UNIVERSIDAD AUTÓNOMA METROPOLITANA

Unidad Iztapalapa

División de Ciencias Básicas e Ingeniería

"Estudio de los Efectos de la Cooperación

en la Evolución Estructural de Redes Complejas"

TESIS

que para obtener el grado de

MAESTRO EN CIENCIAS

(Ciencias y Tecnologías de la Información)

PRESENTA:

Lic. Jorge Antonio Muñoz García

DIRECTORES:

Dra. Daniela Aguirre Guerrero

Dr. Ricardo Marcelín Jiménez

JURADO:

Presidente: Dr. Roberto Bernal Jaquez

Vocal: Dr. Armando Castañeda Rojano

Secretaria: Dra. Daniela Aguirre Guerrero

Iztapalapa, Ciudad de México, México

Octubre 11, 2023

Resumen

Las redes complejas, emergen naturalmente en muchos sistemas de diversas áreas del conocimiento, en particular, en el área de Tecnologías de la Información (TI), las podemos encontrar al describir la topología subyacente de la Internet, la World Wide Web (WWW), redes sociales, redes par a par (P2P, por sus siglas en inglés), etc. En las últimas décadas, el estudio de las redes complejas se ha convertido en una importante área de investigación, debido a que, gracias a ellas, se pueden modelar las interacciones entre los componentes de los sistemas que representan. Además, como toda red, tienen ciertas propiedades estructurales, tales como su longitud de trayectoria promedio, diámetro, coeficiente de agrupamiento promedio, modularidad o su distribución de grados, las cuales, determinan algunas de sus propiedades funcionales, tales como su robustez ante fallas aleatorias y ataques dirigidos, velocidad de transmisión de datos, etc.

Esta investigación, propone experimentos basados en conexión y reconexión de aristas, mediante los cuales, un conjunto de redes con topología de malla o anillo, evoluciona hasta obtener características encontradas en las redes complejas. De estos experimentos, emergieron redes con un alto grado de robustez ante fallas aleatorias y ataques secuenciales dirigidos y que, además, en comparación con sus medidas iniciales, cuentan con diámetros y longitudes de trayectoria promedio considerablemente reducidos; un alto coeficiente de agrupamiento, y muestran estructuras de comunidades al contar con una modularidad alta.

El desarrollo de esta investigación, se encuentra dividido en dos etapas:

- Etapa 1 ***"Formación de Redes Complejas"***: En esta investigación, se propone modelar a una red compleja como un sistema distribuido, donde sus vértices, son entidades que procesan información, y sus aristas, son los canales de comunicación que permiten a los vértices interactuar entre sí. Tomando como base la idea de que los sistemas distribuidos pueden ser simulados en computadora, se escribió un simulador de eventos discretos en el lenguaje de programación "python", bajo el paradigma de la programación orientada a objetos, el cual, sigue el modelo de programación distribuida por paso de mensajes. Sobre el simulador, se ejecutaron una serie de experimentos, donde, partiendo de una red inicial con topología de malla o de anillo, cada uno de sus vértices aplica un conjunto de sencillas reglas locales de reconexión de aristas, que le permitirán encontrar atajos que lo conecten a regiones lejanas a su entorno local inicial, modificando así, la topología subyacente de la red. Lo anterior, permitió hacer un estudio riguroso sobre los efectos que tiene la cooperación en la evolución estructural de las redes iniciales, que al ser sometidas a diversos procesos de reconexión de aristas, experimentan cambios hasta el punto en el que sus propiedades estructurales convergen a propiedades comúnmente encontradas en las redes complejas.

- Etapa 2 ***"Degradación de Redes Complejas"***: En esta etapa, se midió la robustez (capacidad de una red de seguir operando bajo circunstancias adversas) de las redes obtenidas en la etapa 1, ante dos diferentes procesos de degradación: *fallas aleatorias*, bajo la selección de vértices con base en una función de distribución de probabilidad uniforme, y *ataques secuenciales dirigidos*, bajo la selección de vértices con base en su número de aristas incidentes, en orden descendente. Lo anterior, se llevó a cabo mediante el uso de una métrica denominada "promedio de la fiabilidad media entre dos terminales" (μ-A2TR, por sus siglas en inglés), la cual, entrega el umbral de robustez ante ambos procesos de cada una de las redes. Los resultados de esta segunda etapa, muestran que la robustez de las redes emergentes en la etapa 1, varía conside-

rablemente de acuerdo a las configuraciones de diversos parámetros utilizados en los experimentos, tales como el algortimo de encaminamiento, la regla local de reconexión de aristas, la distancia máxima que puede alcanzar una arista dinámica, la topología inicial subyacente en la red y el número máximo de conexiones que un vértice puede aceptar.

Palabras Clave: Algoritmos Distribuidos, Ciencia de Redes, Ciencias de la Complejidad, Simulación de Eventos Discretos.

Abstract

Complex networks, emerge naturally in many systems of different knowledge areas, in particular, in the Information technologies area, we can find them by describing the underlying topology of the Internet, the World Wide Web, social networks, peer to peer networks, and more. In recent decades, the study of complex networks has become in an important research area, because of, thanks to them, the interactions between the components of the systems they represent can be modeled. Moreover, like any network, they have certain structural properties, such as their average path length, diameter, average clustering coefficient, modularity or their degree distribution, which determine some of their functional properties, such as their robustness to random failures and targeted attacks, data transmission speed, and more.

This research, proposes experiments based on connection and reconnection of edges, through which a set of networks with mesh or ring topology, evolves until obtaining characteristics found in complex networks. From these experiments, networks emerged with a high degree of robustness against random failures and targeted sequential attacks and, moreover, compared to their initial measures, have considerably reduced diameters and average path lengths; a high clustering coefficient and show a community structure by having a high modularity.

The development of this research is divided in two stages:

- Stage 1 ***"Complex Network Formation"***: In this research, we propose to model a complex network as a distributed system, where its vertices are entities that process information, and its edges are the communication channels that allow the vertices to communicate with each other. Based on the idea that the distributed systems can be simulated on a computer, a discrete event simulator was written in the "python" programming language, under the oriented-object programming paradigm, which, follows the distributed programming model by messages passage. On the simulator, a series of experiments were executed, where, starting with an initial network with a mesh or ring topology, each of its vertices applies a set of easy rules of edge reconnection, which will allow it to find shortcuts that connect it to far regions from its initial local environment, thus modifying the underlying topology of the network. This made it possible to carry out a rigorous study on the effects of cooperation on the structural evolution of the initial networks, which, when subjected to various processes of edges reconnection, undergo changes to the point where their structural properties converge to properties commonly found in complex networks.

- Stage 2 ***"Complex Networks Degradation"***: In this stage, the robustness (ability of a network of still operating under adverse circumstances) of the networks obtained in the stage 1 was measured, under two different degradation processes: *random failures*, under the selection of vertices based on a uniform probability distribution function, and *directed sequential attacks*, under the selection of vertices based on their number of incident edges, in descending order. This was carried out through the use of a metric denominated "mean of the average two terminal reliability", which gives the robustness threshold to both processes of each network. The results of this second stage, show the robustness of the emerging networks in the stage 1, vary considerably according to the configurations of diverse parameters used in the experiments, such as the routing algorithm, the local rule of edge reconnection, the maximum distance that can be reached by a dynamic edge, the underlying initial topology in the network and the maximum number of connections that a vertex can accept.

Keywords: Distributed Algorithms, Network Science, Complexity Sciences, Discrete Event Simulation.

Agradecimientos

Esta idónea comunicación de resultados, refleja el amor y la pasión que siento hacia mi carrera.

Infinitas gracias...

A él, que todo lo sabe, que está contigo cuando más lo necesitas, que es la grandeza de nuestro existir, que me otorgó la dicha de la vida y que me dio las habilidades, actitudes y aptitudes necesarias para llegar hasta este punto en mi vida profesional, a él, le regalo esta idónea comunicación de resultados.

A ella, que a través de su ternura y amor, me cuida y guía en cada paso que doy, y que, con su manto divino me protege en todas partes, logando eliminar toda preocupación en mí, a ella, le regalo, también, esta idónea comunicación de resultados.

A ella, que me dio la vida, que fomenta en mí las ganas de crecer día a día, que me regaña, motiva, cuida y protege, por todo su esfuerzo y dedicación y que, en todo momento, me apoya y ve por mi bienestar, le regalo mi alma y mi corazón. Te amo mamá.

A él, que me dio la vida, que a base de sacrificios y trabajo me ha apoyado para llegar hasta este punto en mi vida personal y profesional, y que siempre está cuando lo necesito, brindándome sus consejos y su sabiduría, mil besos y abrazos. Te amo papá.

A ella, por siempre estar conmigo, por las risas, enojos, momentos de diversión y sobre todo, por ser parte de mi vida. Te amo hermanita.

A ella, que desde el cielo me cuida, sé que a diario me manda bendiciones para que me vaya y esté bien. Te amo y te mando mil besos donde quiera que estés, Tazi.

A la Universidad Autónoma Metropolitana, por recibirme y apoyarme para realizar mis estudios de posgrado, por brindarme la oportunidad de estudiar el idioma inglés en la Universidad de Nuevo México y por todas las experiencias académicas y profesionales que viví durante mi estancia en sus instalaciones. Un placer ser parte de esta maravillosa y prestigiosa casa de estudios.

A ella, por todo su apoyo incondicional para realizar esta investigación, por sus consejos, por la confianza que me ha brindado, por todos los conocimientos que me regaló, por siempre estar para mí y, sobre todo, por aceptar ser mi asesora. Abrazos, doctora Daniela.

A él, por recibirme en sus clases, por inspirarme a ingresar al posgrado, por aceptarme en su equipo de trabajo, por la confianza brindada hacia mi persona y por todo el apoyo que me dio durante mi estancia en la UAM. Abrazos, doctor Ricardo.

A los doctores Magali López, José Luis Quiroz, Eric Rincón y Román Mora, por su activa participación y ayuda para el desarrollo de esta investigación.

Sé que cuento con todos ustedes en este camino tan difícil que es el de trascender.

No te preocupes tanto por lo que no tienes, preocúpate por engrandecer lo que tienes

Su amigo de siempre...

Jorge Antonio

Talleres y concursos

Esta investigación, fue presentada en los siguientes talleres:

1. "Society for Industrial and Applied Mathematics Workshop on Network Science (NS22)", Northwestern University, Evanston Illinois, September 13-15, 2022.

2. "UAM Cuajimalpa es tu casa", Universidad Autónoma Metropolitana, Unidad Cuajimalpa, Ciudad de México, 10, 11, 17 y 18 de Noviembre, 2022.

Así como en el concurso:

1. "UAM Cuéntame tu tesis, 2ª edición", Universidad Autónoma Metropolitana, Unidad Iztapalapa, Ciudad de México, Mayo, 2023.

Contenido

Lista de Figuras

Lista de Tablas

Lista de Algoritmos

Capítulo 1

Introducción

"De alguna manera, nada podría ser tan simple como una red, ... , una red no es más que una colección de objetos conectados unos con otros de alguna manera" -Duncan J. Watts [6]

1.1. Motivación

Las redes, están presentes en muchos aspectos de nuestras vidas: redes de amigos, de telecomunicaciones, de computadoras, de transporte y la Web, son ejemplos de sistemas con los que interactuamos externamente, mientras que las células de nuestro cerebro y las proteínas dentro de nuestro cuerpo, forman redes que determinan nuestra supervivencia, las cuales, son internas a nuestros organismos. Cuando las personas usan Facebook o Twitter, compran mercancía en Amazon, hacen búsquedas en Google, o compran un boleto de avión, usan a las redes sin saberlo [1]. En cierto sentido, cada aspecto de la realidad parece estar compuesto por un conjunto de elementos que interactúan entre sí. Podemos entender a las redes como una representación de las interacciones entre los componentes de un sistema. En particular, las redes de telecomunicaciones, se han convertido en una pieza clave de infraestructura para la sociedad, debido a que facilitan la comunicación y el intercambio de recursos, ideas, bienes y servicios. Las redes de telecomunicaciones, deben ser confiables, en el sentido de que al enviar un mensaje a algún destino (no importando que tan lejos se encuentre

del origen), se debe asegurar que el mensaje llegue al destino y que, además, se entregue en el menor tiempo posible. Por otro lado, estas redes deben de operar de una manera tal que si existieran fallas de algunos de sus componentes, o si se les presentara un ataque deliberado, deberían seguir trabajando de una manera transparente, es decir, el usuario no notaría que algo está sucediendo.

En diversas áreas del conocimiento, aparecen sistemas que presentan fenomenología emergente, también llamada *compleja*, tal que no puede derivarse simplemente del entendimiento de las propiedades y leyes que rigen a cada uno de sus componentes. La emergencia de tal fenomenología, es el resultado de las interacciones entre los componentes de tales sistemas. Aunque no existe una definición universalmente aceptada de *sistema complejo*, la mayoría de los investigadores, describirían un sistema de agentes conectados que exhibe un comportamiento global emergente, no impuesto por un control central, sino resultante de las interacciones no lineales entre sus agentes, como *complejo*. Estos agentes, pueden ser insectos, pájaros, personas, o empresas, y su número puede oscilar entre cientos y millones [11]. Los sistemas complejos pueden, naturalmente, ser descritos como redes complejas, donde los componentes del sistema, son representados por los vértices de la red y las interacciones entre los componentes, son representadas por las aristas que conectan a los vértices en la red [2]. En particular, dentro del área de TI, podemos representar a la Internet como una red compleja, donde sus vértices, son computadoras y dispositivos de encaminamiento, mientras que sus aristas, son las conexiones que les permiten tener comunicación unos con otros. Otro ejemplo, sería la World Wide Web, la cual, se puede representar mediante una red, donde sus vértices, serían las páginas web y sus enlaces, los hipervínculos que permiten que las páginas interactúen entre sí.

Una red compleja, puede describirse en términos de la teoría de gráficas, como una colección de objetos interconectados, la cual, puede ser representada en términos matemáticos y visuales con un diagrama, llamado gráfica. Para que el lector realice una revisión, y obtenga comprensión de la notación y los conceptos relacionados con la teoría de gráficas utilizados

en esta investigación, se sugiere, consulte el Apéndice A. Los elementos de la red compleja, son denominados vértices o nodos, y sus conexiones, aristas o enlaces. Una red, produce una descripción general del sistema bajo investigación. Los vértices o nodos, son los elementos de interés dentro del sistema a estudiar, mientras que las aristas o enlaces, representan sus mutuas interacciones. En general, no existe una receta única para identificar a los vértices y aristas de un sistema complejo, por el contrario, corresponde a los investigadores concebir el modelo más adecuado para una situación específica [16]. Sobre las gráficas que representan a las redes, se pueden realizar diversos cálculos matemáticos para obtener información relevante acerca del sistema que representan. Por lo anterior expuesto, en esta investigación, se hace uso de la *Ciencia de Redes*, la cual, a su vez, usa conceptos de la *Teoría de Gráficas*, de las *Ciencias de la Complejidad* y de las *Ciencias de la Computación*, en búsqueda de modelos y métricas, que permitan tener un claro entendimiento de los efectos que emergen de la interacción entre los elementos que intervienen en ellas, y así, obtener conocimiento acerca de los sistemas complejos que dichas redes representan.

Esta investigación, da continuidad a un trabajo previo [12], proponiendo una serie de mecanismos de formación de redes complejas, basados en la toma de decisiones locales de sus componentes y la cooperación e interacción entre ellos, las cuales, modelan sistemas de telecomunicaciones. Anteriormente, se han propuesto modelos de formación de redes complejas basados en propiedades de conexión-reconexión de aristas y conexión preferencial. En este trabajo, se proponen mecanismos de formación tales que unen las propiedades anteriores, en búsqueda de redes con características muy similares a las encontradas en las redes complejas, estudiando el efecto de la estructura del grafo inicial [36], el número de conexiones que soportan los vértices, la distancia más lejana a la que pueden conectarse-reconectarse, el algoritmo de encaminamiento y las reglas de recableado propuestas. Más aún, cada una de las redes formadas, es sometida a diferentes procesos de degradación, con el objetivo de medir su robustez ante tales procesos.

1.2. Objetivos

1.2.1. Objetivo general

Estudiar los efectos de la cooperación en la evolución estructural de redes complejas.

1.2.2. Objetivos particulares

1) Reconocer las principales propiedades estructurales que caracterizan el estado de una red.

2) Reconocer parámetros locales de los que pueden emerger propiedades globales comúnmente encontradas en las redes complejas.

3) Proponer, al menos, un mecanismo que permita describir la formación de una red en el que intervengan vértices cooperadores y semi-cooperadores.

4) Desarrollar una herramienta de simulación sobre la cual se ejecutarán los experimentos que permitirán caracterizar la evolución estructural de las redes propuestas.

5) Caracterizar la evolución estructural de un conjunto de redes mientras son sometidas a diversos procesos de conexión y reconexión de aristas.

6) Evaluar la robustez de las redes obtenidas mientras son sometidas a diversos procesos de degradación.

Capítulo 2

Fundamentos Teóricos

"Colonias de hormigas, tráfico en carreteras, economías de mercado, sistemas inmunitarios - en todos estos sistemas, los patrones no están determinados por una autoridad centralizada, sino por interacciones locales entre componentes descentralizados" -Mitchel Resnick [28]

2.1. Ciencia de redes

2.1.1. Introducción

Existen muchos sistemas de interés científico, tales que están conformados por un conjunto de elementos individuales enlazados de alguna manera buscando cumplir algún objetivo. Las personas, constantemente interactuamos con tales sistemas y, tal vez, ignoramos su estructura y solo nos enfocamos en el resultado que proveen. En este sentido, podemos pensar en la Internet, la cual es un sistema que interconecta millones de computadoras, y del cual muchas personas actualmente hacen uso prácticamente a diario. Por otro lado, podemos pensar en la red de individuos en la que estamos inmiscuidos, la cual, es llamada sociedad y que no es otra cosa que un conjunto de personas unidas por lazos de amistad o interacción social. A menudo, un sistema puede estar compuesto de otros sistemas, por ejemplo, la sociedad requiere de la interacción y cooperación de millones de personas para poder operar. Asimis-

mo, para lograr tal interacción, se requiere de infraestructuras de comunicación, tales como la Internet, o redes de telefonía móvil, y a su vez, estas infraestructuras también requieren de millones de dispositivos electrónicos para operar de manera correcta y cumplir con su objetivo. Por otro lado, nuestro cerebro, para poder crear conocimiento, inteligencia y capacidad para razonar, hace uso de millones de neuronas que se encuentran interconectadas dentro de el, y que trabajando en conjunto, logran tal objetivo. Estos son solo unos cuantos ejemplos de sistemas de la vida real que también son llamados *sistemas complejos*, los cuales, son sistemas que no pueden ser completamente entendidos, si lo único con lo que contamos es la perfecta descripción de cada uno de sus elementos. Los sistemas complejos, juegan un papel fundamental en nuestra vida diaria, debido a que pueden ser encontrados en diversas áreas del conocimiento, tales como Biología, Matemáticas, Economía, Tecnologías de la Información, etc. El patrón de interacciones de un sistema complejo arbitrario, puede ser representado como una red. Una *red*, es en su forma más simple, un conjunto de puntos unidos mediante líneas, bajo algún criterio. En la jerga de la ciencia de redes, los puntos que conforman a una red, son llamados vértices o nodos, mientras que las líneas que los interconectan, son llamadas aristas o enlaces [10]. Un ejemplo de red con 7 vértices y 10 aristas, se muestra en la Figura 2.1. Los vértices y aristas de una red, pueden ser etiquetados con información adicional del sistema que representan. Más aún, pueden existir diferentes tipos de vértices y de aristas en una red, por ejemplo, piense en la red que modela las interacciones entre los elementos de la sociedad, sus vértices pueden representar hombres o mujeres, personas de diferentes nacionalidades, ubicaciones, edades, ingresos, etc. Mientras que sus aristas, pueden representar amistad, conocimiento profesional o proximidad geográfica. Las aristas, también pueden llevar pesos, representando, por decir, que tan estrecha es la relación entre pares de personas [13].

Ejemplos de redes del mundo real incluyen [13]:

1) *Redes Sociales*: Una red social, es un conjunto de personas o grupos de personas, con algún patrón de contactos o interacciones entre ellas, por ejemplo, las redes de amistad,

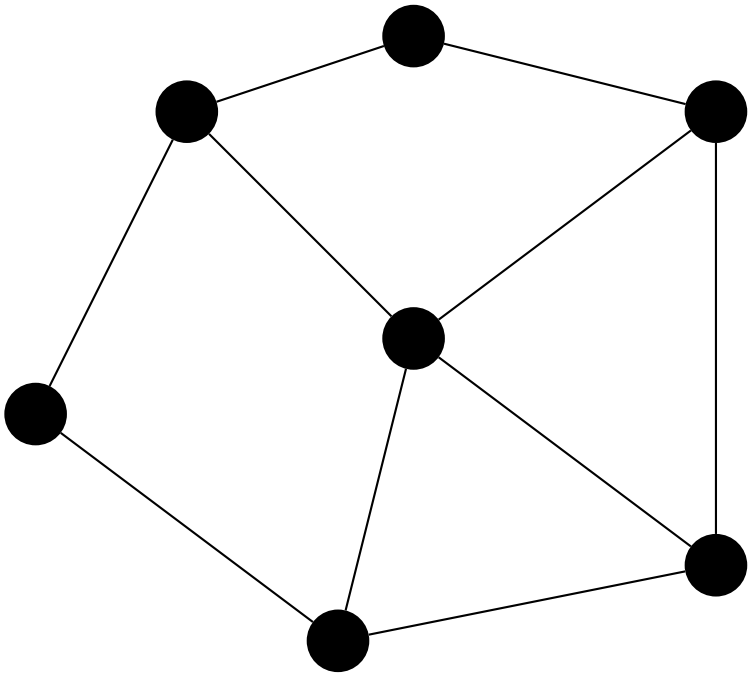


Figura 2.1: Ejemplo de red con 7 vértices y 10 aristas

de relaciones entre organizaciones, familiares, etc. Las redes sociales representan a las personas como sus vértices, y las interacciones entre ellas mediante sus aristas.

2) *Redes de Información*: Las redes de información, son también llamadas *redes de conocimiento*. El ejemplo clásico de este tipo de redes, es la red de citas entre artículos científicos. Las citas, forman una red en la cual los vértices representan a los artículos, y las aristas las citas entre ellos. La estructura de la red de citas, refleja la estructura de la información almacenada en sus vértices, de ahí el término "red de información". Otro ejemplo muy importante, es la World Wide Web, la cual, es una red de páginas Web que contienen información, enlazadas unas a otras mediante hipervínculos. La Web, no debe confundirse con la Internet, la cual es una red física de computadoras enlazadas a través de enlaces de fibra óptica y otras conexiones de datos.

3) *Redes Tecnológicas*: Las redes tecnológicas, son redes artificiales diseñadas normalmente para la distribución de algún producto o recurso, como la electricidad o información.

Por ejemplo, la red eléctrica, es una red de líneas de transmisión trifásicas de alta tensión que se extiende por un país o parte de el. Otro ejemplo importante en este rubro, es la Internet.

4) *Redes Biológicas*: Diversos sistemas biológicos, pueden ser representados como redes, por ejemplo, considere una red trófica, en la que los vértices representan las especies de un ecosistema y una arista dirigida de la especie A hacia la especie B, indica que A es un depredador de B.

De acuerdo con [5] se puede realizar una clasificación de las redes como sigue:

1) ***Redes simples***: Una red simple, es una red que podemos describir completamente de manera analítica. Sus propiedades estructurales son exactas y triviales. Se puede tener una fórmula simple que devuelva la información que se requiere saber. Para ejemplificar este tipo de redes, piense en una red $G = (V, E)$, con topología malla cuadrada, con $i \times i$ vértices, la cual, es una red en la que cada vértice $v \in V$ está conectado a sus cuatro vecinos más cercanos, tal y como se muestra en la Figura 2.2. Algunas de sus propiedades estructurales, son definidas como sigue:

   a) Diámetro: $D_G = 2(i - 1)$.

   b) Coeficiente de agrupamiento: $c_v = 0 \ \forall \ v \in V$.

   c) Coeficiente de agrupamiento promedio: $C_G = 0 \ \forall \ i \in \mathbb{N}$.

   d) Orden: $n = i^2$.

   Al conocer la regla de conexión y las fórmulas mostradas previamente, se puede representar cualquier malla cuadrada que se nos venga a la imaginación, debido a este hecho, es denominada una red simple.

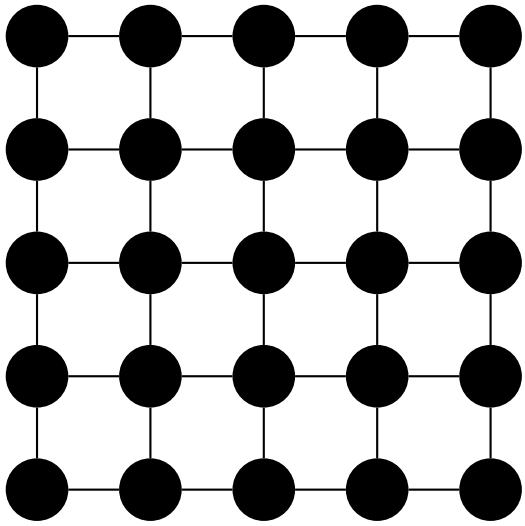


Figura 2.2: Malla con 5×5 vértices

2) ***Redes complejas***: Las redes complejas, modelan las interacciones entre los elementos de los sistemas complejos. De dichas interacciones, emergen propiedades globales que no son solo la suma de las propiedades locales de sus elementos. Debido a esto, las redes complejas, no pueden ser descritas por una simple regla de conexión (como el caso anterior de la malla), ni un conjunto de fórmulas, de tal suerte que, incluso conociendo todas sus reglas de conexión, algunas propiedades pueden emerger de manera imprevista. Las redes complejas, representan sistemas cuya complejidad está comprendida entre estructuras regulares y aleatorias. Más concretamente, el campo de las redes complejas aparece en la última década del siglo pasado con dos artículos seminales, el de Duncan Watts y Steven Strogatz sobre redes de mundo pequeño [2], y el de Albert Lázló Barabási y Réka Albert sobre redes libres de escala [17].

A continuación, se enuncian algunos ejemplos de redes complejas que representan a diversos sistemas complejos [4]:

1) La red que codifica las interacciones entre genes, proteínas y metabolitos, integra estos componentes en las células vivas. La existencia misma de esta *red celular* es un prerrequisito para la vida.

2) El diagrama de conexiones entre neuronas, llamado *red neuronal*, es la clave para entender el funcionamiento del cerebro y nuestra conciencia.

3) La red formada por los lazos profesionales, de amistad y familiares de las personas, a menudo llamada *red social*, es el tejido de la sociedad y determina la difusión de conocimientos, comportamientos, recursos e, incluso, la propagación de enfermedades.

4) Las *redes de comunicaciones*, que describen los dispositivos de comunicación que interactúan entre sí, a través de conexiones cableadas de Internet o enlaces inalámbricos, constituyen el núcleo del sistema de comunicación moderno.

5) La *red eléctrica*, una red de generadores y líneas de transmisión, suministra energía a prácticamente todas las sociedades modernas.

6) Las *redes comerciales*, mantienen nuestra capacidad de intercambiar bienes y servicios.

Una red compleja difiere de una gráfica debido a que describe también las interacciones presentes entre sus elementos, mientras que una gráfica, en general, no tiene por que estar asociada a un sistema complejo. Como se mencionó anteriormente, las redes complejas pueden modelar diversos sistemas complejos. La Tabla 2.1, muestra un resumen de ejemplos de redes complejas encontradas en la vida real [14].

Bajo este contexto, surge la necesidad inminente de comprender el comportamiento los sistemas complejos, examinando a detalle las redes que los representan. Los científicos, a lo largo de los últimos años, han desarrollado un conjunto extenso de herramientas matemáticas, computacionales y estadísticas para analizar, modelar y entender a las redes, lo cual da vida a la *"Ciencia de Redes"*. Estas herramientas, comúnmente, representan a los sistemas como redes, y sobre de ellas ejecutan una serie de cálculos matemáticos, los cuales, dan información útil, por ejemplo, cual es el vértice más conectado en la red, o simplemente, el camino más corto entre cualesquiera dos vértices de la red. Por otro lado, existen herramientas que buscan

Tabla 2.1: Resumen de ejemplos de redes complejas de la vida real

Red compleja	**Vértices**	**Aristas**
Red de actores	Actores	Actuar en la misma película
Red de colaboraciones	Científicos	Coautores en una publicación
Red de citas bibliográficas	Artículos científicos	Citación
Red de Facebook	Personas	Amistad en Facebook
Redes metabólicas	Metabolitos	Aparecer en la misma reacción
Redes de interacción de proteínas	Proteínas	Interacción física
Red cerebral	Neuronas	Conexiones sinápticas
Internet	Computadoras y dispositivos de encaminamiento	Conexiones físicas
World Wide Web	Páginas Web	Direcciones URL
Redes de aeropuertos	Aeropuertos	Vuelos entre aeropuertos
Redes eléctricas	Centrales eléctricas	Conexiones eléctricas

predecir comportamientos futuros en los sistemas que las redes representan, por ejemplo, la propagación de enfermedades, el tráfico que fluye a través de la Internet, etc.

