 extends Thread
{
public static int nequipos;
public static int tablapdeconexiones [][];
public static int grafo [][];
public static int pesos [][];
public static int puertos [][];
public static int nconex;
public static int f=0;
public static int dat=0;
public static String tabladenodoip [];
int nodo;
int puerto;
String ip;
int port = 2510;
public static Scanner sc = new Scanner( System.in );
public static int peso_inicial=10;
        //función de inicialización
public servidorv1(){}
public servidorv1(int nn,int pp)
{
nodo=nn;
puerto=pp;
}
public void run()
{
long tiempoInicio2 = System.currentTimeMillis();
try
{
//sacar la ip del nodo
ip=tabladenodoip[nodo];
            DatagramSocket socketUDP = new DatagramSocket();
            byte[] mensaje;
```

```
            byte[] men;
            byte[] men2;
           byte[] puerto_vecindad;
            InetAddress hostServidor = InetAddress.getByName(ip);
            int puertoServidor = puerto;
            int nco=0;
//numero de conexiones que esta el primer nodo;
          for(int i=0;i<nequipos;i++)
{
if(tablapdeconexiones[nodo][i]==1)
nco++;
}
            String union=Integer.toString(nco);
            men=union.getBytes();
            DatagramPacket numeros = new DatagramPacket(men,men.length, hostServidor,
puertoServidor);
          //se manda numeros
            //*******************************manda numero de conenxones
          socketUDP.send(numeros);
              String suip=tabladenodoip[nodo];
            men2=suip.getBytes();
            DatagramPacket suips = new DatagramPacket(men2,men2.length, hostServidor,
puertoServidor);
            //manda su ip
            socketUDP.send(suips);
          for(int i=0;i<nequipos;i++)
// Construimos un datagrama para enviar el mensaje al servidor
{
             if(tablapdeconexiones[nodo][i]==1)
          {
            mensaje=tabladenodoip[i].getBytes();
            DatagramPacket peticion = new DatagramPacket(mensaje,mensaje.length, hostServidor,
puertoServidor);
            // Enviamos el datagrama
            socketUDP.send(peticion);
            String unpuerto=Integer.toString(puertos[nodo][i]);
            puerto_vecindad=unpuerto.getBytes();
            DatagramPacket tu_puerto = new
```

```
DatagramPacket(puerto_vecindad,puerto_vecindad.length,hostServidor, puertoServidor);
              socketUDP.send(tu_puerto);
              // Construimos el DatagramPacket que contendrá la respuesta
             byte[] bufer = new byte[1024];
              DatagramPacket respuesta = new DatagramPacket(bufer, bufer.length);
              socketUDP.receive(respuesta);
              String frase = new String(respuesta.getData());
             // Enviamos la respuesta del servidor a la salida estandar
              System.out.println("Respuesta: "+ frase );
             }
}
}
catch (Exception e)
{
System.out.println(e);
}
long totalTiempo2 = System.currentTimeMillis() - tiempoInicio2;
System.out.println("El tiempo en establecer conexión con el nodo "+nodo+": "+ totalTiempo2 + "
miliseg");
}
public static void main(String[] args)
{
long tiempoInicio = System.currentTimeMillis();
Scanner entrada=new Scanner(System.in);
//"inserte en numero de equipos a conectar"
nequipos=entrada.nextInt();
dat=entrada.nextInt();
java.util.Date fecha = new Date();
System.out.println(".....");
System.out.println (""+fecha);
System.out.println(".....");
System.out.println("inicio de sesión el servidor");
System.out.println("numero de nodos a conectar "+nequipos);
tabladenodoip=new String[nequipos];
for(int i=0;i<nequipos;i++)
{
//"inserte la ip del equipo "+i);
tabladenodoip[i]= entrada.next();
```

```
}
//"inserte la topologia - que nodos estan conectados");
//"al terminar ponga 0 - 0"
tablapdeconexiones=new int[nequipos+1][nequipos+1];
pesos=new int[nequipos+1][nequipos+1];
            puertos=new int[nequipos+1][nequipos+1];
            grafo=new int[nequipos+1][nequipos+1];
int a=2,b=2,p=0,pes=0;
while(a+b>0)
{
a=entrada.nextInt();
b=entrada.nextInt();
                pes=entrada.nextInt();
                p=entrada.nextInt();
tablapdeconexiones[a][b]=1;
tablapdeconexiones[b][a]=1;
puertos[a][b]=p;
                puertos[b][a]=p;
                //grafo[f][0]=a;
            //grafo[f][1]=b;
            f=f+1;
              if(a!=b)
              {
                pesos[a][b]=pes;
pesos[b][a]=pes;
              }
}
for(int i=0;i<nequipos;i++)
{
pesos[i][i]=999;
}
tablapdeconexiones[0][0]=0;
long totalTiempo = System.currentTimeMillis() - tiempoInicio;
System.out.println("*tiempor de reconocimiento de topologia: " + totalTiempo + " miliseg");
            for(int i=0;i<nequipos;i++)
             {
              System.out.println("nodo ip "+i+" : "+tabladenodoip[i]);
             }
```

```
System.out.println("");
System.out.println("*************** Matriz de adyacencia ************* ");
        for(int i=0;i<nequipos;i++)
          { for(int j=0;j<nequipos;j++)
            {
                  System.out.print(tablapdeconexiones[i][j]+"");
            }
            System.out.println("");
           }
        System.out.println("");
        System.out.println("**************** Matriz de pesos *****************");
        for(int i=0;i<nequipos;i++)
          { for(int j=0;j<nequipos;j++)
            {
             if(pesos[i][j]==999)
             {
                   System.out.print("∞   ");
                 }
                 else
                 {
                   System.out.print(pesos[i][j]+"");
                 }
            }
            System.out.println("");
           }

        System.out.println("");
        System.out.println("****************** Matriz de puertos **********************");
        for(int i=0;i<nequipos;i++)
          { for(int j=0;j<nequipos;j++)
            {
             if(puertos[i][j]==0)
             {
                   System.out.print("cone   ");
                 }
                 else
                 {
                   System.out.print(puertos[i][j]+"");
```

```
                    }

                }
                System.out.println("");
              }
            System.out.println("");
System.out.println("esperando la respuesta del servidor");
for(int i=1;i<nequipos;i++)
{
servidorv1 c1=new servidorv1(i,2510+10*(i-1));
                    c1.start();
                }
}
}
```

ANEXO D: Programa del protocolo instalado en cada nodo de la red.

```
import java.io.*;
import java.net.*;
public class nodov1
{
public static String tablapropiaconex[];
public static int propios_puertos[];
public static void main(String args[]) throws Exception
{
int port = 2520;
while(true)
{
System.out.println(".....");
System.out.println("esperando conexion .....");
int i=0;
DatagramSocket socketUDP = new DatagramSocket(port);
byte[] bufer = new byte[1024];
byte[] responde = new byte[1024];
byte[] bufer1 = new byte[100];
byte[] bufer2 = new byte[100];
byte[] bu = new byte[100];
byte[] repuerto = new byte[1024];
String conex;
String rpta="llego";
int intnconexiones=0;
responde=rpta.getBytes();
//numero con quienes esta conectado
DatagramPacket numero = new DatagramPacket(bufer1, bufer1.length);
socketUDP.receive(numero);

System.out.println("inicio de sesión");
DatagramPacket mip = new DatagramPacket(bufer2, bufer2.length);
socketUDP.receive(mip);
String miip = new String(mip.getData());
System.out.println("nodo: "+miip);
int ncifras=0;
for(i=0;i<bufer1.length;i++)
```