La estructura y la evolución de las redes que soportan a los sistemas complejos, sin considerar sus diferencias en forma, tamaño y naturaleza, se rigen por un conjunto compartido de leyes y principios fundamentales. De acuerdo con [5], la ciencia de redes, es definida como sigue:

"La ciencia de redes, es la búsqueda del entendimiento de la forma de un sistema complejo, a través de la inspección de las relaciones entre sus componentes interactuantes"

A pesar de que la mayoría de las redes mostradas anteriormente no son relativamente nuevas, la ciencia de redes emerge a principios del siglo XXI, gracias al interés que surgió en esa época hacia las redes, basado en el gran número de citas que recibieron dos artículos seminales: en 1959 Paul Erdös y Alfréd Rényi publicaron un artículo titulado "Sobre gráficas aleatorias" [8], el cual, marca el inicio del estudio de las redes aleatorias en la teoría de gráficas. Por otro lado, Mark Granovetter publicó su trabajo titulado "La fuerza de los vínculos débiles" [9], el cual se ha convertido en el artículo referente a redes sociales más

citado [4].

2.1.2. Características

De acuerdo con [4], existe un conjunto de características que definen a la ciencia de redes, las cuales le permiten a dicha ciencia comprender a los sistemas complejos:

1) ***Naturaleza Interdisciplinaria***: La ciencia de redes, ofrece un lenguaje a través del cual distintas disciplinas pueden interactuar sin problemas entre sí. Tanto los biólogos celulares como los científicos del cerebro y los informáticos, se enfrentan a la tarea de caracterizar el diagrama de cableado que hay detrás de su sistema, extraer información de conjuntos incompletos y comprender la robustez de sus sistemas frente a fallos o ataques.

2) ***Naturaleza empírica y basada en datos***: Varios conceptos clave de la ciencia de redes, tienen sus raíces en la teoría de gráficas, un campo de las matemáticas discretas. Lo que distingue a la ciencia de redes de la teoría de gráficas, es su naturaleza empírica, es decir, su enfoque en los datos, la función y la utilidad. La ciencia de redes no se conforma con solo aplicar herramientas matemáticas abstractas de la teoría de gráficas y las ciencias de la computación para describir una determinada propiedad de la red. Cada herramienta que se aplica, se pone a prueba con datos reales y su valor se juzga por los conocimientos que ofrece sobre las propiedades y el comportamiento de un sistema.

3) ***Naturaleza cuantitativa y matemática***: Para contribuir al desarrollo de la ciencia de redes y el uso adecuado de sus herramientas, es esencial dominar el formalismo matemático que la sustenta. La ciencia de redes tomó prestado de la teoría de gráficas el formalismo para tratar con las gráficas y de la física estadística el marco conceptual para tratar con la aleatoriedad y buscar principios universales organizadores. Últimamente, este campo se está beneficiando de conceptos prestados de la ingeniería, como la teoría

del control y la información, que permiten entender los principios de control de las redes; y de la estadística, que ayuda a extraer información de conjuntos de datos incompletos y ruidosos.

4) ***Naturaleza computacional***: Dado el tamaño de muchas de las redes de interés práctico, y la cantidad excepcional de datos auxiliares que hay detrás de ellas, los científicos de las redes se enfrentan regularmente a una serie de formidables retos computacionales. De ahí que este campo tenga un marcado carácter computacional, con usos de algoritmos, gestión de bases de datos, y minería de datos. Existe una serie de herramientas informáticas para abordar estos problemas computacionales, lo que permite a profesionales con diversos conocimientos computacionales analizar las redes que les interesan.

Sobre las redes ocurren procesos dinámicos, de los cuales emerge un comportamiento que les dan forma conforme avanza el tiempo [17]. La ciencia de redes busca comprender el comportamiento emergente, analizando a fondo los efectos que emergen de la topología de la red en sus procesos dinámicos y, a la vez, prediciendo sus comportamientos futuros. Para lograr lo anterior, la ciencia de redes busca construir modelos que reproduzcan las propiedades de redes reales. Desde una perspectiva de modelación, una red es un objeto relativamente simple, que consiste solo de vértices y aristas. El reto real, sin embargo, es decidir donde colocar las aristas entre los vértices para que se pueda reproducir la complejidad de un sistema real [4]. A lo largo de la historia, han surgido diferentes modelos de redes a partir del surgimiento de la teoría de gráficas. La cronología de las redes complejas, se muestra a continuación:

- 1959: El modelo de red aleatoria propuesto por Paul Erdös y Alfréd Rényi.
- 1967: El experimento de Stanley Milgram, el cual, da origen al fenómeno de los mundos pequeños.

- 1998: El modelo de red de mundos pequeños propuesto por Duncan Watts y Steven Strogatz.

- 1999: El modelo de red con distribución de ley de potencias propuesto por Albert Lázló Barabási y Réka Albert.

A continuación, se detallan cada uno de los hitos mostrados en la cronología anterior [11, 4, 14, 18, 8, 2, 17, 19]:

2.1.3. Redes aleatorias

El estudio de las redes aleatorias, fue iniciado en 1959 por dos matemáticos Erdös y Rényi (ER), con el propósito original de estudiar, mediante métodos probabilísticos, las propiedades de las gráficas en función de un incremento aleatorio de conexiones en la gráfica. Aunque la intuición indica claramente que muchas redes complejas de la vida real no son ni totalmente regulares, ni completamente aleatorias, el modelo ER fue el único enfoque sensato y riguroso que dominó el pensamiento de los científicos acerca de las redes complejas en la segunda mitad del siglo XX. El modelo de red aleatoria, fue introducido de manera independiente por Edgar Nelson Gilbert [39] el mismo año que Erdös y Rényi publicaron su primer artículo sobre el tema. Sin embargo, el impacto del trabajo de Erdös y Rényi es tan abrumador que se les considera los fundadores de la teoría de redes aleatorias.

Una *red aleatoria* G, consiste en $N = n$ vértices, donde cada pareja de vértices es conectada con una probabilidad p. Existen dos definiciones de una red aleatoria:

1) Modelo Erdös - Rényi $G = (N, L)$: N vértices etiquetados son conectados con L aristas colocadas aleatoriamente.

2) Modelo Edgar Nelson Gilbert $G = (N, p)$: Cada pareja de N vértices etiquetados, es conectada con una probabilidad p.

Note que el modelo $G = (N, p)$, fija la probabilidad p de que dos vértices estén conectados, mientras que el modelo $G = (N, L)$, fija el número total de aristas L. En las redes de la vida real, el número de aristas raramente se mantiene fijo, por lo que a continuación se explorará a detalle el modelo $G = (N, p)$.

El Algoritmo 1, muestra los pasos para construir una red aleatoria con base en el modelo Edgar Nelson Gilbert $G = (N, p)$.

Algoritmo 1 Construcción de una red aleatoria con base en el modelo Edgar Nelson Gilbert

Requiere:

Una gráfica $G = (V, E)$ con $N = n$ vértices aislados y $E = \emptyset$.

Una lista K con las $\frac{N(N-1)}{2}$ posibles parejas de vértices de G.

Un valor p que representa la probabilidad de conexión.

1: $lista \leftarrow K$
2: $probabilidad \leftarrow p$
3: **Para** $i = 1$ hasta $\frac{N(N-1)}{2}$ **Hacer**
4: $aleatorio \leftarrow\sim U[0, 1]$
5: **Si** $aleatorio \geq probabilidad$ **Entonces**
6: Genera una arista incidente en la i-ésima pareja de vértices contenida en $lista$
7: **Sino**
8: Mantener desconectada la i-ésima pareja de vértices contenida en $lista$
9: **Termina Si**
10: **Termina Para**

Al generar diversas realizaciones de redes aleatorias con los mismos parámetros N, p, se tendrá un número diferente de aristas, por lo que resulta útil determinar cuántas aristas se esperan para una realización en particular de una red aleatoria con parámetros N, p fijos.

La probabilidad de que una red aleatoria tenga exactamente L aristas, se define como sigue:

$$P_L = \binom{\frac{N(N-1)}{2}}{L} p^L (1-p)^{\frac{N(N-1)}{2}-L} \tag{2.1}$$

Donde:

- p^L representa la probabilidad de que L de los intentos de conectar los $\frac{N(N-1)}{2}$ pares de vértices hayan resultado en una arista.
- $(1-p)^{\frac{N(N-1)}{2}-L}$ representa la probabilidad de que los $\frac{N(N-1)}{2} - L$ intentos no hayan resultado en una arista.
- $\binom{\frac{N(N-1)}{2}}{L}$ representa el número de diferentes maneras en que se pueden establecer L aristas entre $\frac{N(N-1)}{2}$ parejas de vértices.

Note que la Ecuación 2.1, es análoga a una distribución Binomial, por lo que el número esperado de aristas (de acuerdo a las propiedades de tal distribución de probabilidad), denotado $\langle L \rangle$, en una red aleatoria es:

$$\langle L \rangle = \sum_{L=0}^{\frac{N(N-1)}{2}} L p_L = p\frac{N(N-1)}{2} \tag{2.2}$$

Sean $m = \langle L \rangle$ y $n = N$, para determinar el grado medio $\langle \delta \rangle$ de una red aleatoria, se puede sustituir la Ecuación 2.2 en la Ecuación A.1 y obtener:

$$\langle \delta \rangle = \frac{2m}{n} = p(n-1) \tag{2.3}$$

Lo cual, indica que el grado medio de una red aleatoria está dado por el producto de la probabilidad p de que dos vértices estén conectados y el número máximo de aristas incidentes $(n-1)$ que un vértice puede tener en una red de orden n. Por otro lado, a medida que se

incrementa p, la red se vuelve más densa.

La probabilidad de que un vértice v tenga exactamente k aristas incidentes, se define como sigue:

$$P_k = \binom{N-1}{k} p^k (1-p)^{N-1-k} \tag{2.4}$$

Donde:

- p^k representa la probabilidad de que k de las aristas incidentes en v estén presentes.
- $(1-p)^{N-1-k}$ representa la probabilidad de que las $N-1-k$ aristas no estén presentes en v.
- $\binom{N-1}{k}$ representa el número de diferentes maneras que se pueden seleccionar k aristas de $N-1$ aristas potenciales que un vértice puede tener.

Lo cual indica que la distribución de grados de una red aleatoria, sigue una distribución Binomial con parámetros (N, p).

En muchas redes reales, se cumple que $\langle\delta\rangle \ll N$. En este límite, la distribución de grados mostrada en la Ecuación 2.4, es bien aproximada por una distribución de Poisson con media $\langle\delta\rangle$, como se muestra a continuación:

$$P_k = e^{-\langle\delta\rangle} \frac{\langle\delta\rangle^k}{k!} \tag{2.5}$$

La Ecuación 2.5, representa solo una aproximación a la Ecuación 2.4, la cual, es válida en el caso que $\langle\delta\rangle \ll N$. La ventaja de la distribución Poisson, es que ella solo depende de un parámetro $\langle\delta\rangle$, lo cual, la hace más simple que la distribución Binomial.

Considere una red aleatoria con grado medio $\langle\delta\rangle$. Un vértice en esta red tiene, en promedio:

- $\langle\delta\rangle$ vértices a distancia 1.
- $\langle\delta\rangle^2$ vértices a distancia 2.
- $\langle\delta\rangle^3$ vértices a distancia 3.
- $\langle\delta\rangle^d$ vértices a distancia d.

Se puede calcular el número esperado de vértices a distancia d, denotado $N(d)$, desde un vértice inicial arbitrario, como sigue:

Sea:

$$N(d) = 1 + \langle\delta\rangle + \langle\delta\rangle^2 + \langle\delta\rangle^3 + ... + \langle\delta\rangle^d \tag{2.6}$$

Multiplicando ambos lados de la igualdad en la Ecuación 2.6 por $\langle\delta\rangle$, se obtiene:

$$\langle\delta\rangle N(d) = \langle\delta\rangle + \langle\delta\rangle^2 + \langle\delta\rangle^3 + \langle\delta\rangle^4 + ... + \langle\delta\rangle^{d+1} \tag{2.7}$$

Restando 2.6 de 2.7, se obtiene:

$$\langle\delta\rangle N(d) - N(d) = \langle\delta\rangle^{d+1} - 1 \tag{2.8}$$

Despejando a $N(d)$:

$$N(d) = \frac{\langle\delta\rangle^{d+1} - 1}{\langle\delta\rangle - 1} \tag{2.9}$$

Note que $N(d)$, no debe exceder el número total de vértices (N) en la red. Se puede identificar, entonces, el diámetro de la red (D_G) y definir:

$$N(D_G) \approx N \tag{2.10}$$

Luego, se puede realizar lo siguiente:

Sea $N(d) \approx N(D_G)$, por transitividad, se puede afirmar que $\frac{\langle\delta\rangle^{D_G+1}-1}{\langle\delta\rangle-1} \approx N$, luego, asumiendo que $\langle\delta\rangle \gg 1$, se puede prescindir del término -1 del numerador y del denominador de la expresión anterior y obtener: $\frac{\langle\delta\rangle^{D_G+1}}{\langle\delta\rangle} \approx N$, después, realizando las simplificaciones correspondientes, se obtiene:

$$\langle\delta\rangle^{D_G} \approx N \tag{2.11}$$

Aplicando logaritmos naturales en ambos lados, y despejando a D_G, el diámetro de una red aleatoria, se define como sigue:

$$D_G \approx \frac{\ln N}{\ln\langle\delta\rangle} \tag{2.12}$$

La Ecuación 2.12, representa la formulación matemática del fenómeno del mundo pequeño (del cual, se hablará más adelante), no obstante, para la mayoría de las redes, ofrece una mejor aproximación a la longitud de trayectoria promedio (LTP_G) que al diámetro. Esto se debe a que, a menudo, el diámetro es dominado por unos cuantos caminos extremos, mientras

que en la longitud de trayectoria promedio, se promedian todos los pares de vértices, por lo tanto, el fenómeno del mundo pequeño, es definido como sigue:

$$LTP_G \approx \frac{\ln N}{\ln\langle\delta\rangle} \tag{2.13}$$

Por otro lado, para calcular el coeficiente de agrupamiento de un vértice v en una red aleatoria, se necesita estimar el número esperado de aristas L_v entre los $\delta(v)$ vecinos de v. La probabilidad de que dos vértices adyacentes a v estén conectados entre sí, es p, y además, como existen $\frac{\delta(v)(\delta(v)-1)}{2}$ posibles aristas entre los $\delta(v)$ vértices adyacentes a v, el número esperado de L_v, denotado $\langle L_v\rangle$, es definido como sigue:

$$\langle L_v\rangle = p\frac{\delta(v)(\delta(v)-1)}{2} \tag{2.14}$$

Sea $T(v) = \langle L_v\rangle$. Sustituyendo la Ecuación 2.14 en la Ecuación A.6, se puede obtener una expresión para el coeficiente de agrupamiento de una red aleatoria G, como sigue:

$$C_G = \frac{2\frac{(p)\delta(v)(\delta(v)-1)}{2}}{\delta(v)(\delta(v)-1)} = p \tag{2.15}$$

Ahora, despejando a p de la Ecuación 2.3 y, por transitividad, se puede afirmar:

$$C_G = \frac{\langle\delta\rangle}{N} \tag{2.16}$$

Note que en la Ecuación 2.16 se omite el término -1 en el denominador, debido a que se asume $N \gg 1$.

2.1.4. El experimento de Milgram

En 1967, el psicólogo social Stanley Milgram, durante su estancia en Harvard, realizó un experimento simple [15], el cual, muestra que a pesar del número tan grande de personas que vivían en ese entonces en los Estados Unidos, y el número relativamente pequeño de conocidos de una persona en particular, dos personas seleccionadas aleatoriamente están muy cercanamente conectadas una a la otra. Milgram, explicó cómo fue llevado a cabo su estudio de la siguiente manera:

La idea general, fue obtener una muestra de hombres y mujeres de todas las clases sociales. A cada una de estas personas, se le daría el nombre y la dirección de la misma persona objetivo, una persona elegida aleatoriamente que vive en algún lugar de los Estados Unidos. A cada uno de los participantes, se le pediría trasladar un mensaje hacia la persona objetivo, usando solo una cadena de amigos y conocidos. A cada persona se le pediría transmitir el mensaje al amigo o conocido que piense sería más probable de conocer a la persona objetivo. Los mensajes podían dirigirse solo a personas con las que tengan una relación personal muy cercana.

En un primer estudio, los mensajes fueron dados a personas residentes en Wichita, Kansas, y debían llegar a la esposa de un estudiante de teología residente en Cambridge, Massachusetts, y en un segundo estudio, los mensajes fueron dados a personas residentes en Omaha, Nebraska, y debían llegar a un corredor de bolsa que trabaja en Boston y vive en Sharon, Massachusetts. De las 160 cadenas que empezaron en Nebraska, 44 fueron completadas. El número de intermediarios necesarios para alcanzar a la persona objetivo en el estudio de Nebraska, está distribuido como se muestra a continuación:

$$(2,2)(3,4)(4,9)(5,8)(6,11)(7,5)(8,1)(9,2)(10,2)$$

El primer número de las parejas ordenadas, representa la longitud de la cadena (número de intermediarios), y el segundo, el número de cadenas completadas. La mediana de la

distribución es igual a 5 y su media aritmética es 5.43. Sin duda, estos resultados no son muy precisos, ya que las estadísticas son demasiado pobres para hacer afirmaciones serias. Sin embargo, el hecho de que dos personas aleatoriamente seleccionadas estén conectadas por solo una cadena corta de conocidos (a lo más 6), es referido como el *fenómeno del mundo pequeño*, y ha sido verificado para muchas redes sociales diferentes. El fenómeno del mundo pequeño, es también conocido como *los seis grados de separación*.

2.1.5. Modelos de redes complejas

Las interacciones entre elementos de diversos sistemas de la vida real no son del todo aleatorias, en este sentido, surge la necesidad de crear nuevos modelos de redes complejas que vayan más allá de la aleatoriedad. Dos modelos importantes, son las redes de mundo pequeño y las redes libre de escala.

2.1.5.1. Redes de mundo pequeño

Duncan Watts y Steven Strogatz, tuvieron un interés muy grande por el fenómeno de la sincronización en conjuntos masivos de osciladores. Este fenómeno ocurre cuando un conjunto de osciladores ajustan su frecuencia y fase, conduciendo a modos de movimiento colectivos coordinados. Su conjetura fue que existía una red de comunicación subyacente tal que permitía el intercambio de señales de una manera rápida, provocando la emergencia del fenómeno de la sincronización. Entonces, se preguntaron:

1. ¿Existirá algún sistema de comunicaciones en los osciladores, o alguna red, que les permita comunicarse y sincronizarse?

2. Asumiendo la existencia de tal red, ¿cuáles son sus propiedades estructurales?

Watts y Strogatz, realizaron diversos estudios, y entonces, decidieron proponer una extensión del modelo de red aleatoria con el objetivo de explicar dos observaciones que se habían descrito en las redes reales:

1. El fenómeno del mundo pequeño, el cual caracteriza a las redes reales por tener una longitud de trayectoria promedio dependiente del logaritmo del orden de la red.

2. El coeficiente de agrupamiento promedio de las redes reales es mucho mayor que el esperado en una red aleatoria de orden y tamaño similares.

Watts y Strogatz, propusieron un modelo de reconexión de aristas que interpola entre redes regulares y aleatorias. Dicho modelo, parte de una red regular $G = (V, E)$ con topología anillo, de orden $N = n$, donde cada vértice $v \in V$, está conectado a sus k vecinos más cercanos ($k = 2r, r \in \mathbb{N}$), tal que $N \gg k \gg \ln(N) \gg 1$, donde cada arista de la red es reconectada con una probabilidad p, pasando de la regularidad $p = 0$, hasta el desorden $p = 1$, con el objetivo de explorar la región $0 < p < 1$. Sus redes de interés, tienen muchos vértices con aristas dispersas, pero no tanto como para que la red corra el riesgo de desconectarse.

El Algoritmo 2, muestra los pasos para construir una red con base en el modelo Watts-Strogatz.

El estudio de Watts-Strogatz, requiere $N \gg k \gg \ln(N) \gg 1$, donde $k \gg \ln(N)$ garantiza que una red aleatoria estará conectada. Watts y Strogatz, encontraron lo siguiente:

1. $LTP_G \sim \frac{N}{2k} \gg 1$ y $C_G \sim \frac{3}{4}$ conforme $p \to 0$.

2. $LTP_G \approx LTP_{random} \sim \frac{\ln(N)}{\ln(k)}$ y $C_G \approx C_{random} \sim \frac{k}{N} \ll 1$ conforme $p \to 1$.

Lo que significa que una red regular con $p = 0$, tiene un alto coeficiente de agrupamiento promedio y una longitud de trayectoria promedio que crece de manera lineal con respecto a N. Mientras que una red aleatoria, con $p = 1$, es un mundo pequeño, con coeficiente de agrupamiento promedio bajo y una longitud de trayectoria promedio que crece de manera logarítmica con respecto a N. Lo anterior, hace suponer lo siguiente:

1. C_G alto, está asociado a LTP_G alta.

Algoritmo 2 Construcción de una red con base en el modelo Watts-Strogatz

Requiere:

Una gráfica $G = (V, E)$ con topología anillo, de orden N, donde cada vértice $v \in V$ esté etiquetado con un número $i, i \in \mathbb{N}$ en el sentido de las manecillas del reloj; y esté conectado a sus k vecinos más cercanos ($k = 2r, r \in \mathbb{N}$), tal que $N \gg k \gg \ln(N) \gg 1$.

Un valor p que representa la probabilidad de reconexión.

1: $probabilidad \leftarrow p$
2: $j \leftarrow 1$
3: **Mientras** $j \leq \frac{k}{2}$ **Hacer**
4: **Para** $i = 1$ hasta N **Hacer**
5: Selecciona $e = (i, u)$, tal que u sea el j-ésimo vértice más cercano a i, en sentido de las manecillas del reloj
6: Genera un número aleatorio, denominado $aleat$, con distribución $\sim U[0, 1]$
7: **Si** $aleat \leq probabilidad$ **Entonces**
8: Selecciona un vértice b de G aleatorio con distribución $\sim U[1, N]$
9: **Si** $b \in N(i)$ **Entonces**
10: Mantener e intacta
11: **Sino**
12: Reconecta e tal que $e = (i, b)$
13: **Termina Si**
14: **Sino**
15: Mantener e intacta
16: **Termina Si**
17: **Termina Para**
18: $j = j + 1$
19: **Termina Mientras**

2. C_G bajo, está asociado a LTP_G bajo.

No obstante, existe un intervalo de p sobre el cual LTP_G es casi tan baja como LTP_{random} y, también, $C_G \gg C_{random}$. Estas redes son mundos pequeños resultantes de la caída inmediata de LTP_G causada por el surgimiento de unas cuantas aristas de largo alcance. Tales "atajos", conectan vértices que, de otro modo, estarían más alejados que LTP_{random}. Esta idea, revela el papel clave de los atajos en las redes, sugiriendo que el fenómeno del mundo pequeño, podría ser común en redes dispersas con muchos vértices, ya que, incluso, una pequeña cantidad de atajos serían suficientes para lograrlo. El fenómeno del mundo pequeño, para Watts y Strogatz, consiste en redes que tienen una longitud de trayectoria promedio baja, aunado a coeficientes de agrupamiento promedio altos.

2.1.5.2. Redes libres de escala

Las redes aleatorias, tradicionalmente se usaron para describir redes con topologías complejas, pero a falta de datos sobre redes grandes, las predicciones de las redes aleatorias fueron escasamente probadas en el mundo real. Albert Lázló Barabási y Réka Albert, exploraron muchas bases de datos que describen la topología de grandes redes, por ejemplo, la World Wide Web, y demostraron que, independientemente del sistema y de la identidad de sus componentes, la probabilidad $P(k)$ de que un vértice en la red interactúe con otros k vértices decrece como una *ley de potencias*. Las leyes de potencias, son distribuciones de probabilidad de la forma:

$$P(k) \sim k^{-\gamma} \tag{2.17}$$

Las distribuciones de probabilidad de ley de potencias, son teóricamente interesantes debido a que tienen "colas pesadas", lo cual quiere decir que las colas derechas de las distribuciones contienen mucha probabilidad. Esta cola, puede ser tan extrema que la desviación estándar de la distribución puede ser indefinida (para $\gamma < 3$), o incluso, la media (para

$\gamma < 2$). Estas cualidades, forman un sistema libre de escala, en el cual, todos los valores son esperados de ocurrir sin un tamaño característico o escala.

Aplicando logaritmos, y algunas de sus propiedades, en la Ecuación 2.17, se obtiene:

$$\log P(k) \sim -\gamma \log k \tag{2.18}$$

Lo cual quiere decir que se espera que $\log P(k)$, dependa linealmente de $\log k$, siendo γ, la pendiente de esta línea. Bajo este contexto, una red que sigue una distribución de grados de ley de potencias, es llamada *red libre de escala.*

Los grados de los vértices de una red, son números $k \in \mathbb{N} \cup \{0\}$. La distribución de probabilidad de ley de potencias, en su versión discreta, nos dice la probabilidad de que un vértice en la red tenga exactamente k aristas, y se define como sigue:

$$P(k) = Ck^{-\gamma} \tag{2.19}$$

La constante C, puede ser determinada mediante la condición de normalización mostrada en la Ecuación A.3. Al sustituir la Ecuación 2.19 en la Ecuación A.3, se obtiene:

$$C \sum_{k=1}^{\infty} k^{-\gamma} = 1 \tag{2.20}$$

Despejando a C de la Ecuación 2.20, se obtiene:

$$C = \frac{1}{\sum_{k=1}^{\infty} k^{-\gamma}} \tag{2.21}$$

Sustituyendo la Ecuación 2.21 en la Ecuación 2.19 y asumiendo $k > 0$, la distribución discreta de ley de potencias tiene la forma:

$$P(k) = \frac{k^{-\gamma}}{\sum_{k=1}^{\infty} k^{-\gamma}} \tag{2.22}$$

Note que la Ecuación 2.22 diverge en $k = 0$. Si se necesitara, se puede especificar, separadamente, $P(0)$. En ese caso, el cálculo de C, en la Ecuación 2.21, necesitaría incorporar $P(0)$.

Una característica común de los modelos de redes propuestos por Erdös-Rényi y Watts-Strogatz, es que los vértices con un grado alto están prácticamente ausentes. En contraste, la cola de ley de potencias que caracteriza a $P(k)$, indica que vértices altamente conectados tienen grandes oportunidades de ocurrir, estos vértices son llamados *concentradores*. Los concentradores, representan la diferencia más notable entre una red aleatoria y una red libre de escala. Un par de características importantes de las redes con distribución de grados de ley de potencias, se definen a continuación:

1. Tienen muchos vértices con grados bajos.

2. Tienen pocos vértices con grados altos (concentradores).

Por otro lado, existen dos aspectos genéricos de las redes reales que no son incorporados en los modelos ER y WS. Primero, ambos modelos asumen que se inicia con un número fijo de vértices que son aleatoriamente conectados (modelo ER), o reconectados (modelo WS), sin eliminar o aumentar vértices a las redes. En contraste, muchas redes reales se forman por la adición continua de nuevos vértices a la red. Segundo, los modelos de redes aleatorias, asumen que la probabilidad de que dos vértices sean adyacentes es aleatoria y uniforme, en cambio, muchas redes reales, exhiben conectividad preferencial. Por ejemplo, en la World Wide Web, una página recién creada, será más probable de incluir enlaces a páginas populares que ya tienen alta conectividad. Este ejemplo indica que la probabilidad con la cual un vértice nuevo se conecta a vértices existentes no sigue una distribución uniforme.

El Algoritmo 3, muestra los pasos necesarios para construir una red con base en el modelo propuesto por Albert Lázló Barabási y Réka Albert.

Algoritmo 3 Construcción de una red libre de escala con base en el modelo Barabási-Albert

Requiere:

Una gráfica $G = (V, E)$ con N vértices, donde sus aristas estén colocadas arbitrariamente, siempre que cada vértice tenga, al menos, una arista incidente.

Un valor t, que representa el número de vértices máximo deseado en la red.

1: **Para** $i = 1$ hasta t **Hacer**
2: Genera un nuevo vértice v, con m aristas, tal que $m \leq N$
3: **Para** $j = 1$ hasta m **Hacer**
4: Selecciona un vértice $u \in V$, con base en una probabilidad $\prod(u) = \frac{\delta(u)}{\sum_{s,s\in V}\delta(s)}$
5: Establece $e_j = (v, u)$
6: **Termina Para**
7: **Termina Para**

La *conectividad preferencial*, es un mecanismo probabilístico, ya que un nuevo vértice es libre de conectarse con cualquier vértice de la red, tanto si es un concentrador o si tiene una sola arista incidente. La ecuación mostrada en la línea 4 del Algoritmo 3, refleja, sin embargo, que si un nuevo vértice puede elegir conectarse a un vértice de grado dos y uno de grado cuatro, es el doble de probable que se conecte al vértice de grado cuatro. En resumen, el modelo Barabási-Albert, indica que dos mecanismos simples, el crecimiento de la red y la conectividad preferencial, son responsables de la emergencia de redes libres de escala.

El diámetro de una red G, construida con base en el modelo Barabási-Albert, para $m > 1$ y N grande, se define como sigue:

$$D_G \sim \frac{\ln N}{\ln(\ln(N))} \tag{2.23}$$

Por lo tanto, el diámetro crece tan lento como $\ln N$, provocando que las distancias en las

redes Barabási-Albert sean tan pequeñas como las distancias observadas en una red aleatoria de orden similar.

Por otro lado, el coeficiente de agrupamiento promedio de una red Barabási-Albert, se define como sigue:

$$C_G \sim \frac{(\ln N)^2}{N} \tag{2.24}$$

La predicción mostrada en la Ecuación 2.24, es bastante diferente de la dependencia $\frac{1}{N}$, obtenida en el modelo de red aleatoria. La diferencia viene en el término $(\ln N)^2$, que incrementa el coeficiente de agrupamiento para N grande. Consecuentemente, las redes Barabási-Albert están, localmente, más agrupadas que una red aleatoria.

2.2. Robustez en redes complejas

2.2.1. Introducción

Con el propósito de estudiar el impacto en el rendimiento de cualquier servicio provisto por redes de telecomunicaciones, los investigadores, se han enfocado en evaluar la robustez de las redes en casos de múltiples escenarios de degradación. La definición tradicional de *robustez*, la cual está basada en la teoría de gráficas, está, principalmente, centrada en la conectividad de las gráficas. En esta investigación, se asumen un par de definiciones más contemporáneas que a la letra dicen: *La robustez, es la capacidad de una red de mantener su rendimiento total bajo la eliminación de vértices y aristas.* Esta definición, toma en consideración los procesos dinámicos que se ejecutan sobre una red, mientras que la definición basada en teoría de gráficas, no [37]. En el contexto de las redes de telecomunicaciones, la robustez es definida como *la capacidad de una red de operar y mantener un nivel aceptable de servicio en presencia de condiciones adversas.* En este concepto, subyace el requerimiento de que, cuando los problemas surjan, el servicio prestado por la red debe permanecer accesible, incluso si ello

significue operar en un modo degradado, y que la red deba iniciar acciones para salir del modo degradado en forma rápida y automática [40].

El objetivo fundamental de estudiar el comportamiento de las redes, es obtener conocimiento de los sistemas complejos que representan. Un aspecto importante de esto, es entender el efecto de los procesos de degradación de los componentes individuales en el rendimiento de todo el sistema basado en red. La motivación detallada para estudiar este efecto, depende del sistema en red concreto bajo consideración. Por ejemplo, es claramente importante conocer cómo la falla de dispositivos de encaminamiento individuales en la Internet afecta la función general de la red. Similarmente, si la red en cuestión es una red de contactos, en la cual, puede propagarse una enfermedad, resulta fundamental entender como la eliminación efectiva de vértices de la red (por ejemplo, a través de vacunación) afecta la propagación de la enfermedad. Es claro, de ejemplos como estos, que identificar aquellos vértices que afectan más crucialmente la función de un sistema en red es, a menudo, de gran importancia. En algunos casos, como el de la Internet, se desea identificar estos vértices para que los elementos más cruciales del sistema puedan ser protegidos de fallas o ataques, y el funcionamiento de todo el sistema pueda mantenerse de manera efectiva. En otros casos, el objetivo es identificar los vértices clave en una red para que todo el sistema pueda ser efectivamente degradado mediante su remoción. Situaciones en las que este último objetivo es pertinente, incluyen redes de contactos para enfermedades infecciosas y redes criminales y terroristas. Entender la robustez de sistemas en red contra la falla o pérdida de sus componentes está, cercanamente, relacionado al estudio de la percolación en redes [38].

2.2.2. Teoría de la percolación

El proceso que resulta de tomar una red y remover alguna fracción de sus vértices (junto con las aristas que los conectan), se denomina *percolación*. Por ejemplo, la falla de los dispositivos de encaminamiento en la Internet, o la vacunación de individuos para prevenir la propagación de una enfermedad, pueden ser representadas, formalmente, por la remoción

de los vértices correspondientes de las redes. Aunque un dispositivo de encaminamiento que ha fallado o un individuo que ha sido vacunado sigan presentes en la red, desde un punto de vista funcional, pueden haber sido eliminados [38].

La remoción de un solo vértice, tiene un impacto limitado en la conexidad de la red. La remoción de muchos vértices, sin embargo, puede romper la red en demasiados componentes aislados. Obviamente, mientras más vértices sean removidos, mayores son las oportunidades de dañar la red, llevando a la siguiente cuestión: ¿cuántos vértices se deben de eliminar para fragmentar una red en componentes aislados?, por ejemplo, ¿cuántos dispositivos de encaminamiento deben salir de operación para que la Internet se convierta en conjuntos de computadoras que sean incapaces de comunicarse entre sí? [4].

2.2.3. Métricas de la robustez en redes

En esta sección, se presentan tres métricas que han sido propuestas explícitamente para llevar a cabo la tarea de evaluar la robustez de una red.

2.2.3.1. Orden relativo del componente más grande

Uno de los aspectos clave de estudiar la percolación en una red G, es entender como cambia el orden del componente más grande conforme sus vértices son removidos. Esto es, claramente relevante, para el tema de la robustez de la red, ya que si el orden del componente más grande es suficientemente pequeño, relativo al orden original de la red, es razonable pensar que cualquier pareja arbitraria de vértices se encuentra desconectada, con mucha probabilidad. Para una red inicial G de orden n, sea G_ρ la red que resulta de remover una fracción ρ de vértices de acuerdo a algún procedimiento específico. Sea G_ρ^λ el componente más grande de G_ρ. La cantidad clave que se desea estudiar, es el orden $\sigma(\rho)$ de G_ρ^λ relativo al orden inicial de la red n; esto es, $\sigma(\rho) = \frac{|G_\rho^\lambda|}{n}$, donde $\mid G_\rho^\lambda \mid$ denota el número de vértices en G_ρ^λ. Computar σ como una función de ρ, permite cuantificar cómo la robustez de una red, depende de la fracción de vértices que son removidos [38].

El orden relativo del componente más grande, entrega una idea aproximada de la robustez de la red, pero en realidad, no mide hasta qué punto la red todavía es capaz de mantener la comunicación entre pares de vértices. Una mejor medida es el $A2TR(p)$, la fiabilidad promedio entre dos terminales, la cual, se define a continuación.

2.2.3.2. Fiabilidad promedio entre dos terminales ($A2TR(p)$)

La fiabilidad promedio entre dos terminales $A2TR(p)$, representa la probabilidad de que un par de vértices, aleatoriamente seleccionados con base en una función de distribución de probabilidad uniforme, estén conectados cuando p vértices son removidos de la red. Si la red está totalmente conectada, su valor de $A2TR(p)$ es 1. De otra manera, cuando p vértices son removidos, su valor de $A2TR(p)$ es calculado como la suma del número de parejas de vértices en cada componente fuertemente conectado SCC, dividido por el número total de parejas de vértices en la red residual, tal y como se muestra en la Ecuación 2.25.

$$A2TR(p) = \frac{\sum_{i=1}^{|SCC|} \mid C_i \mid (\mid C_i \mid -1)}{n'(n'-1)} \tag{2.25}$$

Donde:

1. $\mid C_i \mid$ representa el número de vértices del i-ésimo SCC.

2. n' representa el orden de la red residual $n - p$, siendo n el orden de la red.

La relación mostrada en la Ecuación 2.25, indica la fracción de parejas de vértices que están conectados unos con otros. Por lo tanto, mientras más alto sea el valor del $A2TR(p)$ (para un número dado de vértices removidos), mas conectada estará la red [42].

En el momento que $A2TR(p) = 0$, ocurren dos cosas:

1. Las redes han sido totalmente fragmentadas.

2. La comunicación entre los vértices que aún viven en las redes, ya no es posible.

2.2.3.3. Media de la fiabilidad promedio entre dos terminales ($\mu - A2TR$)

La media de la fiabilidad promedio entre dos terminales $\mu - A2TR$, caracteriza que tan difícil es romper una red en componentes cuando se considera un escenario incremental de degradación. Su ecuación, se define como sigue:

$$\mu - A2TR = \frac{\sum_{p=0}^{n-2} A2TR(p)}{n-1} \tag{2.26}$$

Donde:

1. p representa el número de vértices que han sido removidos de la red.

2. n representa el orden de la red.

3. $A2TR(p)$ representa el valor $A2TR(p)$ de la red para p vértices removidos.

$\mu - A2TR$, toma valores en el intervalo $[0, 1]$. Mientras más grande el valor de $\mu - A2TR$, más robusta es la red, en términos de conectividad, y más difícil de segregar en componentes es [42].

Para ejemplificar el uso de las métricas $A2TR(p)$ y $\mu - A2TR$, suponga una red G arbitraria de orden $n = 2500$, la cual, es degradada mediante algún proceso. En el proceso de degradación utilizado, se consideró calcular su $A2TR(p)$ cada $p\,\%$ de vértices removidos. Posteriormente, como resultado, se obtiene un $\mu - A2TR = 0.48$ y un $A2TR(47) = 0.69$, lo cual, indica que, al cabo del $47\,\%$ de los vértices removidos de la red, la probabilidad de establecer comunicación, entre cualesquiera dos vértices que aún viven en ella, es del $69\,\%$. La gráfica en la Figura 2.3, muestra la "historia" de la degradación de G, hasta que sus vértices pierden total comunicación, es decir, hasta que su $A2TR(p) = 0$. El punto de intersección de

las rectas perpendiculares, denotadas en color rojo, representa el umbral $\mu - A2TR$.

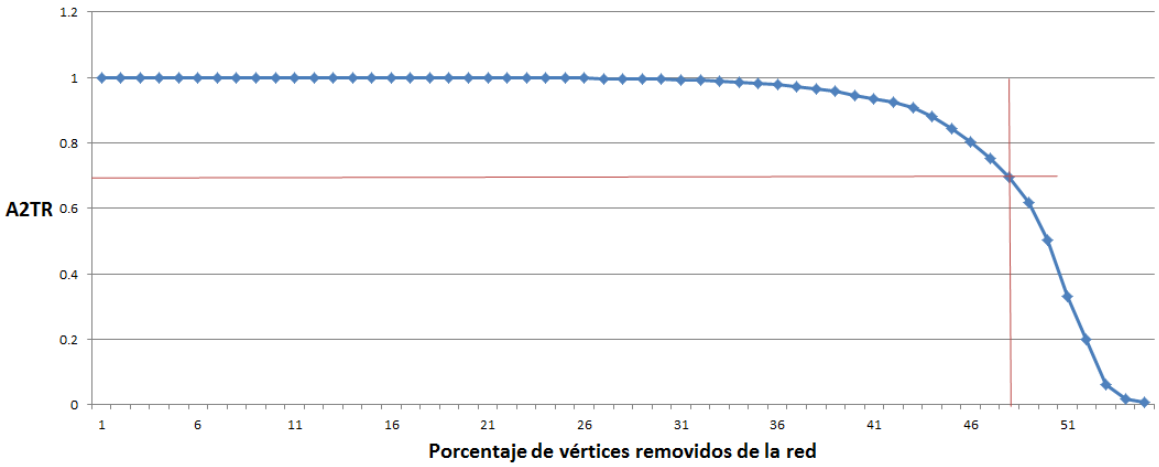


Figura 2.3: "Historia" del proceso de degradación de G

2.2.4. Múltiples escenarios de degradación

Las fallas, pueden afectar la operación normal de los elementos de la red (vértices o aristas). Por lo tanto, los servicios soportados por una red y los usuarios conectados a ella, pueden vivir consecuencias catastróficas. La Figura 2.4, muestra como pueden ser clasificadas las fallas.

De acuerdo a una dimensión temporal, las fallas pueden ser clasificadas como estáticas o dinámicas (propagación). Las *fallas estáticas*, son esencialmente fallas puntuales que afectan a uno o varios elementos en un momento dado. El conjunto de los elementos cuya falla es atribuible a una causa específica no varía con el tiempo, excepto que su cardinalidad puede decrecer temporalmente debido a reparaciones o reemplazos. En otras palabras, la falla no se propaga. Las *fallas dinámicas*, tienen una dimensión temporal, es decir, las fallas progresan en el tiempo, de modo que el conjunto de elementos afectados es diferente de un instante a otro. Pueden ser determinadas como epidémicas o fallas en cascada. Las *fallas epidémicas*, son causadas, principalmente, por virus y gusanos informáticos, aunque implementaciones erróneas de protocolos y otros errores de software, podrían, potencialmen-

te, provocarlas también. Su única característica, es que el agente que provoca la falla en un vértice dado, se replica y se propaga de manera autónoma a través de la red. Las *fallas en cascada*, también llamadas *fallas inducidas*, son aquellas en las cuales dos o más elementos fallan, pero todos ellos fueron llevados a ese estado por la falla de un solo elemento. La sutil diferencia entre fallas epidémicas y en cascada, es que, en las primeras, el agente que inicia la falla, interviene, repetidamente, sobre diferentes elementos, mientras que en las últimas, la combinación de propiedades estructurales de la red (por ejemplo, distribución de grados) y dinámica (por ejemplo, flujo de tráfico), juegan un rol en la inducción de la emergencia de más fallas, incluso en un completo colapso. Otros escenarios, se presentan cuando existe una acción premeditada para causar el mayor daño a la conexidad. Hablamos entonces de un *ataque*. Así, cuando un objeto que causa un ataque conoce y usa información precisa de la estructura topológica de la red, es llamado un *ataque dirigido*. Sin embargo, cuando el atacante tiene poca o nula información, es considerado un *ataque aleatorio*. Los primeros estarían más relacionados a fallas intencionales, mientras que los últimos, serían fallas no intencionadas. En las *fallas aleatorias*, las fallas de vértices o aristas ocurren aleatoriamente, por ejemplo, un corte de fibra ocasionado por un desastre natural. Mientras que en las *fallas dirigidas*, los elementos de la red son atacados (removidos) con el propósito de maximizar el impacto del ataque sobre la red, por ejemplo, en las redes troncales de telecomunicaciones, los dispositivos de encaminamiento más vulnerables pueden ser identificados mediante el número de rutas más cortas que pasan a través de un dispositivo de encaminamiento dado, o por el número de enlaces físicos de un dispositivo de encaminamiento a otros. En las *fallas dirigidas*, existen dos esquemas distintos para seleccionar los elementos que serán removidos. En un *ataque simultáneo dirigido*, la métrica de centralidad es calculada para todos los elementos (vértices o aristas) en la red y luego una fracción específica de los elementos es removida en orden de la medida de centralidad, del más alto al más bajo. En un *ataque secuencial dirigido*, la medida de centralidad es calculada para todos los elementos en la red inicial y el elemento con el mayor valor de centralidad es removido. Después, las medidas de centralidad de todos los elementos en la red resultante son recalculadas y, una vez más, se elimina el elemento

mejor clasificado. Este proceso de recalcular las medidas de centralidad y remover el elemento mejor clasificado se continúa hasta que la fracción deseada de elementos han sido removidos [41].

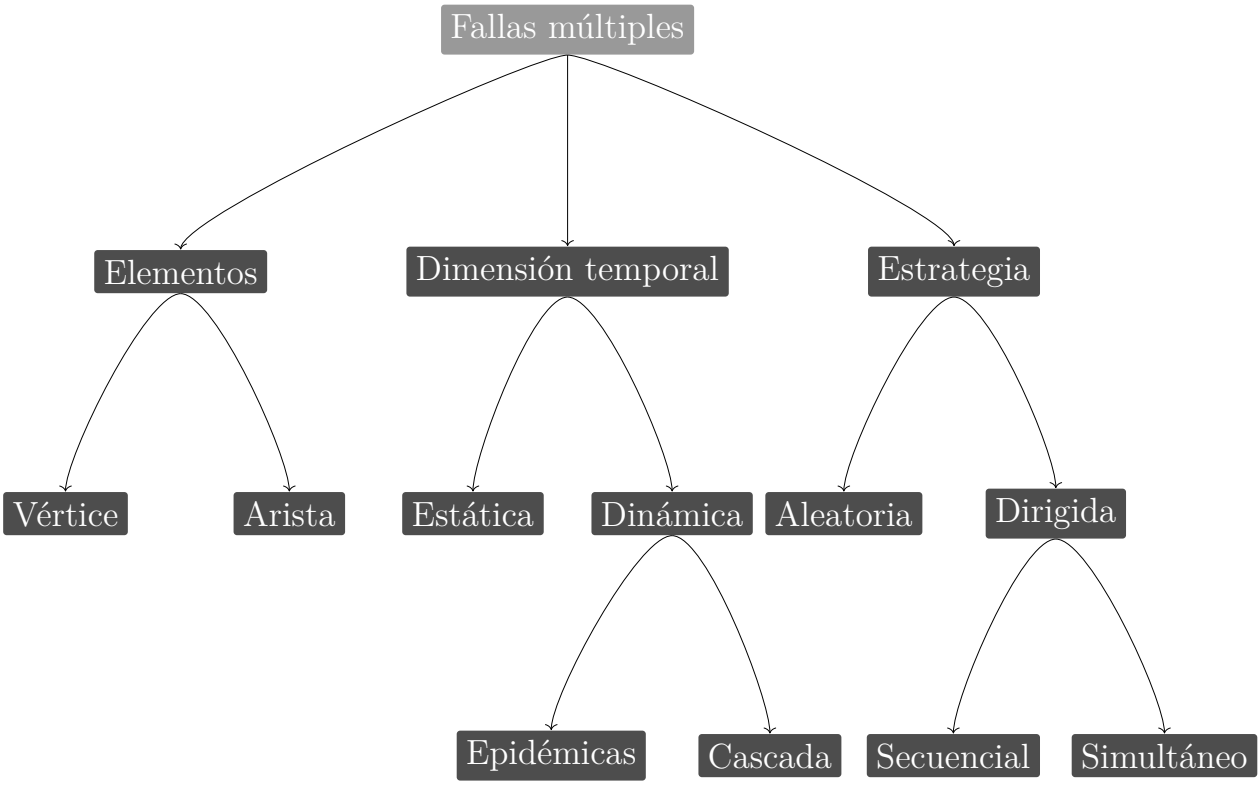


Figura 2.4: Clasificación de las fallas

2.3. Ciencias de la complejidad

2.3.1. Complejidad

El término *complejidad*, deriva etimológicamente del latín *plexus*, que significa "entrelazado". Intuitivamente, esto implica que algo complejo está compuesto por elementos que son difíciles de separar. Esta dificultad, surge de las interacciones relevantes que se llevan a cabo entre sus componentes. No existe una definición general de complejidad, debido a que el concepto adquiere diferentes significados en diferentes contextos. A pesar de esto, se puede decir que un sistema es *complejo*, si consiste de muchos elementos que interactúan entre sí, lo que

provoca que el comportamiento del sistema sea difícil de deducir a partir del conocimiento de cada uno de sus elementos. A pesar de que no existe una definición general de la medida de la complejidad, una noción relativa se muestra a continuación [27]:

La complejidad de un sistema C_{sys}, escala con el número de sus elementos $\#E$, el número de interacciones entre ellos $\#I$, las complejidades de sus elementos C_e y las complejidades de sus interacciones C_i:

$$C_{sys} \sim \#E\#I \sum_{j=0}^{\#E} C_{e_j} \sum_{k=0}^{\#I} C_{i_k} \tag{2.27}$$

La complejidad de una interacción C_i, puede ser medida como el número de diferentes interacciones posibles que dos elementos pueden tener.

Una medida bien aceptada de la complejidad, es la cantidad de información requerida para describir un fenómeno a una escala dada. Desde este punto de vista, mientras más complejo sea un fenómeno, requerirá más información para ser descrito. Es importante notar que la escala es relevante para determinar la cantidad de información, por ejemplo, un gas requiere mucha más información para ser descrito en una escala atómica (con todos los detalles de las posiciones y momentos de las moléculas), que en una escala humana (donde todos los detalles moleculares son promediados para producir temperatura, presión, volumen, etc.) [26].

2.3.2. Definición y características

De acuerdo con [25], se define a las *ciencias de la complejidad* y a diversas características presentes en los sistemas complejos, como sigue:

Las ciencias de la complejidad, también llamadas ciencias de sistemas complejos, estudian la forma en que grandes conjuntos de componentes, interactuando localmente entre sí, a

pequeña escala, pueden espontáneamente auto-organizarse y presentar estructuras globales y comportamientos no triviales a mayores escalas, sin intervención externa, autoridad central o líderes que determinen el comportamiento colectivo. Las propiedades del todo pueden no ser entendidas o predichas a partir del conocimiento total de cada una de sus partes. Las colecciones de elementos que presentan estas propiedades, son sistemas complejos, y requieren de nuevos marcos matemáticos y métodos científicos para ser estudiados.

2.3.2.1. Interacciones

Los sistemas complejos, suelen caracterizarse por tener muchos componentes que interactúan de formas múltiples entre sí y, potencialmente, con su entorno. Estas partes, forman redes de interacciones, a veces con unos pocos componentes involucrados en muchas interacciones. Las interacciones pueden generar información nueva que complica el estudio individual de las partes o la predicción correcta de su futuro. Adicionalmente, los componentes de un sistema, pueden también ser nuevos sistemas, es decir, sistemas de sistemas interdependientes entre sí. El mayor reto de las ciencias de la complejidad, no es solo apreciar las partes y conexiones, sino también entender como estas interacciones dan lugar al todo. Ejemplos, se enuncian a continuación:

1. Billones de neuronas interactuando en el cerebro humano.

2. Computadoras comunicándose a través de la Internet.

3. Humanos en relaciones multifacéticas.

2.3.2.2. Emergencia

En sistemas simples, las propiedades del conjunto pueden entenderse o predecirse a partir de la suma o agregación de sus componentes. En otras palabras, las propiedades macroscópicas de un sistema simple, pueden deducirse de las propiedades microscópicas de sus partes. Sin embargo, en sistemas complejos, las propiedades del conjunto, a menudo, no pue-

den entenderse, ni predecirse, a partir del conocimiento de sus componentes, debido a un fenómeno conocido como *emergencia*. Este fenómeno, involucra diversos mecanismos que, a través de la interacción entre los componentes de un sistema, generan información nueva y exhiben tanto estructuras como comportamientos no triviales a escalas más grandes. Este hecho, generalmente, se resume con la frase "el todo es más que la suma de sus partes". Ejemplos, se enuncian a continuación:

1. Una cantidad masiva de moléculas de aire y vapor formando un tornado.

2. Múltiples células formando un organismo.

3. Billones de neuronas en un cerebro produciendo conciencia e inteligencia.

2.3.2.3. Dinámica

Los sistemas, pueden ser analizados en términos de cómo cambian sus estados en el tiempo. Un estado, se describe como un conjunto de variables que caracterizan a dicho sistema de la mejor manera. A medida que el sistema cambia sus variables, lo hacen, habitualmente, respondiendo a su entorno. Este cambio, se denomina lineal si es directamente proporcional al tiempo, al estado actual del sistema, o a cambios en el entorno; o no lineal, si no es proporcional a ellos. Los sistemas complejos, son típicamente no lineales y cambian a diferentes velocidades, dependiendo de sus estados y entorno. También, pueden tener estados estables en los que pueden permanecer aún si son perturbados, o inestables en los que los sistemas pueden ser alterados por una pequeña perturbación. En algunos casos, pequeños cambios en el entorno pueden cambiar completamente el comportamiento del sistema, un fenómeno conocido como *bifurcaciones, transiciones de fase o puntos de quiebre*. Algunos sistemas son *caóticos*, es decir, extremadamente sensibles a pequeñas perturbaciones, e impredecibles a largo plazo. Un sistema complejo, también puede depender de su trayectoria, es decir, su estado futuro no solo depende de su estado actual, sino también de su pasado. Ejemplos, se enuncian a continuación:

1. El clima, cambiando constantemente de manera caótica.

2. La volatilidad de los mercados financieros.

2.3.2.4. Auto-Organización

Las interacciones entre los componentes de un sistema, pueden producir un patrón o comportamiento globales. Esto puede describirse como *auto-organización*, ya que no existe un control central o externo. Más bien, el "control" de un sistema auto-organizante, está distribuido entre sus componentes y se integra a través de sus interacciones. La auto-organización, puede producir estructuras físicas-funcionales, como patrones cristalinos de materiales y morfologías de organismos vivos o bien, comportamientos dinámicos-informacionales, como los de cardúmenes de peces y patrones eléctricos propagándose en músculos de animales. Al incrementarse la organización del sistema mediante este proceso, nuevos patrones de interacción pueden emerger en el tiempo, potencialmente llevando a la producción de mayor complejidad. En algunos casos, los sistemas complejos pueden auto-organizarse hacia un estado "crítico" que solo podría existir manteniendo un sutil balance entre aleatoriedad y regularidad. Los patrones que emergen en tales estados críticos auto-organizados, comúnmente muestran diversas propiedades peculiares, tales como auto-similitud y distribuciones de leyes de potencias de propiedades de patrones. Ejemplos, se enuncian a continuación:

1. Una célula cigoto dividiéndose y, eventualmente, auto-organizándose en la forma compleja de un organismo.

2. Ciudades, creciendo al atraer más habitantes y dinero.

3. Una población grande de aves, mostrando patrones complejos de parvada.

2.3.2.5. Adaptación

Los sistemas complejos, con frecuencia, se encuentran activos y responden al ambiente donde se encuentran, en vez de simplemente tratar de alcanzar un estado asintótico estable,

esta es la diferencia entre una esfera que rueda hasta el fondo de una pendiente y se detiene, y un pájaro que, en pleno vuelo, se adapta a las corrientes de viento. Esta adaptación puede ocurrir en múltiples escalas: cognitiva, a través del desarrollo psicológico; social, al compartir información con nuestro entorno social; y aún también evolutiva, mediante la mutación genética y la selección natural. Cuando los componentes de estos sistemas se descomponen o son eliminados, con frecuencia, los sistemas son capaces de adaptarse y recuperar su grado de funcionamiento previo y, en ocasiones, mejorarse a sí mismos. Esto se logra debido a la robustez, la habilidad de resistir a las perturbaciones; la resilencia, la habilidad de recuperar el estado original luego de una perturbación prolongada; o la adaptación, la habilidad que tiene el sistema para auto-recuperarse a fin de mantenerse funcional y sobrevivir. Los sistemas complejos que poseen estas propiedades son conocidos como *sistemas complejos adaptativos*. Ejemplos, se enuncian a continuación:

1. El sistema inmunológico continuamente aprendiendo como reaccionar ante nuevos patógenos.

2. Una colonia de termitas que reparan su nido luego de sufrir daños.

3. La vida en la tierra que ha sobrevivido a numerosas catástrofes a lo largo de millones de años de historia.

2.3.2.6. Interdisciplinariedad

Los sistemas complejos, aparecen en todos los ámbitos científicos y profesionales, incluyendo Física, Biología, Ecología, Ciencias Sociales, Finanzas, Administración, Política, Psicología, Antropología, Medicina, Ingeniería, Tecnologías de la Información, entre otras. Muchas de las últimas tecnologías, desde las redes sociales y tecnologías móviles hasta vehículos autónomos, producen sistemas complejos con propiedades emergentes que resultan esenciales para entender y predecir el bienestar de la sociedad. Un concepto clave de las ciencias de la complejidad, es la *universalidad*, la idea de que muchos sistemas de diversos ámbitos pre-

sentan fenómenos con características en común que pueden ser descritas usando los mismos modelos científicos. Las ciencias de la complejidad, pueden proveer un enfoque comprensivo y multidisciplinario que complementa alcances científicos tradicionales que se concentran en temas específicos en cada ámbito. Ejemplos, se enuncian a continuación:

1. Propiedades en común de varios sistemas de procesamiento de información (sistemas nerviosos, la Internet, infraestructuras de comunicaciones).

2. Patrones universales en varios en varios procesos de difusión (epidemias, modas, incendios forestales).

2.3.2.7. Métodos

Los sistemas complejos, involucran muchas variables y configuraciones que no pueden ser exploradas con la intuición mediante cálculos simples usando papel y lápiz. En vez de ello, es necesario, casi siempre, incorporar la modelación matemática y computacional, combinando una aproximación analítica con simulaciones, para poder entender cómo este tipo de sistemas se estructuran y cambian en el tiempo. Con la ayuda de computadoras, se puede probar si un conjunto específico de reglas hipotéticas genera, efectivamente, los comportamientos observados en la naturaleza, y luego, el conocimiento de esas reglas puede usarse para generar predicciones de distintos escenarios. Las computadoras, también pueden ser usadas para analizar grandes volúmenes de datos que típicamente producen los sistemas complejos, de tal manera que se pueden revelar y visualizar los patrones inherentes del sistema que suelen estar ocultos al ojo humano. Estos métodos computacionales, pueden conducir a descubrimientos que ayuden a profundizar nuestra comprensión y apreciación de la naturaleza. Ejemplos, se enuncian a continuación:

1. Modelado basado en agentes de parvadas de aves.

2. Modelos matemáticos y computacionales del cerebro.

3. Predicción del clima con modelos computacionales.

4. Modelos computacionales de la dinámica de peatones.

2.4. Simulación de eventos discretos

2.4.1. Conceptos previos

De acuerdo con [20], a continuación se detalla que es la simulación y diversos conceptos inherentes a ella:

2.4.1.1. Sistema

Un *sistema*, es cualquier cosa compuesta por partes o elementos que se relacionan e interactúan entre sí. Sus elementos, cumplen una función, ocupan un lugar y se integran en un orden, mientras que su interacción, engendra cualidades que no poseen por sí solos. La modificación de un elemento, trae cambios en otros y la transformación del sistema. En el contexto de simulación, se define "sistema", como un conjunto de entidades encapsuladas en un área definida, física o virtual, y que siguen reglas de operación para poder responder a estímulos de entidades externas que cruzan los límites hacia el interior de esa área con el objetivo de obtener un servicio o resultado. La palabra "entidad", se refiere a un objeto o persona, mientras que las "reglas de operación", son procesos o ecuaciones.