```
{
if(bufer1[i]!=0)
{   ncifras++;   }
else
{break;}
}
System.out.println("nconecciones del nodo:"+ncifras);
double nconexiones=0;
//convierte a entero
for(i=0;i<ncifras;i++)
{
nconexiones=((bufer1[i]-48)*Math.pow(10,ncifras-1-i))+nconexiones;
}
System.out.println("numero de conexiones del nodo:  "+nconexiones);
intnconexiones=(int)(nconexiones);
tablapropiaconex=new String[intnconexiones];
propios_puertos=new int[intnconexiones];
for(i=0;i<nconexiones;i++)
{
// Construimos el DatagramPacket para recibir peticiones
DatagramPacket peticion = new DatagramPacket(bufer, bufer.length);

// Leemos una peticiÃ³n del DatagramSocket
socketUDP.receive(peticion);
            String frase = new String(peticion.getData());
            tablapropiaconex[i]=frase;
            //recibe el puerto
            DatagramPacket recibepuerto = new DatagramPacket(repuerto, repuerto.length);
// Leemos una peticiÃ³n del DatagramSocket
socketUDP.receive(recibepuerto);
            ncifras=0;
for(int ii=0;ii<repuerto.length;ii++)
{
if(repuerto[ii]!=0)
{   ncifras++;   }
else
{break;}
}
```

```
System.out.println(ncifras);
double puerto_f=0;
//convierte a entero
for(int ii=0;ii<ncifras;ii++)
{
puerto_f=((repuerto[ii]-48)*Math.pow(10,ncifras-1-ii))+puerto_f;
}
int intpuerto=(int)(puerto_f);
System.out.println("puerto abierto:"+intpuerto);
propios_puertos[i]=intpuerto;
// Construimos el DatagramPacket para enviar la respuesta
DatagramPacket respuesta =  new DatagramPacket(responde, responde.length,
peticion.getAddress(), peticion.getPort());
// Enviamos la respuesta, que es un eco
socketUDP.send(respuesta);
}
try{
System.out.println("tiempo de espera limite 5 seg");
Thread.sleep(5000);
}
catch(Exception e)
{
System.out.println(e);
}
System.out.println("cerrando conexion");
socketUDP.close();
}
}
}
```

ANEXO E: Programa del algoritmo de hormigas.

```
public class Hormigas{

//Variables del enrutamiento
int nodos;
int n;
double[][] distancias;
//Constantes
double
alfa = 0.85,
beta = 0.75,
ro = 0.5,
epsilon = 0.1,
fer_ini = 0.25;
//Información de las aristas y nodos
double[][] feromonas;
double[][] probabilidades;
int[][] vecinos;
//Iteraciones y numero de m
int m = 50;
int max_it = 40;
//Distancias
int nodo_inicial;
int nodo_final;
//Método Constructor
//Inicio m
Hormigas(double[][] d, int ini , int fin){
nodo_inicial=ini;
nodo_final=fin;
//Arreglo de distancia entre nodos
nodos = d[0].length;
n = nodos;
distancias = new double[nodos][nodos];
for ( int i = 0 ; i<nodos ; i++ ) {
for ( int j = 0 ; j<nodos ; j++ ) {
distancias[i][j] = d[i][j];
}
```

```
}
//Arreglo de Feromona con valores iniciales
feromonas = new double[nodos][nodos];
for ( int i = 0 ; i<nodos ; i++ ) {
for ( int j = 0 ; j<nodos ; j++ ) {
if(distancias[i][j] == 0.0)
feromonas[i][j] = 0.0;
else
feromonas[i][j] = fer_ini;
}
}
//Arreglo de probabilidades
probabilidades = new double[nodos][nodos];
for ( int i = 0 ; i<nodos ; i++ ) {
for ( int j = 0 ; j<nodos ; j++ ) {
if(distancias[i][j] == 0.0)
probabilidades[i][j] = 0.0;
else
probabilidades[i][j] = Math.pow(feromonas[i][j],alfa)*Math.pow(1.0/distancias[i][j],beta);
}
}