2.4.1.2. Modelo

Un *modelo*, es una representación de un sistema real (existente o en diseño). además, es un mecanismo de abstracción que, necesariamente, deja fuera algunos aspectos del sistema. Un mismo sistema puede tener más de un modelo que lo represente. El modelo, sirve para sustituir al sistema en un ambiente de estudio y análisis, describiendo su comportamiento y usando teorías para predecir su comportamiento futuro. En particular, en esta investigación, se hace uso de *modelos computacionales*, en los cuales, el modelo se representa, en su mayoría,

a través del uso de programas informáticos, es decir, el uso de ecuaciones matemáticas es parte de una metodología computacional. El modelo es una secuencia de subrutinas a ejecutar en la computadora, escritas en algún lenguaje de programación, pero puede utilizar relaciones matemáticas y probabilísticas en las mismas.

Imagine una red de telecomunicaciones, de la cual, se deben de estudiar algunas de sus operaciones. En muchos casos será suficiente con un monitoreo adecuado que permita caracterizar las tareas que son de nuestro interés. Sin embargo, si la red existiera únicamente en planos, si se quisieran analizar situaciones hipotéticas, si se quisiera predecir sus comportamiento, incluso, si la acción de monitoreo interfiriera con la misma calidad de los servicios, entonces se tendría que buscar un camino alternativo: un modelo abstracto que representara al sistema bajo estudio y sobre el que se pudieran efectuar acciones cuyas consecuencias permitieran inferir las respuestas que interesan. Construir un modelo como sustituto de un sistema real significa usar un lenguaje formal mediante el cual se describen las partes del sistema que parecen relevantes para nuestro problema y establecemos las relaciones entre estos componentes. Las construcciones de un lenguaje formal pueden manipularse mediante un conjunto de herramientas para producir inferencias sobre las propiedades planteadas por el modelo [30].

2.4.1.3. Simulación

La *simulación*, puede ser entendida como el proceso de representar un sistema real mediante la elaboración de un modelo mixto (matemático e informático), para entender el funcionamiento del sistema y poder predecir su comportamiento en el tiempo futuro, con el fin de mejorar su desempeño. Otros autores, plantean "simulación", como un proceso en el cual se diseña un modelo de un sistema real y se realizan experimentos con el para entender el comportamiento del sistema y/o evaluar estrategias alternas dentro de los límites impuestos por un criterio o por un conjunto de criterios. Lo importante a destacar, es que estas definiciones parten de dos fines específicos de la simulación: explicar y predecir. Modelar a través de la simulación facilita la comprensión y análisis de sistemas. Asimismo, permite

predecir el funcionamiento del sistema a futuro.

2.4.2. Definición y características

La *simulación de eventos discretos*, modela sistemas en los que las variables de estado cambian solo en un conjunto discreto de puntos en el tiempo. Los modelos de simulación, son analizados por métodos numéricos en lugar de por métodos analíticos. Los *métodos analíticos*, emplean el razonamiento deductivo de las matemáticas para "resolver" el problema. Por ejemplo, el cálculo diferencial, puede ser utilizado para computar la política de costo mínimo para algunos modelos de inventarios. Los *métodos numéricos*, emplean procedimientos computacionales para "resolver" modelos matemáticos. En el caso de modelos de simulación, lo cuales emplean métodos numéricos, los modelos son "ejecutados", en lugar de resueltos, esto es, se genera una historia artificial del sistema, basada en los supuestos del modelo, y se recolectan observaciones que serán analizadas para estimar medidas de rendimiento del sistema real. Los modelos de simulación del mundo real, son bastante grandes y la cantidad de información que almacenan es vasta, así que, usualmente, se suele ejecutar tales modelos en computadora [24].

De acuerdo con [21], todas las simulaciones de eventos discretos tienen una estructura común, sin importar el sistema que se modela. Las simulaciones, tendrán algunos de los componentes a continuación descritos:

1. *Calendarizador de eventos*: Mantiene una lista de eventos que se espera que sucedan. El calendarizador, permite que los eventos sean manipulados de varias maneras. Algunas de esas actividades de manipulación son las siguientes:

   a) Calendarizar un evento X en un tiempo t.

   b) Mantener un evento X durante un intervalo de tiempo dt.

   c) Cancelar un evento X, previamente calendarizado.

d) Calendarizar un evento X de duración indefinida.

El calendarizador de eventos, es uno de los componentes más frecuentemente ejecutados en una simulación. Es ejecutado antes de cada evento, y puede ser llamado muchas veces durante un evento para calendarizar nuevos eventos.

2. *Reloj de la simulación*: Cada simulación, tiene una variable global que representa el tiempo simulado. El calendarizador de eventos, es el responsable de adelantar este tiempo.

3. *Variables de estado del sistema*: Son variables globales que describen el estado del sistema. Por ejemplo, en una simulación de calendarización de una CPU, la variable de estado del sistema, es el número de tareas en cola, mientras que en tiempo de CPU asignado a cada tarea, es una variable local que debe ser almacenada en la estructura de datos que representa a la tarea.

4. *Rutinas de manejo de eventos*: Cada evento es simulado por su rutina. Estas rutinas, actualizan las variables de estado del sistema y calendarizan otros eventos.

5. *Rutinas de entrada*: Obtienen, por parte del usuario, parámetros del modelo. Por ejemplo, el número de simulaciones que se realizarán, con el objetivo de obtener estadísticas que resulten importantes para nosotros.

6. *Generador de reportes*: Son las rutinas de salida ejecutadas al final de la simulación. Ellas, calculan el resultado final y lo imprimen en un formato específico.

7. *Rutinas de inicialización*: Configuran el estado inicial de las variables de estado del sistema e inicializan varios generadores de números pseudo-aleatorios que se utilizarán durante la ejecución de la simulación.

8. *Rutinas de seguimiento*: Imprimen variables intermedias mientras la simulación está en

ejecución. Por otro lado, ayudan a depurar el programa de simulación.

9. *Administrador de la memoria dinámica*: El número de entidades en una simulación cambia constantemente. Esto requiere la ejecución periódica de un recolector de basura. Muchos lenguajes de programación de propósito general lo ejecutan automáticamente, caso contrario, el programador tiene la responsabilidad de escribir código para gestionar la memoria dinámica.

10. *Programa principal*: Llama a las rutinas de entrada, inicializa la simulación, ejecuta varias iteraciones y, finalmente, llama a las rutinas de salida.

En la simulación de eventos discretos, es necesario asegurar que los eventos ocurren en el orden propio y en el tiempo propio. Para simulaciones escritas en lenguajes de propósito general, es responsabilidad del programador implementar esta facilidad. Por otro lado, la calendarización de eventos es, usualmente, realizada al mantener una lista de eventos ordenados por tiempo. Cada evento, contiene el tiempo al cual éste deberá de ocurrir. Existen dos operaciones que son realizadas frecuentemente sobre esta lista:

1. Insertar nuevos eventos en la lista.

2. Encontrar el siguiente evento a ser ejecutado por el simulador y removerlo de la lista.

2.4.3. El problema de simular sistemas

Considere el problema de simular sistemas físicos [23], también llamados *redes*, el cual, consiste en uno o más procesos físicos. Cada proceso físico, representa un componente del sistema real y opera de manera autónoma, excepto al interactuar con otros procesos físicos en el sistema. La interacción se da a través de mensajes. Los contenidos de estos mensajes, dependen de las características de los procesos (su estado inicial o, quizá, sus reglas de operación) y los mensajes que el proceso ha recibido hasta un momento en particular. Cada proceso físico, es descrito por un conjunto de *eventos* y cada evento, tiene un tiempo de

ocurrencia. Además, los eventos pueden ordenarse con base a la noción de "causa y efecto", es decir, con base en una relación de dependencia causal ($\rightarrow$) entre todos los eventos del sistema. Sean e y e' eventos. Si $(e \rightarrow e')$, se dice que e ocurrió antes que e'. Muchos sistemas reales, pueden ser modelados en términos de procesos y mensajes. Por ejemplo, en una computadora, su CPU, disco duro y memorias, pueden ser pensados como procesos. La CPU, interactúa con la memoria al enviarle un mensaje solicitando o liberando espacio en ella. Los pasos típicos para construir y usar un programa de simulación consisten en [23]:

1. Iniciar con un sistema real y entender sus características.

2. Construir un modelo del sistema real, en el cual, se plasmen aspectos relevantes y se descarten aspectos irrelevantes de la simulación.

3. Construir una simulación del modelo, la cual, puede ser ejecutada en una computadora.

4. Analizar las salidas de la simulación, para entender y predecir el comportamiento del sistema real.

2.4.4. Verificación y validación

Durante el desarrollo de un modelo de simulación, se debe asegurar que el modelo esté correctamente implementado y que sea representativo del sistema real. Estos dos pasos, son llamados *verificación y validación* del modelo, respectivamente. A continuación, se definen, a detalle, ambos conceptos [24]:

Verificación: También llamada depuración, esto es, asegurar que el modelo hace lo que debe de hacer. La verificación, está relacionada con la correcta implementación del modelo, asimismo, se relaciona con la comparación entre el modelo y el código de programación que lo implementa. La verificación, formula las siguientes preguntas: ¿el modelo está implementado correctamente en el software de simulación o lenguaje de propósito general?, ¿están los parámetros de entrada y la estructura lógica del modelo representados correctamente?

Validación: Está relacionada con construir el modelo correcto. La validación, intenta confirmar que el modelo es una representación exacta del sistema real. Es, usualmente, realizada mediante la calibración del modelo, un proceso iterativo que compara el comportamiento del modelo con el sistema actual y usa las discrepancias entre los dos, y los conocimientos obtenidos, para mejorar el modelo. Este proceso es repetido hasta que la precisión del modelo es juzgada como aceptable.

2.5. Sistemas distribuidos

De acuerdo con [43], a continuación se detallará que son los sistemas distribuidos y diversos conceptos inherentes a ellos:

2.5.1. Definición

Un sistema distribuido, es aquel en el que los componentes, ubicados en computadoras en red, se comunican y coordinan sus acciones solo a través de paso de mensajes. En otras palabras, un *sistema distribuido*, consiste en una colección de computadoras autónomas conectadas a través de una red computacional y equipadas con software de sistemas distribuidos. Este software, permite a las computadoras coordinar sus actividades y compartir los recursos del hardware, software y datos del sistema. Un software de sistemas operativos distribuidos sobre una colección de computadoras independientes, conectadas en red, comunicadas y físicamente separadas. Cada computadora individual, mantiene un subconjunto de software específico del sistema operativo global. Cada subconjunto, está compuesto por dos distintos proveedores de servicio:

1. Un kernel mínimo ubicuo, o microkernel, que directamente controla el hardware de la computadora en cuestión.

2. Una colección de alto nivel de componentes de gestión del sistema, que coordina las actividades individuales y colectivas de la computadora en cuestión.

El microkernel y la colección de componentes de gestión, trabajan juntos. Ellos soportan el objetivo del sistema de integrar múltiples recursos y procesar funcionalidad en un sistema estable y eficiente. Esta perfecta integración de computadoras individuales en un sistema global, es conocida como *transparencia*, describiendo la ilusión de los usuarios de la apariencia del sistema global, como una única entidad computacional. Las computadoras que están conectadas por una red, pueden estar espacialmente separadas por cualquier distancia. Pueden estar en continentes separados, en el mismo edificio o en la misma habitación.

En términos generales, puede pensarse en un sistema distribuido como en una colección heterogénea de componentes de hardware, software y datos, interconectados por algún tipo de infraestructura de comunicaciones, mediante la que colaboran para ofrecer un servicio relacionado con el manejo de la información. Por ejemplo, transacciones sobre una base de datos distribuida, cálculos científicos, control de procesos en tiempo en real, etc. Las dificultades de su construcción, tienen su origen en las limitaciones de los componentes con los que se integra un sistema distribuido, así como en la diversidad de características que deben reunirse y coordinarse en un todo funcional. Se desea que un sistema distribuido garantice la calidad de sus servicios, a pesar de que un cierto número de componentes falle o se aleje de sus especificaciones de operación [3].

Ejemplos de sistemas distribuidos, se enuncian a continuación:

1. La Internet.

2. La World Wide Web.

3. Redes P2P.

2.5.2. Ventajas

Los sistemas distribuidos, ofrecen las siguientes ventajas:

1. Intercambio de información.

2. Uso compartido de recursos.

3. Confiabilidad mediante replicación con base en tres tipos de redundancias:

   a) En componentes físicos: Añadiendo componentes extras de hardware al sistema.

   b) De información: Creando copias de la información en diferentes partes del sistema.

   c) En tiempo: Repitiendo operaciones hasta que se completen con éxito cada una de ellas.

4. Desempeño mediante paralelización.

2.5.3. Retos

Por otro lado, existen diversos retos que hay que cubrir en los sistemas distribuidos, por ejemplo:

1. Heterogeneidad: Existen diferencias en los componentes de un sistema distribuido que se deben resolver, por ejemplo, redes, hardware y sistemas operativos.

2. Apertura/Seguridad: Un sistema distribuido, debe ser seguro, resolviendo, por ejemplo, problemas criptográficos en la comunicación.

3. Escalabilidad: Un sistema distribuido, debe ser escalable, por ejemplo, al añadirle componentes nuevos de manera sencilla.

4. Fallas: Los componentes de un sistema distribuido, están sujetos a fallar en cualquier momento de la vida del sistema.

5. Transparencia: El sistema debe ser transparente para el usuario final, es decir, el usuario

no debe notar que existe redundancia, fallas, etc.

2.5.4. Propiedades y modelos

Las propiedades de un sistema distribuido, se enuncian a continuación:

1. Concurrencia: En una red de computadoras, existe, usualmente, una ejecución simultánea de programas. Dos usuarios separados pueden trabajar en sus computadoras mientras comparten recursos, tales como páginas web o documentos, cuando sea necesario. La capacidad del sistema de gestionar los recursos compartidos, puede ser mejorada al agregar más recursos (por ejemplo, computadoras) a la red. La coordinación de los programas, ejecutándose concurrentemente, que comparten recursos, es también un tema vital y recurrente.

2. No existencia de un reloj global: Cuando los programas necesitan colaborar, coordinan sus acciones al intercambiar mensajes. La estrecha coordinación, depende, principalmente, de la idea compartida del momento en el que las operaciones de los programas tienen lugar. Sin embargo, existen restricciones a la precisión de la sincronización del reloj por parte de las computadoras en red. Esto es, no existe un concepto único global del tiempo correcto. Este es un resultado directo del hecho de que la comunicación puede ser realizada solo por enviar mensajes a través de una red.

3. Dificultad para usar una memoria global: Por lo que se recurre al uso de algoritmos de snapshot, por ejemplo, Chandy-Lamport [44].

4. Fallas independientes: Cada computadora del sistema, es susceptible a fallar, por lo tanto, es obligación de los diseñadores del sistema hacer planes si se presentan salidas incorrectas o fallas en los componentes del sistema.

    a) Omisión: Un componente del sistema, omite la transmisión o recepción de algunos mensajes.

b) Bizantina: Un componente del sistema, exhibe un comportamiento totalmente arbitrario.

c) Paro: Un componente del sistema, se desvía de su comportamiento a partir de cierto momento y para siempre.

d) Paro con recuperación: Un componente del sistema, se desvía de su comportamiento a partir de cierto momento, pero luego regresa.

Para realizar simulaciones de sistemas distribuidos, se emplean los siguientes tipos de modelos:

1. De comunicación: Por ejemplo, paso de mensajes o memoria compartida.

2. De ejecución: Por ejemplo, basados en eventos o basados en estados.

3. De tiempo: Por ejemplo, asíncrono o síncrono.

4. De fallas: Por ejemplo, omisión, bizantina, paro o paro con recuperación.

Las entidades que componen un sistema distribuido, realizan en conjunto una serie de tareas en común, para las que es necesario intercambiar información en el espacio y el tiempo. También se sabe que dichas entidades están expuestas a contingencias que pueden manifestarse de diferentes formas. En suma, un modelo debe ser capaz de describir la dinámica del sistema. Esto es, las posibilidades de interacción de cada una de las partes o componentes activos. En consecuencia, el planteamiento de un modelo comienza por reconocer los elementos encargados del procesamiento de la información. Enseguida, deben establecerse los mecanismos que hacen posible el intercambio de información entre estos elementos. Ello implica caracterizar los medios con que se sustenta la comunicación. Por otra parte, modelar al sistema implica, también, definir un orden en las interacciones. Finalmente, tratándose de describir el comportamiento en el largo plazo, también es importante considerar la posibili-

dad de que algún componente se desvíe de sus especificaciones y muestre un comportamiento no deseado. Visto de otra forma, deben definirse las fallas que pueden presentarse [3].

2.5.5. El modelo de comunicación asíncrona por paso de mensajes

En esta investigación, se usó el modelo de comunicación asíncrona por paso de mensajes. De acuerdo con [3], a continuación, se detallan diversos aspectos con respecto a tal modelo.

La investigación en el campo de la computación distribuida, comienza cuando se reconoce un problema de significación práctica y se construye una versión abstracta que puede abordarse mediante tratamiento matemático. En ese momento, se pasa del sistema real al modelo que lo representa. Luego, se propone un algoritmo que resuelve el problema, se le describe y se demuestra que funciona, de acuerdo con las condiciones del modelo.

El modelo propuesto, cuenta con las siguientes ventajas:

1. Recoge las propiedades sobresalientes del sistema real que se busca describir.

2. Ofrece escenarios o condiciones de ejecución independientes de la tecnología. De esta forma es posible valorar el desempeño de un procedimiento o una operación y compararlos, en condiciones justas, con otros de su tipo.

3. Es capaz de capturar y revelar las limitaciones propias de la construcción de un sistema.

El modelo de comunicación asíncrona con intercambio de mensajes, considera un conjunto de entidades activas, denominadas procesos, que se ejecutan sobre diferentes sitios o plazas. Estas entidades, se relacionan al intercambiar mensajes a través de los canales de comunicación que las unen. Un canal que comunica a dos procesos, funciona como una estructura de datos sobre la que el transmisor guarda su mensaje, sin esperar respuesta, mientras el receptor lo recoge de acuerdo con una política de servicio que generalmente corresponde al tipo cola o FIFO (first in-first out). Generalmente, cada sitio tiene asociado un identificador

que lo distingue del resto de los componentes con los que intercambia información. La red de comunicaciones que comprende estos sitios y canales, es caracterizada por medio de una gráfica conexa $G = (V, E)$ y se asume que existe un proceso emplazado en cada vértice. Cada proceso es capaz de reconocer la arista por donde recibe un mensaje, aún cuando desconozca la identidad del transmisor.

En suma, y a menos que se establezca otra cosa, en el modelo de sistema distribuido basado en comunicación asíncrona por paso de mensajes, se supone un conjunto finito de procesos $\pi = \{p_1, ..., p_n\}$ que se comunican enviando y recibiendo mensajes de longitud acotada a través de canales de comunicación. Cada pareja de procesos p_i y p_j, se conecta por un canal asíncrono sin perdida, denotado (i, j). Más aún, se asume que los procesos no comparten ningún tipo de memoria común, ni disponen de una de referencia universal de tiempo.

Sobre las entidades activas del modelo, se pueden ejecutar algoritmos distribuidos, entendidos estos, como un programa de acciones que se encuentra replicado en cada uno de los sitios que participan en el sistema y, a partir del cual, se da una secuencia de interacciones entre estos. Por ello, se dice que todos los procesos que intervienen son simétricos. Es decir, que cada componente se encarga de ejecutar una versión del mismo conjunto de instrucciones a partir de sus circunstancias particulares como son: su identidad, los sitios vecinos con los que interactúa, el orden relativo en que recibe sus mensajes, etc. Normalmente, se supone que ninguna de las entidades activas dispone de un conocimiento total o global sobre el sistema al que pertenece. Esto es, sus reacciones obedecen al conocimiento parcial que dispone y que corresponde con su entorno local.

En particular, se usaron como base dos algoritmos distribuidos propuestos por Adrian Segall en [31]:

1. Algoritmo de propagación de información (PI, por sus siglas en inglés).

2. Algoritmo de propagación de información con retroalimentación (PIF, por sus siglas en inglés).

Tales algoritmos, son descritos a continuación.

2.5.6. El algoritmo de propagación de información

El algoritmo PI, considera la situación donde un número de unidades de cómputo, físicamente distintas, trabajan en un problema común, mientras su operación es coordinada mediante los canales de comunicación que conectan a dichas unidades. Cada unidad de cómputo, tiene cierta capacidad de memoria y procesamiento, y está programada para realizar su parte del cómputo, así como también, para recibir y enviar mensajes de control sobre los canales de comunicación. El algoritmo PI, considera una red, cuya gráfica subyacente es $G = (V, E)$. Para el modelo de red asíncrona por paso de mensajes con el que se trabaja en esta investigación, se requiere que G sea conexa. Suponga que un vértice $v \in V$, recibe del mundo exterior una pieza de información que tiene que ser transmitida a todos los vértices de la red. La manera más simple de realizar esto, es "inundar la red", es decir, v transmite un mensaje que contiene la información a todos sus vecinos, estos lo reciben y lo reexpiden por todas sus líneas de comunicación la primera vez que lo reciben. A su vez, cada vértice que lo reciba, efectuará la misma operación. Suponemos que un vértice arbitrario en la red se envía un mensaje a sí mismo que arranca el algoritmo.

Los atributos y mensajes requeridos en el algoritmo PI, se describen a continuación:

Los atributos requeridos en el i-ésimo vértice, son:

1. **visitado**: Es una variable booleana, que inicialmente se encuentra establecida en $falso$. Esta variable cambia su valor a *verdadero* si ya recibió algún mensaje propio del algoritmo PI.

2. **vecinos**: Representa el conjunto de identificadores de los vértices adyacentes al i.

3. **padre**: Indica el vértice desde el que i recibe M por primera vez. Inicialmente está establecido en nulo.

Los mensajes intercambiados por el algoritmo son:

1. ***M***: Mensaje enviado y recibido por los vértices participantes en el algoritmo PI.

El Algoritmo 4, muestra la lógica de programación del algoritmo PI.

Algoritmo 4 Propagación de información en el vértice i

Requiere:

Una gráfica $G = (V, E)$ conexa.

Si se recibe el mensaje M desde el vértice j **Entonces**
  Si $visitado = falso$ **Entonces**
    $padre \leftarrow j$
    $visitado \leftarrow verdadero$
    Para cada $k \in vecinos$ **Hacer**
      Envía M a k
    Termina Para
  Termina Si
Termina Si

En el algoritmo PI, se cumplen las siguientes propiedades:

1. Todos los vértices de la red reciben el mensaje M y terminan su ejecución local.

2. La propagación de la información es lo más rápida posible, en el siguiente sentido: $\forall\ (i, j) \in E$, sea $w_{i,j}$ el retardo de transmisión del mensaje enviado de i a j. Entonces, el conjunto de aristas $\{(padre, i)\ \forall\ i \in V, i \neq p\}$, forma un árbol T de pesos mínimos para G.

2.5.7. El algoritmo de propagación de información con retroalimentación

Algunas veces, en el algoritmo PI, el vértice que arranca el algoritmo pudiera querer estar en posibilidades de informarse cuando la información ha alcanzado a todos los vértices de la red. Es decir, el algoritmo termina *localmente* en cada vértice sin que el proceso iniciador pueda saber cuándo el algoritmo termina *globalmente*, esto sucede cuando la información transmitida alcanza al último vértice en la red. La corrección a este problema es un algoritmo que toma como base el PI, llamado PIF. En PIF, se incorpora una nueva etapa de retroalimentación, en donde cada vértice que reside en las hojas del árbol T inducido por PI, al terminar de recibir todos los mensajes M, envía un acuse de recibo a su padre. Los vértices de los niveles anteriores, aguardan este informe por cada una de sus aristas incidentes, incluyendo aquellas que los conectan con sus hijos. Una vez que han recibido todos los reportes que esperan, devuelven el mensaje M a sus respectivos padres. Esta información va remontando hacia la raíz del árbol, donde reside el vértice iniciador del algoritmo. Al llegar la información a la raíz, entonces el vértice iniciador del algoritmo, puede saber cuándo ha terminado la propagación de la información en toda la red. Esto dota al algoritmo PIF de la propiedad de terminación global, manteniendo las dos propiedades, mencionadas anteriormente, del algoritmo PI.

Los atributos y mensajes requeridos en el algoritmo PIF, se describen a continuación:

Los atributos requeridos en el i-ésimo vértice, son:

1. **visitado**: Es una variable booleana, que inicialmente se encuentra establecida en $falso$. Esta variable cambia su valor a *verdadero* si ya recibió algún mensaje propio del algoritmo PIF.

2. **N(k)**: Es una tabla clave-valor, donde las claves son los identificadores de los vértices vecinos de i, y los valores asociados a las claves están todos, inicialmente, establecidos

en 0. Cuando i recibe M desde k, entonces $N[k] = 1$.

3. **padre**: Indica el vértice desde el que i recibe M por primera vez. Inicialmente está establecido en nulo.

4. **vecinos**: Representa el conjunto de identificadores de los vértices adyacentes al i.

Los mensajes intercambiados por el algoritmo son:

1. ***M***: Mensaje enviado y recibido por los vértices participantes en el algoritmo PIF.

El Algoritmo 5, muestra la lógica de programación del algoritmo PIF.

2.5.8. El sincronizador β

Los algoritmos asíncronos son, en muchos casos, sustancialmente inferiores en términos de complejidad a los correspondientes algoritmos síncronos, y su diseño y análisis es mucho más complicado. En una red síncrona, se permite que los mensajes sean enviados solo en tiempos enteros, o *pulsos*, de un reloj global. Cada vértice en la red tiene acceso a ese reloj. A lo más, un mensaje puede ser enviado sobre una arista dada en un cierto pulso. El retardo de cada arista es, a lo más, una unidad de tiempo del reloj global. Las aseveraciones de un sistema síncrono, son opuestas a las de un sistema asíncrono, por ello, resulta útil desarrollar una técnica de simulación general conocida como *sincronizador*, la cual, permite al desarrollador escribir un algoritmo como si se ejecutara en una red síncrona. En particular, existe el *sincronizador* β, el cual, requiere de una fase de inicialización, en la cual se elige un líder s en la red y un árbol de expansión con raíz en s, es construido. El sincronizador β, opera como sigue: Después de la ejecución de cierto pulso, el líder, eventualmente aprenderá que todos los vértices en la red son *seguros* (se dice que un vértice es seguro con respecto a cierto pulso, si cada mensaje del algoritmo síncrono enviado por ese vértice en ese pulso, ya ha llegado a su destino, en ocasiones, se requiere que un mensaje de confirmación sea devuelto al vértice origen, así cuando todos sus mensajes hayan sido confirmados, se puede decir que es seguro),

Algoritmo 5 Propagación de información con retroalimentación en el vértice i

Requiere:

Una gráfica $G = (V, E)$ conexa.

Si se recibe el mensaje M desde el vértice j **Entonces**
  $N[j] \leftarrow 1$
  Si $visitado = falso$ **Entonces**
    $padre \leftarrow j$
    $visitado \leftarrow verdadero$
    Si $vecinos \setminus \{padre\} \neq \emptyset$ **Entonces**
      Para cada $k \in vecinos \setminus \{padre\}$ **Hacer**
        Envía M a k
      Termina Para
    Termina Si
  Termina Si
  Si $N[k] = 1 \ \forall \ k \in vecinos$ **Entonces**
    Si $padre \neq i$ **Entonces**
      Envía M a $padre$
    Termina Si
  Termina Si
Termina Si

en ese tiempo, realiza un mensaje de multidifusión (*broadcast*) de un cierto mensaje a lo largo del árbol, notificando a todos los vértices que pueden generar un nuevo pulso. El tiempo anterior, es detectado mediante un patrón de comunicación denominado *convergecast*, el cual se inicia en las hojas del árbol y termina en la raíz. Es decir, siempre que un vértice aprende que es seguro, y todos sus descendientes en el árbol también lo son, reporta este hecho a su padre [33].

Capítulo 3

Metodología

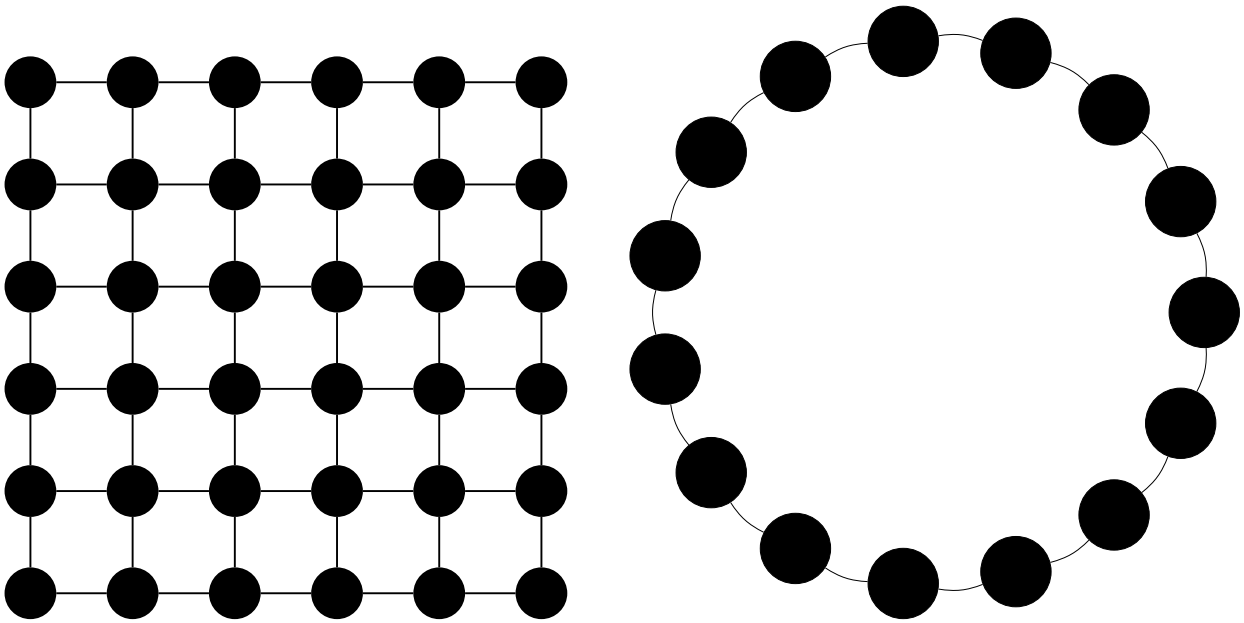
"No hay amor en un átomo de carbono, ni huracán en una molécula de agua, ni colapso financiero en un billete de un dólar" -Peter Dodds

3.1. Introducción

El desarrollo de esta investigación, se trabajó sobre dos topologías de red:

1. Malla: Es aquella topología que consta de i filas y j columnas, $i, j \in \mathbb{N}$, en donde cada uno de sus vértices $v \in V$, se conecta a sus cuatro vecinos más cercanos (hacia la izquierda, derecha, arriba y abajo), formando un vecindario $N(v)$ tipo Von Neumman [29], tal y como se muestra en la Figura 3.1a. En los experimentos de esta investigación, se usaron mallas cuadradas tal que $i = j$.

2. Anillo: Es aquella topología simétrica, en donde cada uno de sus vértices $v \in V$, se conecta exactamente a sus dos vecinos más cercanos, formando una única ruta continua, tal y como se muestra en la Figura 3.1b.



(a) Malla con 6×6 vértices (b) Anillo con 15 vértices

Figura 3.1: Topologías usadas en esta investigación

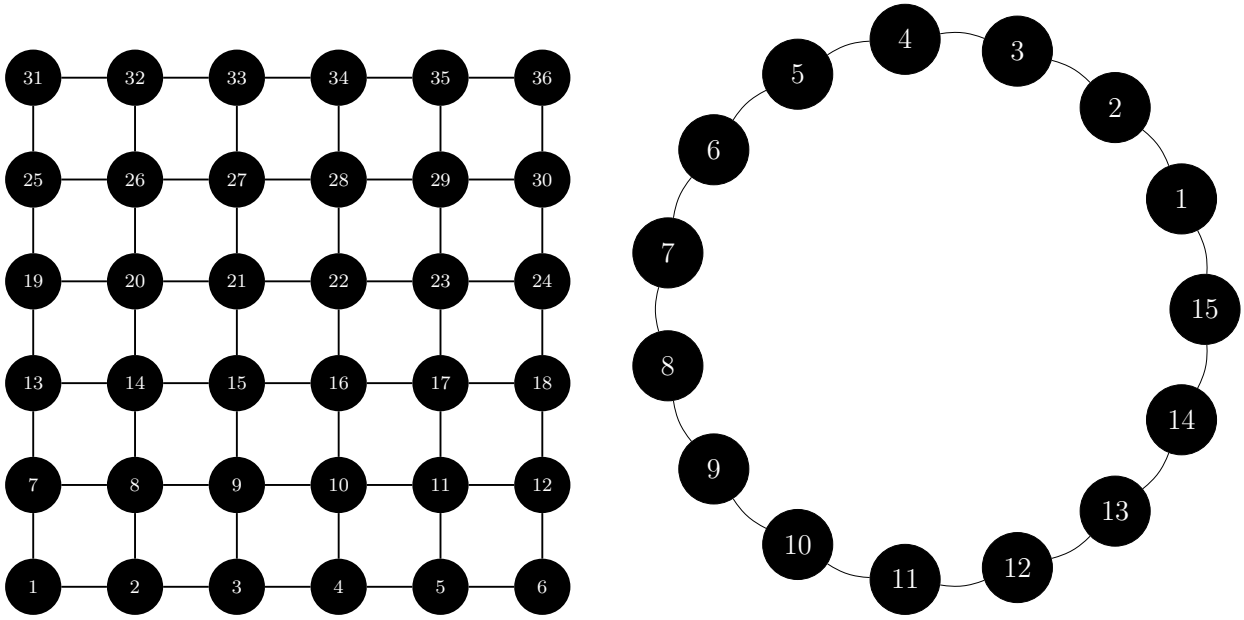
3.2. Etapa 1: Formación de redes

3.2.1. Generalidades

Los experimentos de esta fase, se basan en la forma en que las personas establecen vínculos, por ejemplo, se puede pensar en una persona arbitraria, que en un inicio, establece vínculos con sus conocidos cercanos (familia, vecinos, compañeros de trabajo), sin embargo, conforme avanza el tiempo, usará los enlaces que tiene con su comunidad cercana para conocer gente nueva y crear vínculos lejanos. Los experimentos, siguen esta analogía, un vértice mediante el envío de mensajes adquiere conocimiento parcial sobre la topología de la red subyacente, dicho vértice, usará a su conjunto de vecinos para encontrar aquellos vértices en la red que son potencialmente útiles para establecer atajos, permitiéndole a los mensajes que emite, llegar más rápidamente a regiones lejanas a su entorno local inicial. El experimento, considera los procesos de envío de mensajes que se producen en la red, de modo que este

proceso de intercambio, permite a cada vértice adquirir información sobre la topología subyacente. La forma en que los vértices consiguen dicha información es a través de "paquetes exploradores", los cuales, guardan la ruta a través de la cual viajan y mediante un conjunto de sencillas reglas locales de reconexión, les permitirán seleccionar aquellos vértices de la red que son potencialmente útiles para establecer enlaces de largo alcance o atajos.

Las topologías usadas, son consideradas sistemas distribuidos, donde los vértices, son las entidades activas que procesan información y las aristas, son los enlaces de comunicación mediante los cuales los vértices intercambian información. Cada vértice $v \in V$ de ambas topologías, es etiquetado con un número $id \in \mathbb{N}$, en orden ascendente, tal y como se muestra en las Figuras 3.2a y 3.2b.



(a) Malla con 6×6 vértices etiquetados

(b) Anillo con 15 vértices etiquetados

Figura 3.2: Topologías usadas en esta investigación con identificadores $\mathbb{N}$ en sus vértices

Posteriormente, las gráficas son embebidas, o empotradas, en un espacio Euclidiano, donde cada vértice tiene asociada una coordenada en el eje x y una en el eje y, las cuales, representan un punto en $\mathbb{R}^2$. Para el caso de la malla, se asume trabajar en el primer cuadrante

del plano cartesiano, tal y como se muestra en la Figura 3.3, considerando el punto $\mathrm{O}(0,0)$, como origen del plano.

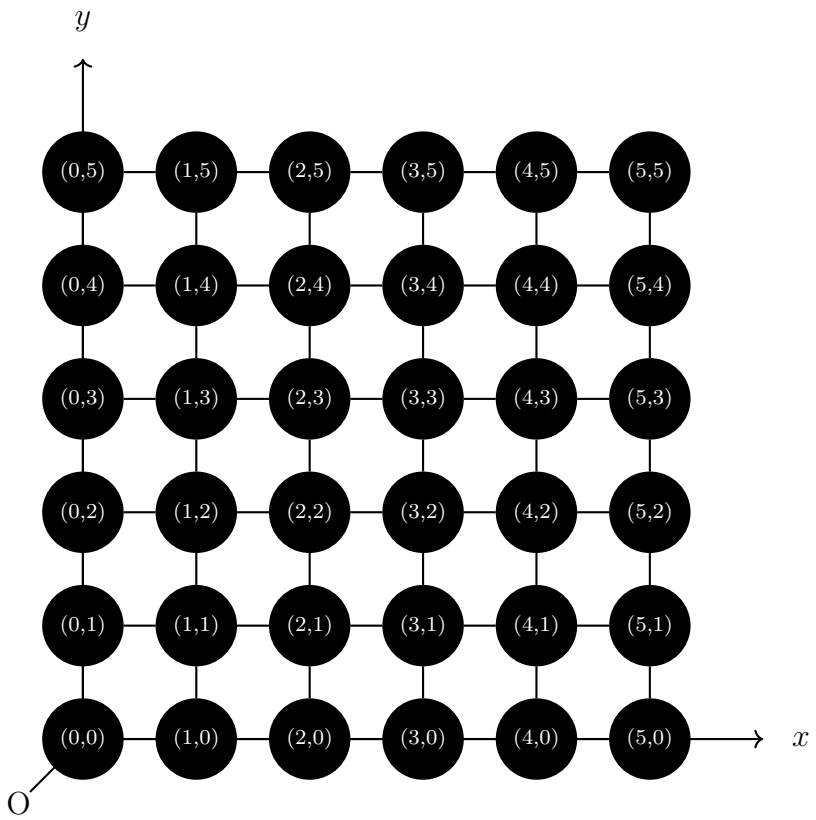


Figura 3.3: Malla con 6×6 vértices asignados al espacio Euclidiano

Por otro lado, para el caso del anillo, se hizo uso del sistema de coordenadas polares para poder obtener las coordenadas rectangulares correspondientes, como sigue:

Suponga que se tienen que calcular las coordenadas rectangulares de los vértices de una red G con topología anillo de orden n, entonces, para obtener las coordenadas rectangulares de cada vértice $v \in V$, se aplica, primero, la Ecuación 3.1:

$$factor = \frac{360}{n} \tag{3.1}$$

Posteriormente, se obtiene el ángulo θ correspondiente a cada vértice $v \in V$ aplicando

la Ecuación 3.2:

$$\theta = factor \cdot id \tag{3.2}$$

Lo cual, daría como resultado los ángulos θ correspondientes a cada vértice $v \in V$. Sean $r = 4$ y $n = 15$, se puede, entonces, graficar a los vértices en el plano polar, a través del sistema de coordenadas polares, donde el polo es representado por el punto O(0,0°), y cada vértice; por el punto (r, θ), tal y como se muestra en la Figura 3.4.

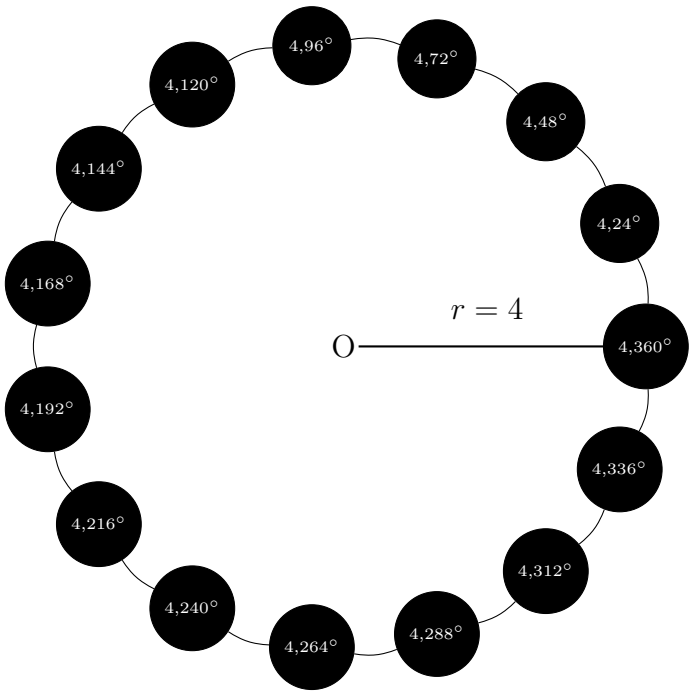


Figura 3.4: Anillo con 15 vértices graficados en coordenadas polares

Finalmente, para encontrar las coordenadas rectangulares (x, y) de cada vértice $v \in V$, se debe hacer uso de las Ecuaciones 3.3 y 3.4, respectivamente:

$$x = r \cdot \cos\theta \tag{3.3}$$

$$y = r \cdot \sin\theta \tag{3.4}$$

Asumiendo el punto O(0,0) como origen del espacio Euclidiano, el resultado se puede visualizar gráficamente en la Figura 3.5.

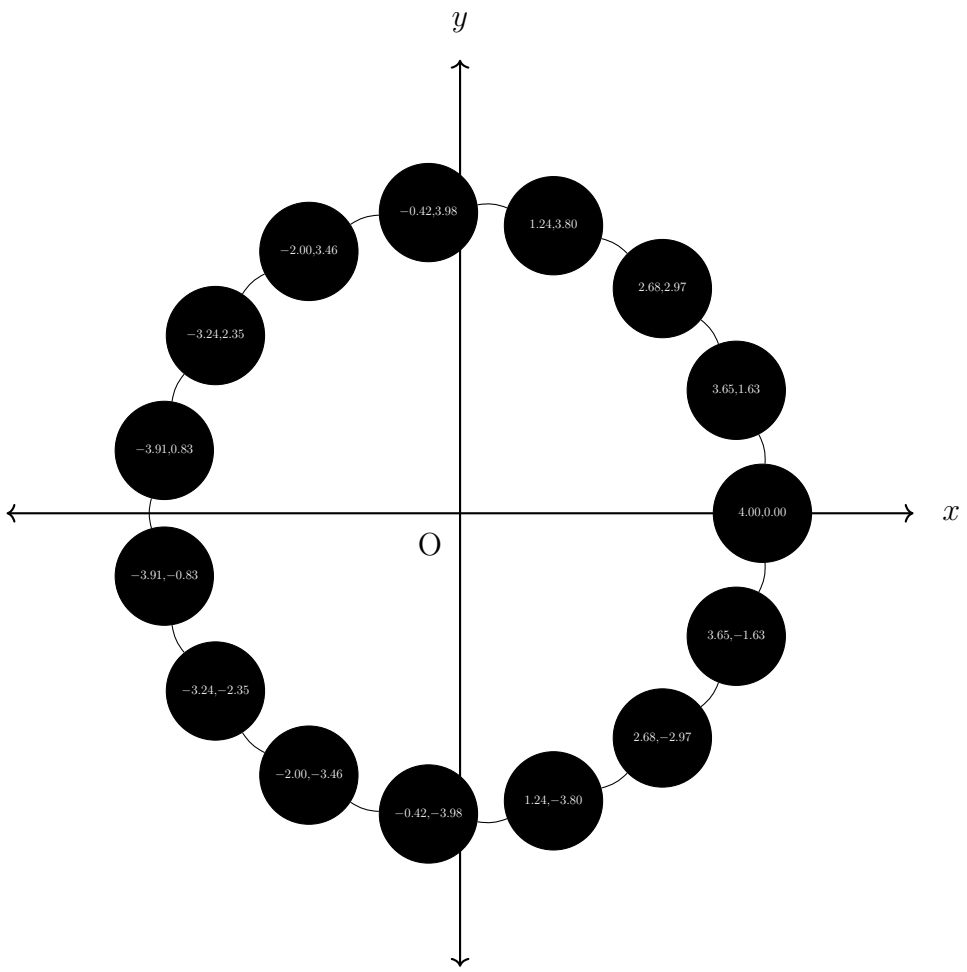


Figura 3.5: Anillo con 15 vértices graficados en coordenadas rectangulares

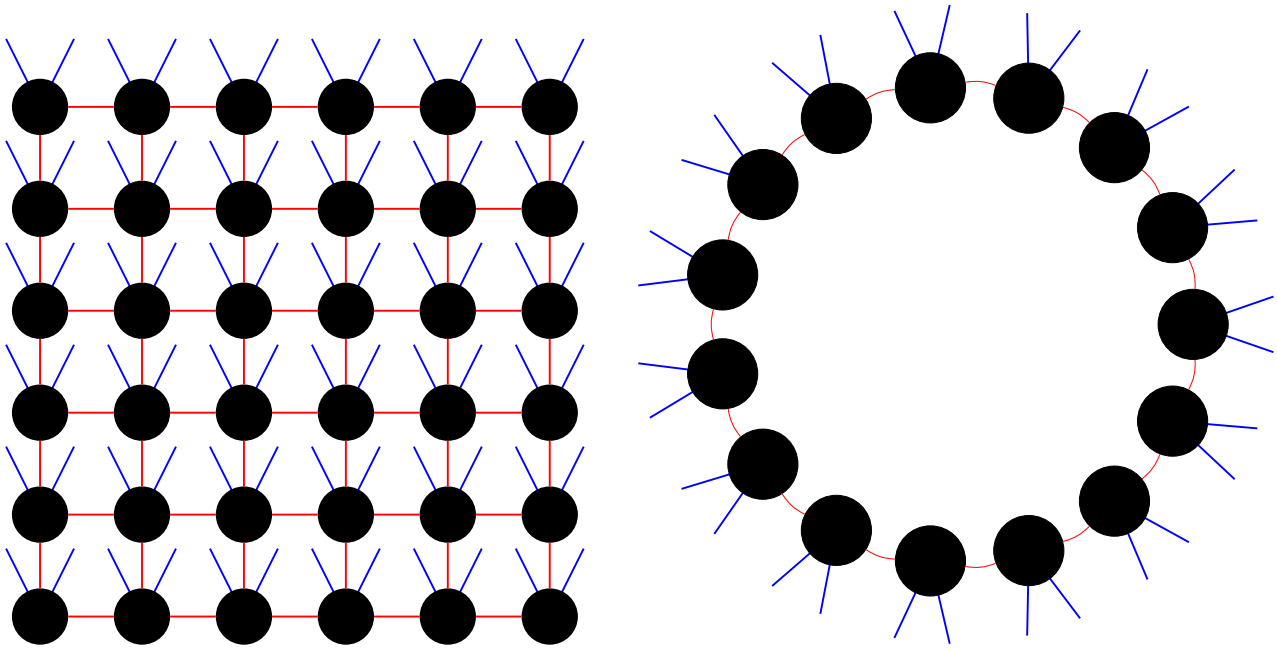
Se asume que cada vértice $v \in V$ de ambas topologías, cuenta con dos tipos de aristas:

1. Fijas: Son aquellas aristas con las que cuenta la topología inicial.

2. Dinámicas: Cada vértice, tiene un conjunto de z aristas dinámicas, identificadas con

números $id\_arista \in \{1, ..., z\}$ (en esta investigación se considera $z = 2$), las cuales, podrá reconectar a regiones lejanas a su entorno local inicial, estableciendo atajos o caminos más cortos.

Ejemplos de aristas fijas (marcadas en color rojo) y dinámicas (marcadas en color azul), se muestran en las Figuras 3.6a y 3.6b.



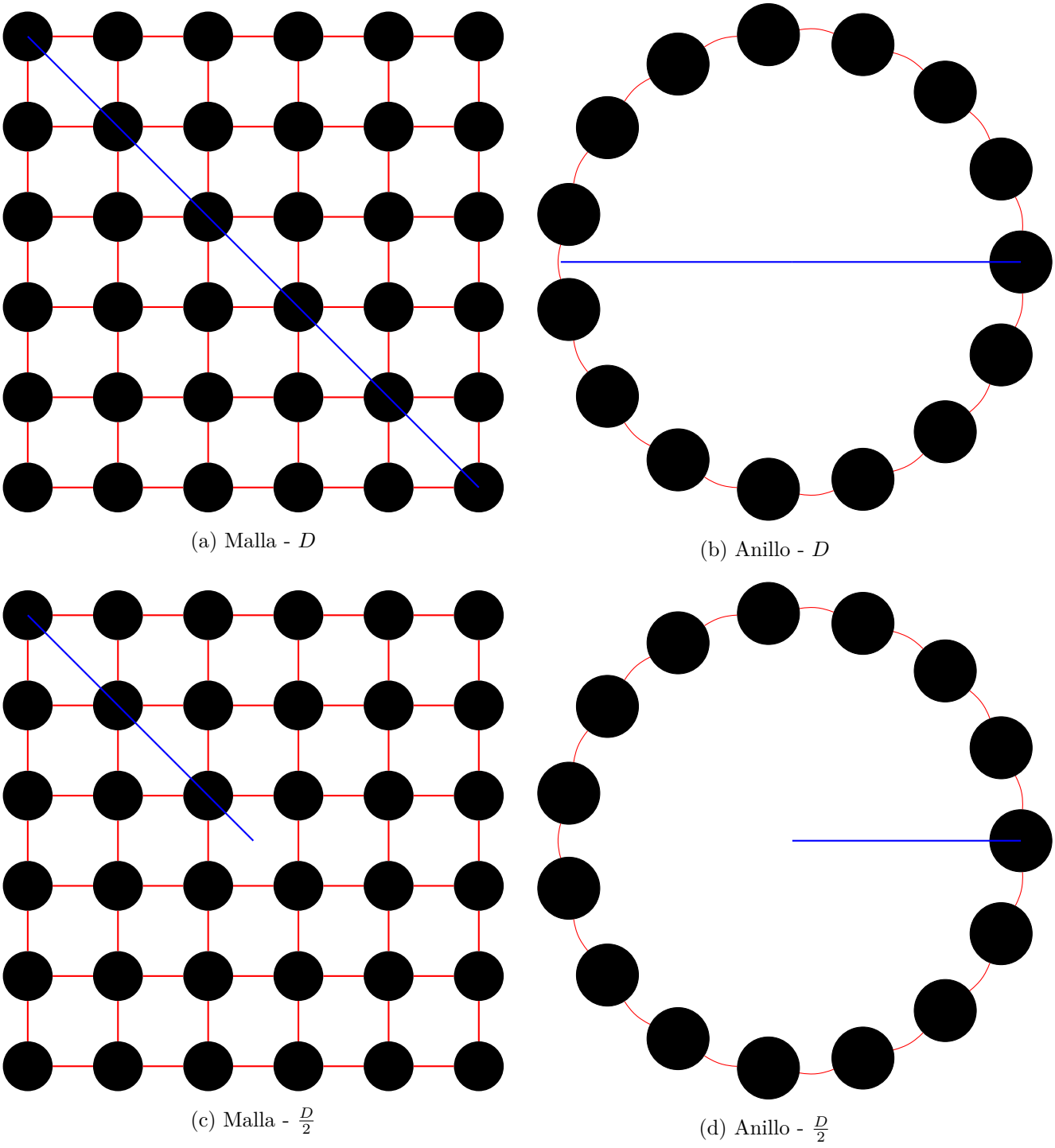
(a) Malla con aristas fijas y dinámicas

(b) Anillo con aristas fijas y dinámicas

Figura 3.6: Topologías usadas en esta investigación con aristas fijas y dinámicas

Cada arista dinámica, tiene una longitud máxima a la que puede llegar dentro del espacio en que está empotrada o embebida la red (espacio Euclidiano). En esta investigación, se consideran las siguientes longitudes: D, $\frac{D}{2}$, $\frac{D}{4}$, $\frac{D}{8}$ y $\frac{D}{16}$. Donde D, en el caso de la topología anillo, es la longitud del diámetro $d = 2 \cdot r$ de la circunferencia que lo representa, y, en el caso de la malla; la longitud de la diagonal principal. Ejemplos de las longitudes D y $\frac{D}{2}$ de las aristas dinámicas, se muestran en las Figuras 3.7a, 3.7b, 3.7c y 3.7d.

Como se mencionó anteriormente, las topologías de red propuestas son consideradas



(a) Malla - D

(b) Anillo - D

(c) Malla - $\frac{D}{2}$

(d) Anillo - $\frac{D}{2}$

Figura 3.7: Ejemplos de tamaño D y $\frac{D}{2}$ de arista dinámica

sistemas distribuidos y, al no poder trabajar con los sistemas reales, surge la necesidad inminente de usar un modelo que los describa. Bajo ese contexto, se hizo uso de los siguientes tipos de modelos:

1. De comunicación: Se usó el modelo de comunicación por paso de mensajes, en donde se destaca el papel de la red de comunicaciones. En cada vértice de la red, se procesan los mensajes que se reciben desde sus vecinos, se realiza un cálculo local y se envían mensajes a algunos vecinos. Todos los mensajes tienen una longitud acotada y solo pueden transportar una cantidad limitada de información. Se garantiza que las aristas, transportan sin pérdida los mensajes que manejan en su interior, con retardo finito, pero impredecible.

2. De tiempo: Se usó el modelo de tiempo asíncrono, donde no se tienen límites en el retardo de los mensajes, se deriva de su reloj o el tiempo para ejecutar un paso. Decir que un sistema es asíncrono, significa que no se hace ninguna suposición relativa a sus tiempos de ejecución.

3. De fallas: Se asume que no existen fallas en los componentes del sistema distribuido.

Más aún, se usó el modelo de comunicación asíncrona por paso de mensajes descrito en la Sección 2.5.5. En esta investigación, se desea que dicho modelo pudiera simular un comportamiento síncrono para ejecutar sobre el un algoritmo de la misma naturaleza. En particular, se hace uso de un *sincronizador* β para lograr tal objetivo.

3.2.2. Las fases del modelo de comunicación asíncrona por paso de mensajes

3.2.2.1. Generalidades

Como se pudo observar en el capítulo anterior, el algoritmo PIF, toma como base al algoritmo PI, al agregar una nueva etapa de retroalimentación. En esta investigación, se

toma como base al algoritmo PIF, agregándole cierta porción de código, para desarrollar un nuevo algoritmo que permite la formación de redes con características muy similares a las encontradas en las redes complejas. El algoritmo PIF, implementa un sincronizador β por defecto, lo cual permite simular la sincronización dentro del modelo asíncrono, ya que el dicho algoritmo induce un árbol generador, en donde el vértice raíz, recibe noticias de que los vértices restantes de la red han completado su parte del algoritmo (propiedad de terminación global). Los experimentos propuestos, se basan en ciclos de conexión-reconexión de aristas dinámicas y, a su vez, cada ciclo se compone de tres fases, las cuales, se enuncian a continuación:

1. **Fase de exploración**.

2. **Fase de negociación**.

3. **Fase de conexión-reconexión de aristas dinámicas**.

Cada fase, es llevada a cabo mediante un algoritmo PIF, al que se le agrega cierto código de acuerdo a su lógica de programación. Los algoritmos de cada fase, se nombran y definen, como sigue:

1. Fase de exploración, se lleva a cabo mediante el algoritmo **PIF-Exploración**, agregándole, al algoritmo PIF, la lógica necesaria para ejecutar, dentro de él, un intercambio de paquetes exploradores, que serán enviados con ayuda de algún algoritmo de encaminamiento.

2. Fase de negociación, se lleva a cabo mediante el algoritmo **PIF-Negociación**, agregándole, al algoritmo PIF, la lógica necesaria para ejecutar, dentro de él, un intercambio de solicitudes de conexión-reconexión de aristas dinámicas.

3. Fase de conexión-reconexión de aristas dinámicas, se lleva a cabo mediante el algoritmo **PIF-Conexión**, agregándole, al algoritmo PIF, la lógica necesaria para ejecutar, dentro

de él, la conexión-reconexión formal de aristas dinámicas..

En cada experimento, se realizaron 50 ciclos de ejecución, es decir, cada vértice de la red, ejecuta 50 veces las tres fases mencionadas. A continuación, se describe la propuesta de algoritmo de formación de redes complejas, realizada en esta investigación:

3.2.2.2. Atributos requeridos en el i-ésimo vértice

Sea G una gráfica no dirigida con topología de anillo o malla. En particular, para esta investigación, se usaron mallas de 50×50; y anillos de 1500 vértices. Los atributos requeridos en el i-ésimo vértice de la red, son:

1. **numEnlacesDinámicos**: Número de aristas dinámicas que i tiene.
2. **máximoConexionesPermitidas**: Número máximo de conexiones que i puede aceptar.
3. **num_paquetes**: Número de paquetes exploradores que i enviará en fase de exploración.
4. **encaminamiento**: Algoritmo de encaminamiento que i usará en los experimentos.
5. **regla**: Regla de conexión-reconexión de aristas que i usará en los experimentos.
6. **contadorCiclos**: Contador que i lleva, correspondiente a su número de ciclos ejecutados.
7. **frecNodo**: Umbral de frecuencia de visita a vértices en un ciclo en particular. Su valor está dado por $num\_paquetes \cdot \frac{1}{2}$
8. **paquetes**: Lista de paquetes exploradores emitidos por i.
9. **paquetesRegreso**: Contador de paquetes exploradores que han regresado a i, después

de realizar su fase de exploración.