//Arreglo de vecinos
vecinos = new int[nodos][];
for ( int i = 0 ; i<nodos ; i++ ) {
int cont=0;
for ( int j = 0 ; j<nodos ; j++ ) {
if(distancias[i][j] != 0.0)
cont++;
}
vecinos[i] = new int[cont];
for ( int j = 0, k = 0 ; j<nodos ; j++ ) {
if(distancias[i][j] != 0.0){
vecinos[i][k] = j;
k++;
}
}
}
```

```
}//Fin Hormigas
//Inicia Actualizacion Global de Feromonas
public void Actualizacion(int[] mejorRuta){
double delta = 0.4;
for ( int i = 1 ; i<mejorRuta.length ; i++ ) {
int x = mejorRuta[i-1];
int y = mejorRuta[i];
feromonas[x][y] = ro*delta+(1.0-ro)*feromonas[x][y];
feromonas[y][x] = feromonas[x][y];
probabilidades[x][y] = Math.pow(feromonas[x][y],alfa)*Math.pow(1/distancias[x][y],beta);
probabilidades[y][x] = probabilidades[x][y];
}
}
//Finaliza Actualizacion
//Inicio Caminos
public int[] Caminos(){
//Arreglos de rutas para cada hormiga
int[][] rutas;
rutas = new int[m][n];
//Nodos visitados por cada hormiga
Boolean[][] visitados;
visitados = new Boolean[m][n];
//Indices
int[] indices;
indices = new int[m];
//Distancias de cada ruta
double[] recorridos;
recorridos = new double[m];
for ( int i = 0 ; i<m ; i++ ) {
indices[i] = 0;
recorridos[i] = 0;
for ( int j = 0 ; j<n ; j++ ) {
rutas[i][j] = n+1;
visitados[i][j] = false;
}
}
//Valores iniciales
for ( int i = 0 ; i<m ; i++ ) {
```

```
rutas[i][indices[i]] = nodo_inicial;
indices[i]++;
visitados[i][nodo_inicial] = true;
}
//Variaciones de nodo para cada hormiga
for ( int i = 0 ; i<m ; i++ ){
int nodo = nodo_inicial;
while (nodo != nodo_final) {
//Seleccionar Vecino
int elegir = (int)(Math.random()%vecinos[nodo].length);
if(elegir<vecinos[nodo].length && !visitados[i][vecinos[nodo][elegir]]){
rutas[i][indices[i]]=vecinos[nodo][elegir];
indices[i]++;
visitados[i][vecinos[nodo][elegir]] = true;
nodo = vecinos[nodo][elegir];
}
else{
for(int j = 0 ; j<vecinos[nodo].length ; j++){
if(!visitados[i][vecinos[nodo][j]]){
rutas[i][indices[i]]=vecinos[nodo][j];
indices[i]++;
visitados[i][vecinos[nodo][j]] = true;
nodo = vecinos[nodo][j];
break;
}
}
}
//Actualizacion Local x -> y
while(true){
int y = rutas[i][indices[i]-1];
int x = rutas[i][indices[i]-2];
feromonas[x][y] = epsilon*fer_ini+(1.0-epsilon)*feromonas[x][y];
probabilidades[x][y] = Math.pow(feromonas[x][y],alfa)*Math.pow(1/distancias[x][y],beta);
feromonas[y][x] = feromonas[x][y];
probabilidades[y][x] = probabilidades[x][y];
recorridos[i]=recorridos[i]+distancias[x][y];
break;
}
```

```
}
}
//Eleccion de mejor ruta
double best = recorridos[0];
int best_road = 0;
for ( int i = 1 ; i<m ; i++ ) {
if(recorridos[i]<best){
best=recorridos[i];
best_road = i;
}
}
Actualizacion(rutas[best_road]);
return rutas[best_road];
}//Fin Caminos
}
```