10. **enlacesDinámicos**: Lista de aristas dinámicas con las que i cuenta.

11. **vecinosConectadosDinámicos**: Lista de identificadores de los vértices con los que i está conectado a través de sus aristas dinámicas.

12. **f_n**: Tabla clave-valor que contiene la frecuencia de visita a los vértices que no son vecinos de i en la última fase de exploración ejecutada. Las llaves de la tabla, son los identificadores de los vértices, mientras que los valores, son las frecuencias de visita.

13. **f_e**: Tabla clave-valor que contiene la frecuencia de uso de cada una de las aristas dinámicas con las que i cuenta. Las llaves de la tabla, son los identificadores de las aristas dinámicas, mientras que los valores, son las frecuencias de uso.

14. **listaNodosSolicitados**: Lista de identificadores de vértices que i solicita para realizar una conexión de sus aristas dinámicas, en un ciclo en particular.

15. **númeroSolicitudes**: Número de solicitudes de conexión-reconexión de aristas dinámicas hechas por i, en un ciclo en particular.

16. **conexionesAceptadas**: Número de conexiones que i ha aceptado a lo largo de la ejecución del experimento.

17. **neighborsPendientes**: Lista de identificadores de vértices que i agregará a su vecindario, en un ciclo en particular. Esto se da en los vértices destino, es decir, los que aceptan las solicitudes de conexión-reconexión de aristas dinámicas.

18. **neighborsPendientesEliminación**: Lista de identificadores de vértices que i eliminará de su vecindario, en un ciclo en particular.

19. **solicitudesPendientes**: Lista de solicitudes de conexión-reconexión de aristas diná-

micas que i realizó, y que resultaron favorables para el. En la lista, se almacenan los paquetes de las solicitudes, debido a que contienen la información necesaria para realizar la conexión-reconexión.

20. **diámetroGrafo**: Diámetro de la gráfica subyacente en la red, en un ciclo en particular.

21. **neighbors**: Representa el conjunto de identificadores de los vértices adyacentes a i.

Más aún, al hacer uso del algoritmo PIF en las tres fases, se requieren los siguientes atributos en el i-ésimo vértice:

1. **visitado**: Es una variable booleana, que inicialmente se encuentra establecida en $falso$. Esta variable cambia su valor a *verdadero* si ya recibió algún mensaje propio del algoritmo PIF correspondiente a la fase en la que se encuentre el experimento.

2. **N(k)**: Es una tabla clave-valor, donde las claves son los identificadores de los vértices vecinos de i, y los valores asociados a las claves están todos, inicialmente, establecidos en 0, cuando i recibe $PIF-EXPLORACIÓN$, $PIF-NEGOCIACIÓN$ o $PIF-CONEXIÓN$ (según corresponda), desde k, entonces $N[k]=1$

3. **padre**: Indica el vértice desde el que i recibe $PIF-EXPLORACIÓN$, $PIF-NEGOCIACIÓN$ o $PIF-CONEXIÓN$ (según corresponda), por primera vez. Inicialmente está establecido en nulo.

3.2.2.3. Mensajes intercambiados

Los mensajes intercambiados por el algoritmo, son:

Sea $i \in V$, el vértice emisor de un mensaje, y $j \in V$, su respectivo destino:

1. ***PIF-EXPLORACIÓN***: Mensaje enviado y recibido por todos los vértices de G, el cual, permite ejecutar un algoritmo PIF que ayuda a completar la fase de exploración,

induciendo la sincronización, por primera vez, en G.

2. ***PACKAGE***: Mensaje que contiene el paquete explorador, el cual, viajará de i a j, guiado por un algoritmo de encaminamiento y guardará la trayectoria por donde este viaja en el paquete explorador.

3. ***ACK***: Mensaje que reconoce que un mensaje *PACKAGE* llegó a su destino. Este mensaje, utiliza la ruta guardada en el paquete explorador, pero esta vez, en sentido inverso para viajar del destino j al origen i del paquete.

4. ***PIF-NEGOCIACIÓN***: Mensaje enviado y recibido por todos los vértices de G, el cual, permite ejecutar un algoritmo PIF que ayuda a completar la fase de negociación, induciendo la sincronización, por segunda vez, en G.

5. ***SOLICITUD-CONEXIÓN***: Mensaje enviado por i, el cual, representa una solicitud de conexión-reconexión de alguna de sus aristas dinámicas hacia j.

6. ***ACEPTO-CONEXIÓN***: Mensaje que, una vez que j recibe un mensaje *SOLICITUD-CONEXIÓN* desde i, le permite aceptarla, informándole de este hecho a i.

7. ***DESCONEXIÓN***: Mensaje que, una vez que i recibe un mensaje *ACEPTO-CONEXIÓN*, le permite mandar una solicitud de desconexión al vértice $z \in V$ con el que, actualmente, está conectado a través de la arista dinámica implicada en el proceso de conexión-reconexión.

8. ***DESCONEXIÓN-RECIBIDA***: Mensaje que, una vez que z recibe un mensaje *DESCONEXIÓN* desde i, le permite eliminarlo de su lista de vecinos.

9. ***RECHAZO-CONEXIÓN***: Mensaje que, una vez que j recibe un mensaje *SOLICITUD-CONEXIÓN* desde i, le permite rechazarla, informándole de este hecho a i.

10. ***PIF-CONEXIÓN***: Mensaje enviado y recibido por todos los vértices de G, el cual, permite ejecutar un algoritmo PIF que ayuda a completar la fase de conexión-reconexión, induciendo la sincronización, por tercera vez, en G y permitiendo iniciar, o no, un nuevo ciclo de experimentación, según sea el caso.

La lógica de cada fase, es detallada a continuación:

3.2.2.4. Fase de exploración

En esta fase de los experimentos, cada vértice de la red recabará información parcial de la topología subyacente, esto lo logrará a través del envío y recepción de mensajes, mediante una modificación al algoritmo PIF, la cual es llamada *PIF-EXPLORACIÓN*. La información que transportan los mensajes del algoritmo, se representa como un *paquete explorador*, el cual, es definido como sigue:

3.2.2.4.1. El paquete explorador

Los paquetes exploradores, representan la información que transportan los mensajes del algoritmo PIF, dichos paquetes, se identifican con un número $id\_paquete \in \mathbb{N}$, y permiten a los vértices de la red, almacenar la ruta por la cual viajan, lo cual, los pone en posibilidad de tomar decisiones de conexión y reconexión de sus aristas dinámicas.

Los paquetes exploradores, almacenan la siguiente información:

1. Identificador del vértice que emite el paquete explorador ($idVert$).

2. Identificador del paquete explorador ($idPaq$).

3. Identificador de la arista dinámica que está haciendo una solicitud de conexión-reconexión ($idEnlace$).

4. Ruta por donde viaja el paquete explorador (*ruta*). En cada paso del encaminamiento, se agregará o eliminará de ella, el identificador del vértice por el que está viajando.

5. Ruta auxiliar, la cual, es una copia de la ruta por donde viaja el paquete, pero esta permanecerá intacta y permitirá regresar el paquete explorador al vértice que lo emitió al usarla en sentido contrario (*rutaAuxiliar*).

6. Distancia máxima en saltos a la que viajará el paquete explorador (*dist*), (para uso con el algoritmo Random-Walk).

7. Identificador del vértice destino del paquete explorador (*destino*), (para uso con los algoritmos Compass-Routing y Shortest-Path).

8. Diámetro de la topología subyacente en la red (*diámetro*), actualizado en cada ciclo de ejecución de los experimentos, (para uso con el algoritmo Random-Walk).

En esta fase de los experimentos, cada vértice $v \in V$, envía 20 mensajes, cada uno con un paquete explorador, a diferentes destinos, los cuales, son seleccionados con base en una función de distribución de probabilidad uniforme ($\sim U[1, n]$). No se permite que el destino sea idéntico al origen; ni que esté contenido en su vecindario. Los paquetes, viajan apoyándose en alguno de los siguientes tres algoritmos de encaminamiento:

1. Random-Walk

2. Compass-Routing

3. Shortest-Path

Los cuales, son detallados a continuación:

3.2.2.4.2. Algoritmo de encaminamiento: "Random-Walk"

En esta investigación, se denomina a este algoritmo como una estrategia de *caminata a ciegas*, debido a que un vértice $v \in V$, simplemente selecciona al siguiente vértice $u \in V$ al que desea enviar un mensaje, con opciones determinadas, solamente, por su conjunto de vecinos $N(v)$, seleccionando a uno de ellos, con base en una función de distribución de probabilidad uniforme. El algoritmo Random-Walk, considera una caminata aleatoria sin retorno, es decir, una vez que v manda un mensaje a u, u tiene estrictamente prohibido regresarle el mensaje a v. El mensaje que viaja, contiene el paquete explorador y este, a su vez, almacena la ruta por donde viaja el mensaje, dotando de cierta "memoria" a cada uno de los vértices por donde viaja, permitiéndoles tomar decisiones, descartando los vértices por los que ya pasó dicho paquete. Cada paquete explorador, generado en cada vértice, viaja hasta una distancia $dist \in \mathbb{N}$ en saltos, la cual, se determina, en cada ciclo de los experimentos, como un número $\sim U[2, D_G]$. Si, eventualmente, el algoritmo detecta que el paquete ya no tiene hacia donde dirigirse y aún no llega a su distancia máxima establecida, regresará al vértice que lo emitió, usando la ruta almacenada en el paquete, pero esta vez, en sentido contrario. La lógica de programación de Random-Walk, se muestra en el Algoritmo 6.

3.2.2.4.3. Algoritmo de encaminamiento: "Compass-Routing"

Suponga que queremos viajar de un vértice inicial $s \in V$ a un vértice destino $t \in V$, y que toda la información disponible para nosotros, en cualquier punto del tiempo, son las coordenadas de nuestro destino; las de nuestra posición actual, y las direcciones de las aristas incidentes con el vértice en que nos encontramos. Iniciando en s, se elegirá y recorrerá, de manera recursiva, la arista de la gráfica incidente a nuestra posición actual, que forme el menor ángulo, con respecto al segmento de recta que conecta al vértice donde estamos parados con t. [32]. En este algoritmo, se hace uso de las coordenadas (x, y) de los vértices empotrados

Algoritmo 6 Encaminamiento Random-Walk en el vértice i

Requiere:

La distancia $dist \sim U[2, D_G]$, a la que viajará el paquete explorador (contenida en el)

La $ruta$ por la que ha viajado el mensaje (contenida en el paquete explorador)

1: $N(i)\_copia \leftarrow neighbors \cup vecinosConectadosDinámicos$
2: **Si** $dist > 1$ **Entonces**
3: **Para** $x \in ruta$ **Hacer**
4: **Si** $x \in N(i)\_copia$ **Entonces**
5: $N(i)\_copia = N(i)\_copia \setminus \{x\}$
6: **Termina Si**
7: **Termina Para**
8: **Si** $N(i)\_copia = \emptyset$ **Entonces**
9: Agrega el identificador de i a $ruta$
10: $rutaAuxiliar \leftarrow ruta$
11: Elimina la última entrada contenida en $ruta$ {en este caso es el identificador de i}
12: $ruta \leftarrow ruta.reverse()$ {el método $reverse()$, regresa la ruta invertida}
13: $siguiente \leftarrow ruta[0]$
14: $ruta \leftarrow ruta \setminus \{ruta[0]\}$
15: Envía mensaje ACK a $siguiente$
16: **Sino**
17: Agrega el identificador de i a $ruta$
 {$random.choice(x)$, devuelve un elemento al azar contenido en x}
18: $siguiente \leftarrow random.choice(N(i)\_copia)$
19: $dist \leftarrow dist - 1$
20: Envía mensaje $PACKAGE$ a $siguiente$.
21: **Termina Si**
22: **Sino**
23: Agrega el identificador de i a $ruta$

24: $rutaAuxiliar \leftarrow ruta$
25: Elimina la última entrada contenida en $ruta$ {en este caso es el identificador de i}
26: $ruta \leftarrow ruta.reverse()$ {el método $reverse()$, regresa la ruta invertida}
27: $siguiente \leftarrow ruta[0]$
28: $ruta \leftarrow ruta \setminus \{ruta[0]\}$
29: Envía mensaje ACK a $siguiente$
30: **Termina Si**

en el espacio Euclidiano, buscando, en cada paso, optimizar dicho espacio, por lo anterior, en esta investigación, se denomina a este algoritmo como una estrategia de *caminata semi-informada*, ya que el algoritmo no encuentra la ruta óptima en términos de saltos, como si lo hace un algoritmo que más adelante se explicará, denominado Shortest-Path. En particular, para el desarrollo de esta investigación, se trabajó con gráficas cuyas topologías subyacentes son de malla y anillo, las cuales, como se mostró anteriormente, se pueden, perfectamente, empotrar en $\mathbb{R}^2$. Para poder calcular el ángulo α que se forma entre las rectas, primero, se debe realizar una traslación de ejes en el plano, la cual, no es otra cosa que el desplazamiento de los ejes de un sistema de coordenadas rectangulares, de tal manera que el nuevo origen sea el punto $O'(h, k)$. Las coordenadas (x', y') con el nuevo origen se pueden calcular mediante las Ecuaciones 3.5 y 3.6, tal y como se muestra en la Figura 3.8.

$$x' = x - h \tag{3.5}$$

$$y' = y - k \tag{3.6}$$

En particular, se puede asumir que las coordenadas de los vértices implicados están dadas por las tuplas:

1. $s(s_x, s_y)$

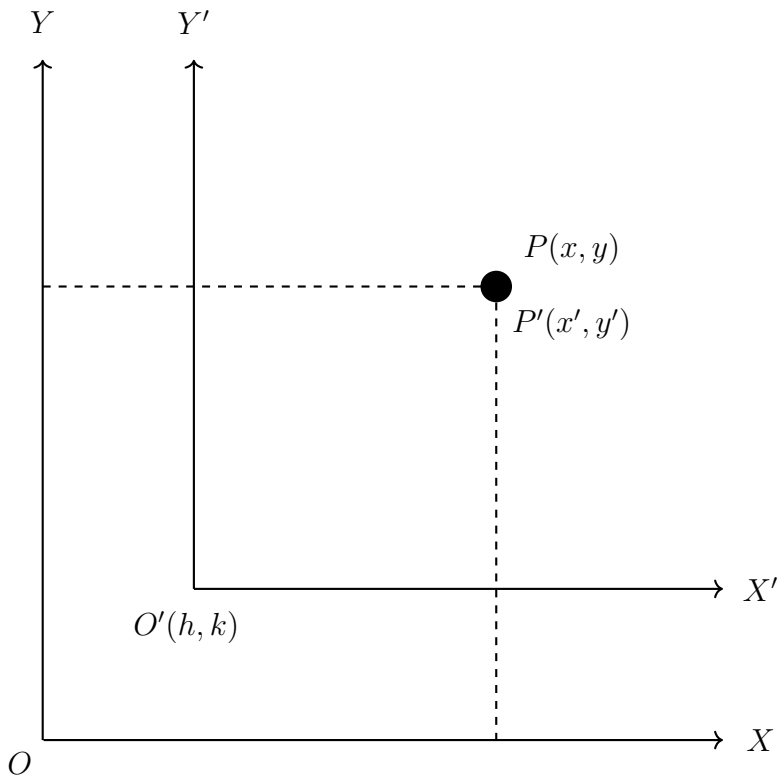


Figura 3.8: Traslación de ejes a un nuevo origen $O'(h, k)$

2. $t(t_x, t_y)$

3. $vecino(n_x, n_y)$

Se deben trasladar esas tres tuplas de coordenadas al sistema de coordenadas rectangulares con origen $s(s_x, s_y)$. Aplicando las Ecuaciones 3.5 y 3.6, se obtiene lo siguiente:

$$x_1 = t_x - s_x \tag{3.7}$$

$$x_2 = t_y - s_y \tag{3.8}$$

$$x'_1 = n_x - s_x \tag{3.9}$$

$$x'_2 = n_y - s_y \tag{3.10}$$

Más aún, a la pareja de vectores obtenida $\vec{u} = (x_1, x_2)$ y $\vec{v} = (x'_1, x'_2)$, se le puede aplicar la Ecuación 3.11, la cual, representa el producto escalar o producto punto entre vectores, dicha ecuación, permite encontrar el ángulo α entre s, sus aristas incidentes y el segmento de recta formado entre s y t.

$$\alpha = \arccos \frac{u[1]v[1] + u[2]v[2]}{\sqrt{u[1]^2 + u[2]^2}\sqrt{v[1]^2 + v[2]^2}} \tag{3.11}$$

Este procedimiento, se aplica a cada una de las aristas incidentes a s para determinar el ángulo menor y, por lo tanto, la arista que se recorrerá. El algoritmo Compass-Routing, cuenta con una deficiencia, la cual, es que no siempre encuentra una ruta para llegar de s a t, debido a que puede generar ciclos al momento de computar el encaminamiento, lo cual, se soluciona con el mecanismo usado en Random-Walk, al prohibir que s envié un mensaje a t y que t decida regresarlo a s. La lógica de programación de Compass-Routing, se muestra en el Algoritmo 7.

3.2.2.4.4. Algoritmo de encaminamiento: "Shortest-Path"

Para mostrar el desarrollo de este algoritmo, es conveniente recordar que una gráfica ponderada, es una gráfica en la que se asignan valores a las aristas y que la longitud de una trayectoria en una gráfica ponderada, es la suma de los pesos de las aristas en la trayectoria. Sean $i, j \in V$, $w(i, j)$ el peso de la arista $e = (i, j)$. En las gráficas ponderadas, con frecuencia

Algoritmo 7 Encaminamiento Compass-Routing en el vértice i

Requiere:

El identificador del vértice destino $destino \sim U[1, n], destino \neq s$ (contenido en el paquete explorador)

La $ruta$ por la que ha viajado el mensaje (contenida en el paquete explorador)

1: $N(i)\_copia \leftarrow neighbors \cup vecinosConectadosDinámicos$
2: **Si** $i \neq destino$ **Entonces**
3: **Para** $x \in ruta$ **Hacer**
4: **Si** $x \in N(i)\_copia$ **Entonces**
5: $N(i)\_copia = N(i)\_copia \setminus \{x\}$
6: **Termina Si**
7: **Termina Para**
8: **Si** $N(i)\_copia = \emptyset$ **Entonces**
9: Agrega el identificador de i a $ruta$
10: $rutaAuxiliar \leftarrow ruta$
11: Elimina la última entrada contenida en $ruta$ {en este caso es el identificador de i}
12: $ruta \leftarrow ruta.reverse()$ {el método $reverse()$, regresa la ruta invertida}
13: $siguiente \leftarrow ruta[0]$
14: $ruta \leftarrow ruta \setminus \{ruta[0]\}$
15: Envía mensaje ACK a $siguiente$
16: **Sino**
17: Agrega el identificador de i a $ruta$
18: $ángulomáximo \leftarrow 360$
19: **Si** $destino \in N(i)\_copia$ **Entonces**
20: $siguiente \leftarrow destino$
21: **Sino**
22: **Para** $vecino \in N(i)\_copia$ **Hacer**
23: $ángulo \leftarrow \arccos \frac{u[1]v[1]+u[2]v[2]}{\sqrt{u[1]^2+u[2]^2}\sqrt{v[1]^2+v[2]^2}}$ {se hace uso de la Ecuación 3.11}

```
            Si ángulo ≤ ángulomáximo Entonces
               siguiente ← vecino
               ángulomáximo ← ángulo
            Termina Si
         Termina Para
      Termina Si
      Envía mensaje PACKAGE a siguiente
   Termina Si
Sino
   Agrega el identificador de i a ruta
   rutaAuxiliar ← ruta
   Elimina la última entrada contenida en ruta {en este caso es el identificador de i}
   ruta ← ruta.reverse() {el método reverse(), regresa la ruta invertida}
   siguiente ← ruta[0]
   ruta ← ruta \ {ruta[0]}
   Envía mensaje ACK a siguiente
Termina Si
```

se desea encontrar la *ruta más corta*, es decir, una trayectoria que tiene la longitud mínima, entre dos vértices dados. El Algoritmo 8, ideado por Edsger W. Dijkstra, resuelve con eficiencia este problema al tener un tiempo de corrida, en el peor caso, de $\Theta(n^2)$. El algoritmo de Dijkstra, trabaja con gráficas G ponderadas. Se supone que los pesos son números positivos y que se quiere encontrar la ruta más corta del vértice $a \in V$ al vértice $z \in V$. El algoritmo de Dijkstra implica asignar etiquetas a los vértices. Sea $L(v)$ la etiqueta del vértice v. En cualquier punto, algunos vértices tienen etiquetas temporales y el resto son permanentes. Sea T el conjunto de vértices que tienen etiquetas temporales. Si $L(v)$ es la etiqueta permanente del vértice v, entonces $L(v)$ es la longitud de una ruta más corta de a a v. Al inicio, todos los vértices tienen etiquetas temporales. Cada iteración del algoritmo cambia el estado de una etiqueta de temporal a permanente; entonces el algoritmo puede terminar cuando z recibe

una etiqueta permanente. En este punto $L(v)$ da la longitud de la ruta más corta de a a z [7].

El algoritmo de Dijkstra, encuentra la longitud de una ruta más corta del vértice a al vértice z en una gráfica ponderada conexa. El peso de la arista $e = (i, j)$ es $w(i, j) > 0$, y la etiqueta del vértice x es $L(x)$. Al terminar, $L(z)$ es la longitud de la ruta más corta de a a z. En estricto sentido, el algoritmo encuentra la ruta de menor peso, pero si el peso representa distancias entonces el resultado coincide con la ruta más corta.

Algoritmo 8 Encaminamiento inicial Shortest-Path (Dijkstra) en el vértice a

Requiere:

Una gráfica G conexa y ponderada, en la que todos los pesos son positivos

Identificador del vértice origen $idVert \in V$, el cual, emite el paquete explorador

Identificador del vértice destino $destino \in V$ del paquete explorador (contenida en el)

1: $L(idVert) \leftarrow 0$
2: **Para** todo vértice $x \neq idVert$ **Hacer**
3: $L(x) \leftarrow \infty$
4: **Termina Para**
5: $T \leftarrow$ conjunto de todos los vértices cuyas distancias más cortas desde $idVert$ no se han encontrado
6: **Mientras** $destino \in T$ **Hacer**
7: Seleccionar $v \in T$ con $L(v)$ mínimo
8: $T \leftarrow T \setminus \{v\}$
9: **Para** $x \in T$ adyacente a v **Hacer**
10: $L(x) \leftarrow min\{L(x), L(v) + w(v, x)\}$
11: Marcar a x con la tupla $(L(x), v)$
12: **Termina Para**
13: **Termina Mientras**
14: Comenzando en *destino*, nos movemos hacia atrás por las etiquetas para encontrar la *ruta* más corta

El Algoritmo 8, se ejecuta siempre que a emita un paquete explorador por primera vez, debido a que computa la *ruta* más corta entre a y z, almacenándola en el paquete explorador y encaminando el paquete hacia el segundo vértice contenido en *ruta* (se va hacia el segundo vértice, porque el primero es el identificador del vértice que emite el paquete). No obstante, cuando el paquete explorador no es emitido por primera vez, sino se requiere encaminarlo en el i-ésimo vértice de la *ruta* computada por el Algoritmo 8, se ejecuta el Algoritmo 9.

Algoritmo 9 Encaminamiento Shortest-Path (intermedio) en el vértice i

Requiere:

El identificador del vértice i que realizará el encaminamiento

La *ruta* generada por primera vez con el Algoritmo 8 (contenida en el paquete explorador).

El identificador del vértice destino *destino* (contenido en el paquete explorador)

1: **Si** $i \neq destino$ **Entonces**
2: $siguiente \leftarrow ruta[0]$
3: $ruta \leftarrow ruta \setminus \{ruta[0]\}$
4: Envía mensaje $PACKAGE$ a $siguiente$
5: **Sino**
6: $ruta \leftarrow rutaAuxiliar$
7: Elimina la última entrada contenida en $ruta$ {en este caso es el identificador de i}
8: $ruta \leftarrow ruta.reverse()$ {el método $reverse()$, regresa la ruta invertida}
9: $siguiente \leftarrow ruta[0]$
10: $ruta \leftarrow ruta \setminus \{ruta[0]\}$
11: Envía mensaje ACK a $siguiente$
12: **Termina Si**

Una vez que cada paquete explorador llega a su destino correspondiente, este responde con un mensaje ACK al vértice que lo emitió, usando la ruta guardada en el paquete explorador, pero esta vez, en sentido inverso.

En particular, en esta fase, se hace uso de los siguientes mensajes:

1. *PIF-EXPLORACIÓN*
2. *PACKAGE*
3. *ACK*

Los cuales, son detallados a continuación:

3.2.2.4.5. El mensaje $PIF - EXPLORACIÓN$

El mensaje $PIF - EXPLORACIÓN$, lleva la lógica de control del algoritmo PIF subyacente en la fase de exploración. La lógica que se agrega en esta fase, es que, después de que un vértice i recibe el mensaje $PIF - EXPLORACIÓN$ por primera vez, lo reexpide a todos sus vecinos, excepto a su padre, e inmediatamente comienza su fase de exploración, encaminando un paquete explorador a un vértice seleccionado con base en una función de distribución de probabilidad uniforme (para el caso de los algoritmos Compass-Routing y Shortest-Path), o a una distancia *dist* en saltos (para el caso del algoritmo Random-Walk) a través de un mensaje $PACKAGE$. Resulta imprescindible considerar a los vecinos de i que están conectados a él a través de aristas fijas y también a los que están conectados a través de aristas dinámicas para poder seleccionar el vértice al que se encaminará el paquete. En el ciclo uno de experimentación, todos los vértices de G, tienen sus aristas dinámicas desconectadas, pero esto, eventualmente, cambiará. El paquete generado por i, es agregado a su lista de paquetes exploradores, e i queda en espera de recibir su mensaje ACK correspondiente al $PACKAGE$ enviado. El objetivo, es informar a todos los vértices en la red que deben ejecutar su fase de exploración. El algoritmo, lo inicia un vértice $c \in V$, al cual, se le denomina "coordinador", ya que será el responsable de dar inicio y fin a las tres fases que completan un ciclo de experimentación. Como se mencionó anteriormente, el algoritmo PIF induce un

árbol de pesos mínimos, el cual, en este caso, tiene raíz en el coordinador y se asegura que la propagación de información es lo más rápida posible. En particular, el padre de un vértice x en el nivel $l+1$ del árbol, será el que emita el mensaje que llegue más rápido desde los vértices del nivel l que se conectan con x. El algoritmo PIF-EXPLORACIÓN, termina cuando todos los vértices de la red han recibido sus 20 mensajes ACK y sus $|N(i)|$ mensajes $PIF-EXPLORACIÓN$ desde todos sus vecinos (conectados a través de aristas fijas y dinámicas). El coordinador, será el encargado de verificar lo anterior mediante la propiedad de terminación global, y dar inicio a la fase de negociación. El Algoritmo 10, muestra la lógica de programación del Algoritmo "Atendiendo mensaje $PIF-EXPLORACIÓN$ en el vértice i".

3.2.2.4.6. El mensaje $PACKAGE$

Un vértice, al recibir un mensaje $PACKAGE$, simplemente ejecutará uno de los Algoritmos 6,7 o 9, en función de su atributo *encaminamiento*, el cual, fue establecido al inicio del experimento en cada vértice de la red. El Algoritmo 11, muestra la lógica de programación del Algoritmo "Atendiendo mensaje $PACKAGE$ en el vértice i".

3.2.2.4.7. El mensaje ACK

El mensaje ACK, representa una confirmación de que un mensaje $PACKAGE$ llegó a su destino. Cada vez que un vértice i recibe un mensaje ACK, verifica que dicho mensaje sea para él, si ese fuera el caso, significa que el paquete explorador que le corresponde fue entregado con éxito. Cuando esto sucede i realiza las siguientes tareas:

1. Su atributo *paquetesRegreso* (el cual, al inicio de la ejecución del experimento fue establecido en 0), se incrementa en una unidad

Algoritmo 10 Atendiendo mensaje $PIF - EXPLORACIÓN$ en el vértice i

Requiere:

Una gráfica $G = (V, E)$ con topología de malla o anillo.

Una opción $encaminamiento \in \{Shortest-Path, Random-Walk, Compass-Routing\}$

Si se recibe el mensaje $PIF - EXPLORACIÓN$ desde el vértice j **Entonces**
  $N[j] \leftarrow 1$
  Si $visitado = falso$ **Entonces**
    $diámetroGrafo \leftarrow diámetro$ {$diámetro$, está contenido en el paquete explorador}
    $padre \leftarrow j$
    $visitado \leftarrow verdadero$
    $conjuntoVecinos \leftarrow neighbors \cup vecinosConectadosDinámicos$
    Si $vecinos \setminus \{padre\} \neq \emptyset$ **Entonces**
      Para cada $k \in conjuntoVecinos \setminus \{padre\}$ **Hacer**
        Envía $PIF - EXPLORACIÓN$ a k
      Termina Para
    Termina Si
    Genera un paquete explorador con sus atributos en función de *encaminamiento*
    Si $encaminamiento \neq Shortest - Path$ **Entonces**
      Agrega el identificador de i a *ruta*
    Sino
      Genera la *ruta* más corta de i a *destino* mediante el Algoritmo 8
      $rutaAuxiliar \leftarrow ruta$
      $ruta \leftarrow ruta \setminus \{ruta[0]\}$ {el identificador de i}
      $siguiente \leftarrow ruta[0]$
      $ruta \leftarrow ruta \setminus \{ruta[0]\}$
    Termina Si
    Agrega el paquete generado a la lista de paquetes de i

24: **Si** $encaminamiento = Shortest - Path$ **Entonces**
25: Envía mensaje $PACKAGE$ a *siguiente*
26: **Sino**
27: Encamina y envía mensaje $PACKAGE$ en función del atributo *encaminamiento*
28: **Termina Si**
29: **Termina Si**
30: **Si** $N[k] = 1 \ \forall \ k \in vecinos$ y $paquetesRegreso = num\_paquetes$ **Entonces**
31: **Si** $padre \neq i$ **Entonces**
32: Envía $PIF - EXPLORACIÓN$ a *padre*
33: Reinicia los atributos del algoritmo PIF de i a sus valores originales
34: **Sino**
35: Reinicia los atributos del algoritmo PIF de i a sus valores originales
36: Envía $PIF - NEGOCIACIÓN$ a i {se inicia la fase de negociación}
37: **Termina Si**
38: **Termina Si**
39: **Termina Si**

Algoritmo 11 Atendiendo mensaje $PACKAGE$ en el vértice i

Requiere:

Una opción $encaminamiento \in \{Shortest-Path, Random-Walk, Compass-Routing\}$

1: **Si** se recibe el mensaje $PACKAGE$ desde el vértice j **Entonces**
2: Ejecuta el algoritmo de encaminamiento seleccionado para i de acuerdo a *encaminamiento*
3: **Termina Si**

2. Actualiza sus tablas clave-valor f_n y f_e, destacando que en la tabla f_n, no se agregan los vértices contenidos en el vecindario de i, aunque aparezcan en la ruta almacenada en el paquete explorador. Si se agregaran dichos vértices a f_n, se daría oportunidad a que dos o más aristas de i, se conecten al mismo vértice.

3. Verifica si $paquetesRegreso < num\_paquetes$, es decir, verifica si ya recibió los 20 mensajes ACK correspondientes a sus mensajes $PACKAGE$ enviados. En caso de que aún no haya recibido sus 20 ACK, entonces genera un nuevo paquete explorador y lo encamina en función de su atributo *encaminamiento*. Si ya recibió todos sus ACK, entonces, verifica si ya recibió el mensaje $PIF - EXPLORACIÓN$ desde todos sus vecinos (conectados a través de aristas fijas y dinámicas). Si esto ocurre y no es el coordinador, envía un mensaje $PIF - EXPLORACIÓN$ a su padre y reinicia sus atributos que permiten controlar su algoritmo PIF, caso contrario, si el es coordinador y ya recibió sus 20 mensajes ACK, se envía un mensaje $PIF - NEGOCIACIÓN$ a el mismo, indicando que la fase de exploración terminó en toda la red y en ese preciso instante, inicia la fase de negociación.

Caso contrario, simplemente reenvía el mensaje al siguiente vértice contenido en $ruta$ (la cual, ya fue invertida desde que se generó por primera vez el mensaje ACK). El Algoritmo 10, muestra la lógica de programación del Algoritmo "Atendiendo mensaje ACK en el vértice i".

3.2.2.5. Fase de negociación

3.2.2.5.1. Reglas locales de conexión-reconexión

En esta fase, cada vértice de la red, negociará la conexión de sus dos aristas dinámicas (si el experimento se encuentra en el ciclo inicial), o la reconexión de alguna de ellas (si ya fueron establecidas todas sus conexiones mediante aristas dinámicas), con base en tres distintas reglas locales de reconexión, las cuales, se definen a continuación:

Algoritmo 12 Atendiendo mensaje ACK en el vértice i

Requiere:

Una opción $encaminamiento \in \{Shortest-Path, Random-Walk, Compass-Routing\}$

La $ruta$ por donde debe viajar el paquete explorador (contenida en el mismo paquete explorador)

Si se recibe el mensaje ACK desde el vértice j **Entonces**

{el método $len(x)$, regresa el número de elementos contenidos en la lista x}

  Si $len(ruta) > 0$ **Entonces**

    $siguiente \leftarrow ruta[0]$

    $ruta \leftarrow ruta \setminus \{ruta[0]\}$

    Envía mensaje ACK a $siguiente$

  Sino

    $paquetesRegreso \leftarrow paquetesRegreso + 1$

    Actualiza la tabla clave-valor f_e, de acuerdo a la arista dinámica utilizada al enviar y recibir el mensaje ACK actual

    Actualiza la tabla clave-valor f_n, de acuerdo al número de veces que se visitaron los vértices explorados (contenidos en $rutaAuxiliar$)

    Si $paquetesRegreso < num\_paquetes$ **Entonces**

      Genera un paquete explorador con sus atributos en función de $encaminamiento$

      Si $encaminamiento \neq Shortest - Path$ **Entonces**

        Agrega el identificador de i a $ruta$

      Sino

        Genera la $ruta$ más corta de i a $destino$ mediante el Algoritmo 8

        $rutaAuxiliar \leftarrow ruta$

        $ruta \leftarrow ruta \setminus \{ruta[0]\}$ {el identificador de i}

        $siguiente \leftarrow ruta[0]$

19: $ruta \leftarrow ruta \setminus \{ruta[0]\}$
20: **Termina Si**
21: Agrega el paquete generado a la lista de paquetes de i
22: **Si** $encaminamiento = Shortest - Path$ **Entonces**
23: Envía mensaje $PACKAGE$ a *siguiente*
24: **Sino**
25: Encamina y envía mensaje $PACKAGE$ en función del atributo *encaminamiento*
26: **Termina Si**
27: **Termina Si**
28: **Si** $N[k] = 1 \ \forall \ k \in vecinos$ y $paquetesRegreso = num\_paquetes$ **Entonces**
29: **Si** $padre \neq i$ **Entonces**
30: Envía $PIF - EXPLORACIÓN$ a *padre*
31: Reinicia los atributos del algoritmo PIF de i a sus valores originales
32: **Sino**
33: Reinicia los atributos del algoritmo PIF de i a sus valores originales
34: Envía $PIF - NEGOCIACIÓN$ a i {se inicia la fase de negociación}
35: **Termina Si**
36: **Termina Si**
37: **Termina Si**
38: **Termina Si**

Sea $v \in V$:

1. ***R1***: v, elige como mejor candidato a conexión-reconexión, a aquel vértice por el que atravesaron la mayor cantidad de paquetes exploradores en la fase de exploración, apoyándose de su tabla clave-valor f_n.

2. ***R2***: v, elige como candidato a conexión, al primer vértice, a distancia 2, del que tiene

conocimiento en su tabla clave-valor f_n.

3. ***R3***: v, elige a un vértice candidato, de forma pseudo-aleatoria, basado en una función de distribución de probabilidad proporcional a las frecuencias de visita registradas en su tabla clave-valor f_n.

3.2.2.5.2. Restricciones para realizar la conexión-reconexión

Toda vez que v elige a un vértice $p \in V$ como candidato a conexión-reconexión de alguna de sus aristas dinámicas a través de una de las tres reglas presentadas previamente, debe verificar que se cumpla un conjunto de restricciones para poder llevar a cabo el procedimiento de conexión-reconexión.

La primer restricción a verificar, consiste en revisar si la longitud de la arista dinámica de v implicada en el proceso de conexión-reconexión, es suficiente para alcanzar al vértice p. Es conveniente recordar que las topologías de red, están embebidas o empotradas en un espacio Euclidiano bidimensional ($\mathbb{R}^2$), lo cual, permite hacer uso de la Ecuación 3.12, la cual, entrega la distancia entre cualesquiera dos puntos $P_1(x_1, y_1)$, $P_2(x_2, y_2)$ de ($\mathbb{R}^2$).

$$d(P_1, P_2) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \tag{3.12}$$

Por ejemplo, suponga que el vértice v con coordenadas $P_1(0, 2)$, desea conectar una de sus aristas dinámicas al vértice p ubicado en $P_2(2, 3)$ de la malla mostrada en la Figura 3.3. Lo primero que se debe realizar, es calcular la distancia entre ambos vértices, entonces, sustituyendo los datos en la Ecuación 3.12, se obtiene:

$$d(P_1, P_2) = \sqrt{(0 - 2)^2 + (2 - 3)^2} = \sqrt{5} \tag{3.13}$$

Suponiendo que la longitud de las aristas dinámicas de v es $\frac{D}{2}$, se debe proceder a verificar si es posible llevar a cabo la conexión de la arista dinámica implicada en el proceso, como sigue:

Es necesario recordar que la longitud de las aristas dinámicas $\frac{D}{2}$, significa que ellas no pueden "estirarse" más allá de la longitud de la diagonal principal de la malla, dividido por dos. En este sentido, resulta indispensable calcular la medida de la diagonal principal de la malla mostrada en la Figura 3.3, como sigue:

Sean $P_3(0,0)$ y $P_4(5,5)$, entonces:

$$d(P_3, P_4) = \sqrt{(0-5)^2 + (0-5)^2} = \sqrt{50} = \sqrt{25}\sqrt{2} = 5\sqrt{2} \tag{3.14}$$

Ahora, dividiendo $d(P_3, P_4)$ por 2, se obtiene:

$$\frac{d(P_3, P_4)}{2} = \frac{5\sqrt{2}}{2} \tag{3.15}$$

Para finalizar, se debe verificar:

$$d(P_1, P_2) \leq \frac{d(P_3, P_4)}{2} \tag{3.16}$$

Sustituyendo los resultados de las Ecuaciones 3.13 y 3.15 en 3.16, se verifica $\sqrt{5} \leq \frac{5\sqrt{2}}{2}$, por lo tanto, v está en libertad de verificar la segunda restricción, la cual se enuncia a continuación:

La segunda restricción a verificar, aplica solo para vértices configurados para trabajar su proceso de conexión-reconexión de aristas dinámicas con base en la regla $R1$. Es decir, los vértices que trabajan con $R2$ y $R3$, no necesitan verificarla. Suponga que v desea verificar su

segunda restricción para conectar una de sus aristas dinámicas hacia p, entonces, se presentan tres casos posibles:

1. Si v usa $R1$, y ya verificó su primer restricción, entonces debe de revisar que el número de veces que sus paquetes exploradores pasaron por p sea mayor que $num\_paquetes \cdot \frac{1}{2}$.

2. Si v usa $R2$, y ya verificó su primer restricción, entonces está en total libertad de mandar una solicitud de conexión de una de sus aristas dinámicas a p.

3. Si v usa $R3$, y ya verificó su primer restricción, entonces está en total libertad de mandar una solicitud de conexión de una de sus aristas dinámicas a p.

En particular, suponga que v usa $R1$, y al terminar su fase de exploración, tiene su tabla clave-valor f_n configurada tal y como se muestra en la Tabla 3.1.

Sean 13 el identificador de v y 21; el de p, entonces:

Tabla 3.1: Tabla clave-valor f_n del vértice con identificador 13 al terminar su fase de exploración

Identificador	7	19	8	14	15	21
Frecuencia	2	3	1	2	1	11

Como se puede observar, los paquetes exploradores de v pasaron 11 veces por p, como en los experimentos se asume $num\_paquetes = 20$, se puede verificar que $11 > (20 \cdot \frac{1}{2})$, lo que pone en total libertad a v de mandar una solicitud de conexión-reconexión de alguna de sus aristas dinámicas hacia p. Caso contrario, aunque v haya cumplido cabalmente con la restricción 1, no será capaz de mandar dicha solicitud.

En particular, en esta fase, se hace uso de los siguientes mensajes:

1. *PIF-NEGOCIACIÓN*
2. *SOLICITUD-CONEXIÓN*
3. *ACEPTO-CONEXIÓN*
4. *DESCONEXIÓN*
5. *DESCONEXIÓN-RECIBIDA*
6. *RECHAZO-CONEXIÓN*

Los cuales, son detallados a continuación:

3.2.2.5.3. El mensaje $PIF - NEGOCIACIÓN$

El mensaje $PIF - NEGOCIACIÓN$, lleva la lógica de control del algoritmo PIF subyacente en la fase de negociación. La lógica que se agrega en esta fase, es que, después de que un vértice i recibe el mensaje $PIF - NEGOCIACIÓN$ por primera vez, lo reexpide a todos sus vecinos, excepto a su padre, e inmediatamente comienza su fase de negociación, verificando si cuenta con aristas dinámicas libres, si cuenta con al menos una, verifica si su atributo *regla* está establecido en $R1$, si este fuera el caso, procede a realizar un ordenamiento descendente de su tabla clave-valor f_n con base en las frecuencias de visita. Si su atributo *regla* no está establecido en $R1$, no es necesario que realice tal ordenamiento. Después, i revisa si existen candidatos potenciales en su tabla clave-valor f_n, para después verificar si su atributo *regla* está establecido en $R3$, si esto se cumple, selecciona a un candidato contenido en f_n con base en una probabilidad proporcional $\lambda(v)$ a la frecuencia de visita, haciendo uso de la Ecuación 3.17.

Sea $f(j)$ la frecuencia de visita al j-ésimo vértice:

$$\lambda(v) = \frac{f(v)}{\sum_{j \in f\_n} f(j)} \tag{3.17}$$

Caso contrario, el cálculo de la probabilidad no será necesario. Sea $v \in V$ el candidato seleccionado. Posteriormente, i verifica la primer restricción del conjunto mostrado en la Sección 3.2.2.5.2, si se cumple con dicha restricción, prepara un paquete explorador, al que agregará la ruta de menor longitud que le permite llegar a v (apoyándose de su lista de paquetes exploradores (*paquetes*) generada en su fase de exploración), agrega a v a su $listaNodosSolicitados$, incrementa en una unidad su atributo $númeroSolicitudes$, envía a v un mensaje $SOLICITUD-CONEXIÓN$ y lo elimina de su tabla clave-valor f_n para no considerarlo en caso de que i tenga otra arista dinámica libre.

En caso de que i no tenga ninguna arista dinámica libre, se verá en la necesidad de usar su tabla clave-valor f_e en búsqueda de la arista dinámica a_m que menos haya utilizado en su última fase de exploración, una vez que la encuentra, selecciona su candidato v a conexión-reconexión, verifica las restricciones 1 y 2 (según la regla de reconexión utilizada), busca la ruta de menor longitud que le permita llegar a v (contenida en su lista de paquetes exploradores *paquetes*), agrega a v a su $listaNodosSolicitados$, incrementa en una unidad su atributo $númeroSolicitudes$ y envía a v un mensaje $SOLICITUD-CONEXIÓN$. Como se pudo observar, se sigue el mismo procedimiento, a excepción de que se selecciona a la arista dinámica menos usada para una posterior reconexión de ella.

El algoritmo PIF-NEGOCIACIÓN, termina cuando todos los vértices de la red tienen su atributo $númeroSolicitudes = 0$ y han recibido sus $|N(i)|$ mensajes $PIF-NEGOCIACIÓN$ desde todos sus vecinos (conectados a través de aristas fijas y dinámicas). El coordinador, será el encargado de verificar lo anterior mediante la propiedad de terminación global, y dar inicio a la fase de conexión-reconexión de aristas dinámicas. El Algoritmo 13, muestra la lógica de programación del Algoritmo "Atendiendo mensaje $PIF-NEGOCIACIÓN$ en el vértice i".

Algoritmo 13 Atendiendo mensaje $PIF - NEGOCIACIÓN$ en el vértice i

Requiere:

Una gráfica $G = (V, E)$ con topología de malla o anillo.

Una opción $regla \in \{R1, R2, R3\}$ de regla de conexión-reconexión

Si se recibe el mensaje $PIF - NEGOCIACIÓN$ desde el vértice j **Entonces**
  $N[j] \leftarrow 1$
  Si $visitado = falso$ **Entonces**
    $padre \leftarrow j$
    $visitado \leftarrow verdadero$
    $conjuntoVecinos \leftarrow neighbors \cup vecinosConectadosDinámicos$
    Si $vecinos \setminus \{padre\} \neq \emptyset$ **Entonces**
      Para cada $k \in conjuntoVecinos \setminus \{padre\}$ **Hacer**
        Envía $PIF - NEGOCIACIÓN$ a k
      Termina Para
    Termina Si
    Si $regla = R1$ **Entonces**
      Ordena descendentemente f_n con base en las frecuencias de visita
    Termina Si
    $hayEnlacesLibres \leftarrow False$
    Si $f\_n \neq \emptyset$ **Entonces**
      Para $enlace \in enlacesDinámicos$ **Hacer**
        Si $enlace$ está desconectado **Entonces**
          $hayEnlacesLibres \leftarrow True$
          Selecciona un vértice v, candidato a conexión con base en el atributo $regla$
          $V1 \leftarrow$ coordenadas en x,y de i
          $V2 \leftarrow$ coordenadas en x,y de v
          $distancia \leftarrow \sqrt{(V1[0] - V2[0])^2 + (V1[1] - V2[1])^2}$

24: **Si** *distancia* $\leq$ longitud de *enlace* **Entonces**
25: Busca la ruta de menor longitud que permita ir de i a v en la lista *paquetes* de i y guárdala en $rutaMin$
26: Genera un nuevo paquete explorador y copia $rutaMin$ en sus atributos $ruta$ y $rutaAuxiliar$
27: $ruta \leftarrow ruta \setminus \{ruta[0]\}$ {el identificador de i}
28: $siguiente \leftarrow ruta[0]$
29: $ruta \leftarrow ruta \setminus \{ruta[0]\}$ {es lo mismo que $siguiente$}
30: Agrega el identificador de v a $listaNodosSolicitados$
31: $númeroSolicitudes \leftarrow númeroSolicitudes + 1$
32: Copia el identificador de *enlace* al atributo $idEnlace$ del paquete explorador
33: Envía $SOLICITUD - CONEXIÓN$ a $siguiente$ con el paquete explorador
34: $f\_n \leftarrow f\_n \setminus \{v\}$
35: **Si** $regla = R2$ **Entonces**
36: **Rompe el ciclo Para**
37: **Termina Si**
38: **Termina Si**
39: **Termina Si**
40: **Termina Para**
41: **Si** $hayEnlacesLibres = False$ **Entonces**
42: Busca la arista dinámica menos usada en f_e
43: Ejecuta las líneas $20 - 23$
44: **Si** *distancia* $\leq$ longitud de *enlace* y $f\_n[v] \geq (20 \cdot \frac{1}{2})$ **Entonces**
45: Ejecuta las líneas $25 - 33$
46: **Termina Si**

47: **Termina Si**
48: **Termina Si**
49: **Termina Si**
50: **Si** $N[k] = 1 \ \forall \ k \in vecinos$ y $númeroSolicitudes = 0$ **Entonces**
51: **Si** $padre \neq i$ **Entonces**
52: Envía $PIF - NEGOCIACIÓN$ a $padre$
53: Reinicia los atributos del algoritmo PIF de i a sus valores originales
54: **Sino**
55: Reinicia los atributos del algoritmo PIF de i a sus valores originales
56: Envía $PIF - CONEXIÓN$ a i {se inicia la fase de conexión-reconexión}
57: **Termina Si**
58: **Termina Si**
59: **Termina Si**

3.2.2.5.4. El mensaje $SOLICITUD - CONEXIÓN$

Como pudo observarse en el Algoritmo 13, cuando un vértice $u \in V$ envía un mensaje $SOLICITUD-CONEXIÓN$ a un vértice $i \in V$, significa que u lo seleccionó, previamente, como candidato con base en una de las tres reglas de conexión-reconexión de aristas dinámicas y, además, cumplió con las dos restricciones mostradas en la Sección 3.2.2.5.2 (en su caso). Una vez que $SOLICITUD - CONEXIÓN$ llega a i, lo primero que hace es verificar si dicho mensaje está dirigido hacia él, si este fuera el caso, copia la $rutaAuxiliar$ en $ruta$ (ambas contenidas en el paquete explorador que viaja con el mensaje), invierte la $ruta$ a modo de preparación para poder dar su respuesta al mensaje que recibió de u, elimina la primer entrada contenida en $ruta$ (que en este caso es su propio identificador), extrae la primer entrada de $ruta$, que es el identificador del vértice al que primeramente debe enviar su mensaje de respuesta, lo elimina de la $ruta$, para después revisar si en su lista de vértices solicitados ($listaNodosSolicitados$) se encuentra el identificador del vértice que le mandó la

solicitud (u), a partir de este punto, si se cumple la condición anterior, surgen dos casos:

1. Si el identificador de i es mayor que el identificador del vértice u y, además, el valor de su atributo *conexionesAceptadas* es estrictamente menor que el valor de su atributo *máximoConexionesPermitidas*, entonces, i incrementa en una unidad su atributo *conexionesAceptadas*, manda un mensaje $ACEPTO - CONEXIÓN$ a u y lo agrega a su lista *neighborsPendientes*.

2. Si no se cumplen las condiciones enunciadas en el punto 1, simplemente i le contesta a u con un mensaje $RECHAZO - CONEXIÓN$.

Caso contrario, si i no solicitó a u, surgen, de nuevo, dos casos posibles:

1. Si el valor del atributo *conexionesAceptadas* de i, es estrictamente menor que el valor de su atributo *máximoConexionesPermitidas*, entonces, i incrementa en una unidad su atributo *conexionesAceptadas*, manda a u un mensaje $ACEPTO - CONEXIÓN$ y lo agrega a su lista *neighborsPendientes*.

2. Si no se cumple la condición enunciada en el punto 1, simplemente i le contesta a u con un mensaje $RECHAZO - CONEXIÓN$.

Si el mensaje $SOLICITUD - CONEXIÓN$ no iba dirigido hacia i, simplemente lo reenvía al siguiente vértice que se encuentra en *ruta*.

El Algoritmo 13, muestra la lógica de programación del Algoritmo "Atendiendo mensaje $SOLICITUD - CONEXIÓN$ en el vértice i".

3.2.2.5.5. El mensaje $ACEPTO - CONEXIÓN$

Toda vez que un vértice i, recibe un mensaje $ACEPTO - CONEXIÓN$, revisa si

Algoritmo 14 Atendiendo mensaje $SOLICITUD - CONEXIÓN$ en el vértice i

Requiere:

Una gráfica $G = (V, E)$ con topología de malla o anillo.

Si se recibe el mensaje $SOLICITUD - CONEXIÓN$ desde el vértice j **Entonces**
{el método $len(x)$, regresa el número de elementos contenidos en la lista x}
  Si $len(ruta) > 0$ **Entonces**
    $siguiente \leftarrow ruta[0]$
    $ruta \leftarrow ruta \setminus \{ruta[0]\}$
    Envía mensaje $SOLICITUD - CONEXIÓN$ a $siguiente$
  Sino
    $ruta \leftarrow rutaAuxiliar$
    $ruta \leftarrow ruta.reverse()$ {el método $reverse()$, regresa la ruta invertida}
    $ruta \leftarrow ruta \setminus \{ruta[0]\}$ {el identificador de i}
    $siguiente \leftarrow ruta[0]$
    $ruta \leftarrow ruta \setminus \{ruta[0]\}$
    Si $u \in listaNodosSolicitados$ **Entonces**
      Si $i > u$ y $conexionesAceptadas < máximoConexionesPermitidas$ **Entonces**
        $conexionesAceptadas \leftarrow conexionesAceptadas + 1$
        Envía $ACEPTO - CONEXIÓN$ a siguiente
        $neighborsPendientes \leftarrow neighborsPendientes \cup \{u\}$
      Sino
        Envía $RECHAZO - CONEXIÓN$ a siguiente
      Termina Si
    Sino
      Si $conexionesAceptadas < máximoConexionesPermitidas$ **Entonces**
        $conexionesAceptadas \leftarrow conexionesAceptadas + 1$
        Envía $ACEPTO - CONEXIÓN$ a siguiente

24: $neighborsPendientes \leftarrow neighborsPendientes \cup \{u\}$
25: **Sino**
26: Envía $RECHAZO - CONEXIÓN$ a siguiente
27: **Termina Si**
28: **Termina Si**
29: **Termina Si**
30: **Termina Si**

dicho mensaje va dirigido hacia él, si ese fuera el caso, inmediatamente después, decrementa en una unidad su atributo *númeroSolicitudes*, agrega a su lista de solicitudes pendientes (*solicitudesPendientes*) el paquete explorador, el cual, contiene la información necesaria para realizar la conexión-reconexión en una fase posterior, busca en su lista de aristas dinámicas (*enlacesDinámicos*) a la arista implicada en el proceso, y verifica si ella está conectada a un vértice, si se cumple lo anterior, le manda un mensaje $DESCONEXIÓN$ a tal vértice, a través de la arista dinámica, es decir, este mensaje llegará en a lo más una unidad de tiempo y, finalmente, incrementa en una unidad su atributo *numeroSolicitudes*. Si el mensaje $ACEPTO-CONEXIÓN$ no iba dirigido hacia i, simplemente lo reenvía al siguiente vértice que se encuentra en *ruta*.

El Algoritmo 15, muestra la lógica de programación del Algoritmo "Atendiendo mensaje $ACEPTO - CONEXIÓN$ en el vértice i".

3.2.2.5.6. El mensaje $DESCONEXIÓN$

Toda vez que un vértice $i \in V$, recibe un mensaje $DESCONEXIÓN$ desde j, agrega a j a su lista de vecinos pendientes de eliminación (*neighborspendientesEliminacion*), decrementa en una unidad el valor de su atributo *conexionesAceptadas* y le contesta a j con un mensaje $DESCONEXIÓN - RECIBIDA$, informándole de tal suceso a través de la arista dinámica implicada en el proceso.

Algoritmo 15 Atendiendo mensaje $ACEPTO - CONEXIÓN$ en el vértice i

Requiere:

Una gráfica $G = (V, E)$ con topología de malla o anillo.

1: **Si** se recibe el mensaje $ACEPTO - CONEXIÓN$ desde el vértice j **Entonces**

{el método $len(x)$, regresa el número de elementos contenidos en la lista x}

2: **Si** $len(ruta) > 0$ **Entonces**

3: $siguiente \leftarrow ruta[0]$

4: $ruta \leftarrow ruta \setminus \{ruta[0]\}$

5: Envía mensaje $ACEPTO - CONEXIÓN$ a $siguiente$

6: **Sino**

7: $númeroSolicitudes \leftarrow númeroSolicitudes - 1$

8: Agrega el paquete explorador recibido a $solicitudesPendientes$

9: Busca en $enlacesDinámicos$, la arista $aristaDinámica$ implicada en el proceso

10: **Si** $aristaDinámica$ ya estaba conectada a un vértice $s \in N(i)$ **Entonces**

11: $númeroSolicitudes \leftarrow númeroSolicitudes + 1$

12: Envía $DESCONEXIÓN$ a s

13: **Termina Si**

14: **Termina Si**

15: **Si** $N[k] = 1 \ \forall \ k \in vecinos$ y $númeroSolicitudes = 0$ **Entonces**

16: **Si** $padre \neq i$ **Entonces**

17: Envía $PIF - NEGOCIACIÓN$ a $padre$

18: Reinicia los atributos del algoritmo PIF de i a sus valores originales

19: **Sino**

20: Reinicia los atributos del algoritmo PIF de i a sus valores originales

21: Envía $PIF - CONEXIÓN$ a i {se inicia la fase de conexión-reconexión}

22: **Termina Si**

23: **Termina Si**

24: **Termina Si**

El Algoritmo 16, muestra la lógica de programación del Algoritmo "Atendiendo mensaje $DESCONEXIÓN$ en el vértice i".

Algoritmo 16 Atendiendo mensaje $DESCONEXIÓN$ en el vértice i

Requiere:

Una gráfica $G = (V, E)$ con topología de malla o anillo.

1: **Si** se recibe el mensaje $DESCONEXIÓN$ desde el vértice j **Entonces**
2: $neighborsPendientesEliminación \leftarrow neighborsPendientesEliminación \cup \{j\}$
3: $conexionesAceptadas \leftarrow conexionesAceptadas - 1$
4: Envía $DESCONEXIÓN - RECIBIDA$ a j
5: **Termina Si**

3.2.2.5.7. El mensaje $DESCONEXIÓN - RECIBIDA$

Toda vez que un vértice $i \in V$, recibe un mensaje $DESCONEXIÓN - RECIBIDA$ desde j, decrementa en una unidad su atributo $númeroSolicitudes$, lo cual, indica que una de sus solicitudes ya fue atendida.

El Algoritmo 17, muestra la lógica de programación del Algoritmo "Atendiendo mensaje $DESCONEXIÓN - RECIBIDA$ en el vértice i".

3.2.2.5.8. El mensaje $RECHAZO - CONEXIÓN$

Toda vez que un vértice i, recibe un mensaje $RECHAZO - CONEXIÓN$, revisa si dicho mensaje va dirigido hacia él, si ese fuera el caso, inmediatamente después, decrementa en una unidad su atributo $númeroSolicitudes$. Si el mensaje $RECHAZO - CONEXIÓN$ no iba dirigido hacia i, simplemente lo reenvía al siguiente vértice que se encuentra en $ruta$.

El Algoritmo 18, muestra la lógica de programación del Algoritmo "Atendiendo mensaje

Algoritmo 17 Atendiendo mensaje $DESCONEXIÓN - RECIBIDA$ en el vértice i

Requiere:

Una gráfica $G = (V, E)$ con topología de malla o anillo.

Si se recibe el mensaje $DESCONEXIÓN - RECIBIDA$ desde el vértice j **Entonces**
  $númeroSolicitudes \leftarrow númeroSolicitudes - 1$
  Si $N[k] = 1 \ \forall \ k \in vecinos$ y $númeroSolicitudes = 0$ **Entonces**
    Si $padre \neq i$ **Entonces**
      Envía $PIF - NEGOCIACIÓN$ a $padre$
      Reinicia los atributos del algoritmo PIF de i a sus valores originales
    Sino
      Reinicia los atributos del algoritmo PIF de i a sus valores originales
      Envía $PIF - CONEXIÓN$ a i {se inicia la fase de conexión-reconexión}
    Termina Si
  Termina Si
Termina Si

$RECHAZO - CONEXIÓN$ en el vértice i".

3.2.2.6. Fase de conexión-reconexión

En esta fase, todos los vértices de la red, realizarán las conexiones y desconexiones que negociaron en la fase de negociación. Resulta necesario destacar que no se realizaron en dicha fase, debido al tránsito de mensajes existente a través de las aristas de la red. En la fase de exploración, los vértices emiten sus paquetes exploradores y guardan las rutas para llegar a los destinos correspondientes en su lista *paquetes*. Dichas rutas, son utilizadas en la fase de negociación para encaminar las solicitudes de conexión-reconexión, evitando invocar a alguno de los tres algoritmos de encaminamiento disponibles, con el objetivo de ahorrar recursos computacionales (tiempo de procesador y cantidad de memoria). Las rutas utilizadas, consideran el uso de las aristas dinámicas presentes en la fase de exploración, lo cual, eventualmente cambia en la fase de negociación. Si se realizara la conexión-desconexión de

Algoritmo 18 Atendiendo mensaje $RECHAZO - CONEXIÓN$ en el vértice i

Requiere:

Una gráfica $G = (V, E)$ con topología de malla o anillo.

1: **Si** se recibe el mensaje $RECHAZO - CONEXIÓN$ desde el vértice j **Entonces**

{el método $len(x)$, regresa el número de elementos contenidos en la lista x}

2: **Si** $len(ruta) > 0$ **Entonces**

3: $siguiente \leftarrow ruta[0]$

4: $ruta \leftarrow ruta \setminus \{ruta[0]\}$

5: Envía mensaje $RECHAZO - CONEXIÓN$ a $siguiente$

6: **Sino**

7: $númeroSolicitudes \leftarrow númeroSolicitudes - 1$

8: **Termina Si**

9: **Si** $N[k] = 1 \ \forall \ k \in vecinos$ y $númeroSolicitudes = 0$ **Entonces**

10: **Si** $padre \neq i$ **Entonces**

11: Envía $PIF - NEGOCIACIÓN$ a $padre$

12: Reinicia los atributos del algoritmo PIF de i a sus valores originales

13: **Sino**

14: Reinicia los atributos del algoritmo PIF de i a sus valores originales

15: Envía $PIF - CONEXIÓN$ a i {se inicia la fase de conexión-reconexión}

16: **Termina Si**

17: **Termina Si**

18: **Termina Si**

las aristas dinámicas en la fase de negociación, algunos mensajes no tendrían la oportunidad de llegar a sus destinos. Lo anterior, obliga a desarrollar una tercera fase para lograr los objetivos de los experimentos.

Todas las conexiones y desconexiones que se negociaron en la fase de negociación, se encuentran guardadas en listas que posee cada vértice de la red y, mediante el algoritmo PIF,

todos ellos se enterarán que deben llevar a cabo su conexión y reconexión formalmente. El mensaje que predomina en esta fase es $PIF - CONEXIÓN$, debido a que no es necesario implicar más mensajes para desarrollar su lógica de programación.

3.2.2.6.1. El mensaje $PIF - CONEXIÓN$

El mensaje $PIF-CONEXIÓN$, lleva la lógica de control del algoritmo PIF subyacente en la fase de conexión-reconexión. La lógica que se agrega en esta fase, es que, inmediatamente después de que un vértice i recibe el mensaje $PIF - CONEXIÓN$ por primera vez, revisa si existen vértices en sus listas $neighborsPendientes$ y $neighborsPendientesEliminación$, y, además, si existen paquetes en su lista $solicitudesPendientes$, si fuera el caso, i procede de la siguiente manera para cada uno de los tres casos:

1. Si existen vértices en $neighborsPendientes$, los agrega a su lista de vecinos $neighbors$.

2. Si existen vértices en $neighborsPendientesEliminación$, los elimina de su lista de vecinos $neighbors$.

3. Si existen paquetes en $solicitudesPendientes$, para cada paquete, busca su arista dinámica $arista$ implicada en el proceso (con base en el $idEnlace$ contenido en el paquete), declara que $arista$ ya no está libre y verifica si ya estaba conectada a algun vértice $s \in V$, si fuera el caso, elimina a s de su lista $vecinosConectadosDinámicos$. Después, conecta $arista$ al vértice d localizado en la última entrada del atributo $rutaAuxiliar$ del paquete en cuestión y lo agrega a su lista $vecinosConectadosDinámicos$.

Inmediatemente después, i se encuentra en totales condiciones para reexpedir su mensaje $PIF - CONEXIÓN$ a su lista de vecinos (actualizada previamente), excepto a su padre. El algoritmo PIF-CONEXIÓN, termina cuando todos los vértices de la red han recibido sus $|N(i)|$ mensajes $PIF - CONEXIÓN$ desde todos sus vecinos (conectados a través de

aristas fijas y dinámicas). El coordinador, será el encargado de verificar lo anterior mediante la propiedad de terminación global, y dar inicio a un nuevo ciclo de experimentación mediante la fase de exploración. El nuevo ciclo, solo se podrá ejecutar si el diámetro de la red no se ha reducido a dos saltos, si esto ya hubiera sucedido, no tendría sentido realizar una exploración más, debido a que todos los vértices se conectan a un vértice super-concentrador de grado $n-1$. El Algoritmo 19, muestra la lógica de programación del Algoritmo "Atendiendo mensaje $PIF - NEGOCIACIÓN$ en el vértice i".

Algoritmo 19 Atendiendo mensaje $PIF - CONEXIÓN$ en el vértice i

Requiere:

Una gráfica $G = (V, E)$ con topología de malla o anillo.

1: **Si** se recibe el mensaje $PIF - CONEXIÓN$ desde el vértice j **Entonces**
2: $N[j] \leftarrow 1$
3: **Si** $visitado = falso$ **Entonces**
4: $padre \leftarrow j$
5: $visitado \leftarrow verdadero$
{el método $len(x)$, regresa el número de elementos contenidos en la lista x}
6: **Si** $len(neighborsPendientes) > 0$ **Entonces**
7: **Para** $x \in neighborsPendientes$ **Hacer**
8: $neighborsPendientes \leftarrow neighborsPendientes \cup \{x\}$
9: **Termina Para**
10: **Termina Si**
11: **Si** $len(neighborsPendientesEliminación) > 0$ **Entonces**
12: **Para** $x \in neighborsPendientesEliminación$ **Hacer**
13: $neighborsPendientes \leftarrow neighborsPendientes \setminus \{x\}$
14: **Termina Para**
15: **Termina Si**

16: **Si** $len(solicitudesPendientes) > 0$ **Entonces**
17: **Para** $package \in solicitudesPendientes$ **Hacer**
18: **Para** $enlace \in enlacesDinámicos$ **Hacer**
19: **Si** identificador de $enlace = idEnlace$ **Entonces** {$idEnlace$ está en package}
20: Declarar que $enlace$ ya no está libre
21: **Si** $enlace$ estába ya conectado a $s \in V$ **Entonces**
22: $vecinosConectadosDinámicos \leftarrow vecinosConectadosDinámicos \setminus \{s\}$
23: **Termina Si**
24: Conectar a $enlace$ con $rutaAuxiliar[len(rutaAuxiliar) - 1]$
25: $vecinosConectadosDinámicos \leftarrow vecinosConectadosDinámicos$
$\cup \quad \{rutaAuxiliar[len(rutaAuxiliar) - 1]\}$
26: **Rompe el ciclo Para**
27: **Termina Si**
28: **Termina Para**
29: **Termina Para**
30: **Termina Si**
31: $conjuntoVecinos \leftarrow neighbors \cup vecinosConectadosDinámicos$
32: **Si** $vecinos \setminus \{padre\} \neq \emptyset$ **Entonces**
33: **Para** cada $k \in conjuntoVecinos \setminus \{padre\}$ **Hacer**
34: Envía $PIF - CONEXIÓN$ a k
35: **Termina Para**
36: **Termina Si**
37: **Termina Si**
38: **Si** $N[k] = 1 \ \forall \ k \in vecinos$ **Entonces**
39: **Si** $padre \neq i$ **Entonces**
40: Envía $PIF - CONEXIÓN$ a $padre$
41: $contadorCiclos \leftarrow contadorCiclos + 1$

Reinicia todos los atributos de i a sus valores originales
Sino
Reinicia todos los atributos de i a sus valores originales
Si $contadorCiclos < 50$ **Entonces**
Calcula el nuevo *diámetro* de la red
Genera un nuevo paquete explorador y agrégale *diámetro*
Si *diámetro* $\neq 2$ **Entonces**
Envía $PIF - EXPLORACIÓN$ a i {se inicia un nuevo ciclo}
$contadorCiclos \leftarrow contadorCiclos + 1$
Termina Si
Termina Si
Termina Si
Termina Si
Termina Si

3.3. Etapa 2: Degradación de redes

En esta investigación, se utilizaron dos estrategias diferentes de degradación:

1. Ataques secuenciales dirigidos: Se ejecutó haciendo uso de la métrica de centralidad grado de un vértice, $\delta(v)$, en orden descendente.

2. Fallas aleatorias: Se ejecutó seleccionando a los vértices de la red, con base en una función de distribución de probabilidad uniforme.

En ambas estrategias, se computó el valor de $A2TR(p) : p \in \{0, 1, 2, 3, 4, 5, ..., n-2\}$. Donde n representa el orden de la red. Lo anterior, se realizó con el objetivo de calcular el umbral de robustez $\mu - A2TR(p)$, considerando un escenario incremental de degradación con base en la Ecuación 2.26.

Más aún, cada $x\,\% : x \in \{0, 1, 2, 3, 4, 5, ..., 99\}$ de vértices removidos de la red, se calcularon las siguientes propiedades estructurales:

1. Orden relativo del componente más grande.

2. Longitud de trayectoria promedio.

3. Diámetro.

Capítulo 4

Experimentos y Resultados

"Nunca comprenderemos a los sistemas complejos, a menos que desarrollemos un profundo conocimiento de las redes que los sustentan" -Albert-László Barabási [4]

Para llevar a cabo los experimentos propuestos en esta investigación, se desarrolló un simulador de eventos discretos, el cual, fue escrito en el lenguaje de programación *python*, con base en el modelo de programación distribuida basado en paso de mensajes y bajo el paradigma de la programación orientada a objetos, el cual, hace uso de diversas bibliotecas para lograr sus objetivos. En particular, se usó la biblioteca *NetworkX*, para poder simplificar el trabajo de calcular diversas medidas de las propiedades estructurales de las redes con las que se experimentó, tomando como base, la herramienta propuesta en [34], realizando las adaptaciones correspondientes, para que dicha herramienta fuera capaz de soportar la ejecución de los algoritmos de las fases de exploración, negociación y conexión-reconexión, presentados en el capítulo anterior. Para que el lector tenga un entendimiento claro de como se realizó la ejecución de los experimentos propuestos en esta investigación, se sugiere, consulte el Apéndice C.

4.1. Experimentos cooperadores

Para comenzar, se consideró la ejecución de 18 experimentos "base", cuyas configuraciones son el resultado de las posibles combinaciones entre los elementos de los siguientes conjuntos:

-Topologías de red: $T = \{\text{Malla 50 x 50 vértices, Anillo 1500 vértices}\}$.

-Algoritmos de encaminamiento: $E = \{\text{Compass-Routing, Random-Walk, Shortest-Path}\}$.

-Reglas locales de conexión-reconexión: $R = \{R1, R2, R3\}$.

Se consideró la ejecución de cada uno de los 18 experimentos "base", tomando en cuenta los cinco tamaños posibles de arista dinámica, dando un total de $18 \cdot 5 = 90$ experimentos en total. Se determinó llevar a cabo 50 ciclos de ejecución por cada uno de ellos y, a su vez, ejecutar 10 repeticiones, para así poder obtener la media aritmética, como medida de tendencia central, y la desviación estándar, como medida de dispersión, correspondientes a los resultados finales obtenidos.

Debido a que en los 90 experimentos se tomó en cuenta que todos los vértices podían conectarse y reconectarse a su conveniencia, y que cada vértice receptor de solicitudes coopera con todas sus capacidades, permitiendo que se establezcan todas las conexiones entrantes, se decidió nombrar a este conjunto *experimentos cooperadores*.

Se realizó un análisis preliminar de los resultados obtenidos mediante los 90 experimentos cooperadores, y lo que se pudo observar es que, en algunos experimentos, en particular los que usan Shortest-Path como algoritmo de encaminamiento, se propicia la formación de vértices concentradores en la red (super-hubs), es decir, vértices que tienen una gran cantidad de vecinos (cuentan con un grado alto), lo que trae por consecuencia propiedades estructurales, tales como un diámetro y longitud de trayectoria promedio muy reducidos. Se sabe que

en un sistema real, esta acción puede generar cargas excesivas de trabajo a vértices que, quizá después, no tengan la capacidad de atender todas las peticiones que reciban siendo concentradores, por lo que se decidió definir un segundo conjunto de experimentos, en los que intervengan vértices semi-cooperadores, es decir, vértices que acepten una cantidad limitada de conexiones, con respecto al orden de la topología de la red.

4.2. Experimentos semi-cooperadores

Para definir este segundo conjunto de experimentos, se consideraron limitaciones en las conexiones que los vértices pueden aceptar, de acuerdo con siguientes configuraciones: $\frac{n}{4}$, $\frac{n}{16}$ y $\frac{n}{64}$. Donde n, representa el orden de la gráfica subyacente en la red con la que se trabaja.

Más aún, considerando tales limitaciones, se busca encontrar los experimentos en los que emergió un vértice concentrador tal que su grado, dividido por cada limitación, sea al menos tres, lo cual, quiere decir que se puede realizar una distribución de carga de las conexiones del concentrador en al menos tres vértices distintos, como sigue:

Sean:

1. v el vértice super-hub que emerge en cada uno de los 900 experimentos cooperadores (considerando las 10 ejecuciones de cada uno de ellos)

2. $\bar{\delta}(v)$ el grado medio de v considerando las 10 ejecuciones de los experimentos de acuerdo a la configuración en cuestión de cada uno de ellos

3. $A = \{4^p \mid p = 1, 2, 3\}$

4. n el orden de G

Se busca que:

$$\frac{a \cdot \bar{\delta}(v)}{n} \geq 3 \quad \forall\, a \in A \tag{4.1}$$

Aquellos experimentos que cumplan la condición mostrada en la Ecuación 4.1, y que además usen Shortest-Path como algoritmo de encaminamiento, serán aquellos que definan al conjunto de experimentos semi-cooperadores. Los experimentos que usan los algoritmos de encaminamiento Compass-Routing y Random-Walk, en su gran mayoría, no cumplen la condición mostrada en la Ecuación 4.1, por esta razón, se decidió solo considerar a los que usan Shortest-Path.

En las Tablas B.1a, B.1b, B.2a, B.2b, y B.2c del Apéndice B, se muestra el análisis completo para cada uno de los cinco posibles tamaños de arista dinámica. Las celdas con letra en color rojo, representan a los experimentos seleccionados para formar el conjunto semi-cooperador. Note que los experimentos denotados mediante celdas con letra en color azul, cumplen con la condición mostrada en la Ecuación 4.1, sin embargo, no utilizan el algoritmo Shortest-Path como algoritmo de encaminamiento, por lo tanto, no son considerados en dicho conjunto de experimentos.

En total, el conjunto de experimentos semi-cooperadores, quedó definido con 47 experimentos a ejecutar.

Una vez obtenidos los resultados y sus medidas estadísticas correspondientes de la etapa de formación de redes complejas, se procedió a ejecutar los experimentos concernientes a la fase de degradación. Se determinó ejecutar 10 repeticiones del proceso de degradación, tanto por fallas aleatorias, como por ataques secuenciales dirigidos, por cada una de las redes, para así poder obtener la media aritmética, como medida de tendencia central, y la desviación estándar, como medida de dispersión, correspondientes a los resultados finales obtenidos.

4.3. Catálogo de resultados

Las Tablas 4.1 y 4.2, muestran las métricas promedio resultantes, al cabo de 10 ejecuciones, de los 90 experimentos cooperadores en ambas etapas. Por otro lado, las Tablas 4.3 y 4.4, muestran las métricas promedio resultantes, al cabo de 10 ejecuciones, de los 47 experimentos semi-cooperadores en ambas etapas.

La nomenclatura de las Tablas 4.1, 4.2, 4.3 y 4.4, se define como sigue:

Sea G, la red con la que se trabaja:

1. $\mid E \mid$ representa la cardinalidad del conjunto de aristas de G.
2. d_G representa la densidad de G.
3. $min(S_G)$ representa el grado mínimo observado en la secuencia de grados de G.
4. $max(S_G)$ representa el grado máximo observado en la secuencia de grados de G.
5. $\langle \delta \rangle$ representa la media aritmética de la secuencia de grados de G.
6. σ representa la desviación estándar de la medida ubicada exactamente una columna atrás.
7. C_G representa el coeficiente de agrupamiento promedio de G.
8. LTP_G representa la longitud de trayectoria promedio de G.
9. D_G representa el diámetro de G.
10. CE representa el ciclo i de estabilización de la medida ubicada exactamente dos columnas atrás. Sea X_i la medida obtenida en el ciclo i. Para C_G, LTP_G y D_G, se con-

sideran tolerancias $\mid X_{50} - X_i \mid < 0.01$, $\mid X_{50} - X_i \mid < 1.1$ y $\mid X_{50} - X_i \mid < 1.1 \ \forall \ i \in \{49, 48, 47, ..., 3, 2, 1\}$, respectivamente. El primer ciclo i en donde no se cumpla la tolerancia, representará el ciclo a partir del cual se estabilizó la métrica en cuestión.

11. M representa la modularidad de G.

12. r representa el coeficiente de asortatividad de G, con respecto al grado de sus vértices.

13. $\mu - A2TR$ representa la media de la fiabilidad promedio entre dos terminales de G.

14. $A2TR(p)$ representa la fiabilidad promedio entre dos terminales de G, en el instante anterior al que se alcanza el umbral $\mu - A2TR$.

15. $\sigma(\rho)^F$ representa el orden relativo del componente más grande de G, en el instante anterior al que se alcanza el umbral $\mu - A2TR$ al degradar la red mediante fallas aleatorias.

16. LTP_G^F representa la longitud de trayectoria promedio de G, en el instante anterior al que se alcanza el umbral $\mu - A2TR$ al degradar la red mediante fallas aleatorias.

17. D_G^F representa el diámetro de G, en el instante anterior al que se alcanza el umbral $\mu - A2TR$ al degradar la red mediante fallas aleatorias.

18. $\sigma(\rho)^A$ representa el orden relativo del componente más grande de G, en el instante anterior al que se alcanza el umbral $\mu - A2TR$ al degradar la red mediante ataques secuenciales dirigidos.

19. LTP_G^A representa la longitud de trayectoria promedio de G, en el instante anterior al que se alcanza el umbral $\mu - A2TR$ al degradar la red mediante ataques secuenciales dirigidos.

20. D_G^A representa el diámetro de G, en el instante anterior al que se alcanza el umbral

$\mu - A2TR$ al degradar la red mediante ataques secuenciales dirigidos.

Al momento de degradar a las redes, mediante ambos procesos, puede que estas se dividan en varios componentes aislados, en este caso, las medidas de las propiedades estructurales son calculadas sobre su componente más grande.

En total, se ejecutaron 137 experimentos, distribuidos de la siguiente manera:

1. En el conjunto de experimentos cooperadores, 90, de los cuales:

   a) Para el anillo de 1500 vértices: 45

   b) Para la malla de 50 x 50 vértices: 45

2. En el conjunto de experimentos semi-cooperadores, 47, de los cuales:

   a) Para el anillo de 1500 vértices: 20

   b) Para la malla de 50 x 50 vértices: 27

4.3.1. Anillo 1500 vértices cooperadores

Tabla 4.1: Resultados finales (primer conjunto) anillo 1500 vértices

			Fase de formación																**Fase de degradación por ataques secuenciales dirigidos**								**Fase de degradación por fallas aleatorias**							
Id	Configuración	$\|E\|$	d_G	$min(S_G)$	$max(S_G)$	$\langle k \rangle$	σ	C_G	σ	CE	LTP_G	σ	CE	D_G	σ	CE	M	σ	r	$\mu - A2TR$	σ	$A2TR(p)$	$\sigma(\rho)^4$	LTP_G^4	D_G^4	$\mu - A2TR$	σ	$A2TR(p)$	$\sigma(\rho)^6$	LTP_G^6	D_G^6
0	Topología inicial	1500	0.001334223	2	2	2	0	0			375.25			750			0.946884	0.00016	NAN	0.01	0.000559	1	1	375.250167	750	0.01	0.000559	1	1	375.250167	750
1	D/R1/CR	4500	0.004	4	244	6	8.757	0.273	0.011	35	3.904	0.099	14	6.9	0.3	18	0.591	0.006	-0.172	0.13	0.007	0.636	0.794	23.528	63.1	0.65	0.033	0.581	0.758	6.335	17.3
2	D/R1/RW	4500	0.004	4	84	6	6.672	0.559	0.01	20	31.881	1.942	10	63.7	4.076	48	0.961	0.002	-0.321	0.02	0.006	0.502	0.608	36.567	139.2	0.1	0.017	0.283	0.428	18.013	52.3
3	D/R1/SP	4500	0.004	4	1499	6	40.921	0.621	0.021	16	2.007	0.027	9	2.6	0.49	13	0.441	0.032	-0.265	0.03	0.006	0.472	0.579	37.362	114.7	0.53	0.132	0.495	0.631	5.309	15.9
4	D/R2/CR	4500	0.004	4	17	6	1.709	0.143	0.003	16	4.826	0.014	11	8	0	12	0.573	0.005	-0.051	0.37	0.005	0.738	0.859	15.848	40.7	0.66	0.013	0.619	0.785	11.236	28.1
5	D/R2/RW	4500	0.004	4	17	6	1.605	0.182	0.003	38	8.58	0.333	46	17.1	0.7	49	0.798	0.008	-0.069	0.35	0.013	0.496	0.672	19.502	62.8	0.61	0.013	0.665	0.802	19.395	53.4
6	D/R2/SP	4500	0.004	4	1499	6	38.571	0.415	0.001	21	1.996	0.001	13	2.2	0.4	20	0.333	0.002	-0.2	0.28	0.003	0.655	0.809	18.792	46.9	0.69	0.179	0.426	0.493	6.418	15.7
7	D/R3/CR	4500	0.004	4	14	6	1.527	0.008	0.001	1	4.369	0.006	1	7	0	1	0.399	0.002	-0.03	0.39	0.006	0.723	0.85	14.197	36.6	0.68	0.008	0.659	0.811	9.229	21.8
8	D/R3/RW	4500	0.004	4	18	6	1.656	0.046	0.002	3	4.447	0.003	1	7	0	1	0.449	0.003	-0.067	0.38	0.003	0.733	0.856	15.294	40.2	0.67	0.01	0.636	0.797	10.028	24.6
9	D/R3/SP	4500	0.004	4	1499	6	38.365	0.415	0.001	24	1.997	0.001	13	2.4	0.49	25	0.333	0.002	-0.2	0.28	0.004	0.674	0.82	18.677	47.3	0.7	0.153	0.42	0.488	6.156	13.9
10	D2/R1/CR	4500	0.004	4	209	6	7.199	0.34	0.012	35	4.36	0.123	12	8	0	15	0.652	0.011	-0.179	0.15	0.006	0.747	0.863	25.385	71.4	0.62	0.025	0.593	0.769	7.371	20.8
11	D2/R1/RW	4500	0.004	4	82	6	6.627	0.554	0.011	18	32.151	2.388	10	64.3	4.941	25	0.961	0.003	-0.316	0.03	0.005	0.332	0.444	36.513	131	0.09	0.017	0.243	0.35	15.609	42.1
12	D2/R1/SP	4500	0.004	4	499	6	23.366	0.551	0.035	38	3.173	0.002	8	6.1	0.3	10	0.586	0.027	-0.208	0.04	0.004	0.44	0.565	42.971	133.9	0.61	0.074	0.471	0.644	3.793	12.3
13	D2/R2/CR	4500	0.004	4	20	6	1.911	0.188	0.005	7	5.088	0.02	11	8.1	0.3	13	0.638	0.004	-0.058	0.36	0.006	0.666	0.815	18.548	51.4	0.65	0.011	0.617	0.785	11.101	28.6
14	D2/R2/RW	4500	0.004	4	17	6	1.592	0.185	0.004	38	8.896	0.582	46	18	1.265	48	0.801	0.013	-0.064	0.35	0.007	0.614	0.775	24.4	82.4	0.6	0.02	0.619	0.767	20.333	53.7
15	D2/R2/SP	4500	0.004	4	494	6	20.966	0.399	0.013	34	3.141	0.025	11	5.6	0.49	13	0.574	0.014	-0.229	0.19	0.007	0.615	0.782	24.457	72.3	0.66	0.097	0.496	0.665	5.949	19.2
16	D2/R3/CR	4500	0.004	4	15	6	1.529	0.026	0.002	2	4.55	0.008	2	7	0	5	0.459	0.004	-0.046	0.39	0.004	0.658	0.811	14.705	41.6	0.69	0.015	0.635	0.796	9.904	25
17	D2/R3/RW	4500	0.004	4	17	6	1.646	0.054	0.003	2	4.977	0.022	1	8.8	0.4	2	0.562	0.014	-0.062	0.38	0.003	0.665	0.814	17.093	48.5	0.67	0.009	0.638	0.798	10.688	26.7
18	D2/R3/SP	4500	0.004	4	486	6	19.829	0.324	0.023	44	3.189	0.057	12	5.7	0.458	21	0.547	0.011	-0.2	0.24	0.007	0.603	0.774	21.193	55.2	0.65	0.055	0.5	0.679	6.456	21
19	D4/R1/CR	4500	0.004	4	71	6	4.83	0.302	0.009	19	5.796	0.08	11	10.8	0.4	14	0.716	0.006	-0.184	0.19	0.007	0.781	0.883	27.816	76.7	0.62	0.018	0.614	0.782	10.607	28.1
20	D4/R1/RW	4500	0.004	4	77	6	6.601	0.553	0.01	17	32.747	1.892	10	65.3	4.1	13	0.961	0.002	-0.31	0.02	0.002	0.483	0.612	34.78	146.2	0.1	0.044	0.301	0.405	17.35	45.4
21	D4/R1/SP	4500	0.004	4	231	6	15.092	0.524	0.021	31	4.705	0.046	8	9.3	0.458	11	0.681	0.012	-0.323	0.04	0.006	0.525	0.641	37.513	120.9	0.54	0.071	0.534	0.697	6.466	21.4
22	D4/R2/CR	4500	0.004	4	30	6	2.143	0.205	0.004	18	5.734	0.025	11	10.1	0.3	15	0.669	0.006	-0.083	0.34	0.006	0.647	0.801	21.496	65.8	0.64	0.01	0.627	0.791	12.335	32.5
23	D4/R2/RW	4500	0.004	4	16	6	1.608	0.186	0.003	38	10.198	0.315	44	20.5	0.5	46	0.82	0.005	-0.068	0.35	0.006	0.569	0.741	25.428	81.9	0.59	0.018	0.5	0.645	17.253	47.2
24	D4/R2/SP	4500	0.004	4	212	6	13.275	0.388	0.011	40	4.576	0.026	10	9	0	11	0.645	0.006	-0.25	0.16	0.01	0.556	0.734	31.266	89.2	0.61	0.027	0.563	0.746	7.744	24
25	D4/R3/CR	4499	0.004	4	16	5.999	1.624	0.052	0.002	26	5.659	0.007	7	10.1	0.3	16	0.616	0.007	-0.07	0.38	0.007	0.619	0.785	21.03	61.4	0.68	0.008	0.641	0.801	12.273	30.8
26	D4/R3/RW	4500	0.004	4	15	6	1.658	0.051	0.002	4	6.717	0.053	2	13	0	3	0.695	0.006	-0.069	0.38	0.01	0.603	0.773	24.967	72	0.67	0.012	0.622	0.77	14.224	36.4
27	D4/R3/SP	4499.7	0.004	3	196	6	9.63	0.243	0.015	46	4.823	0.045	8	9	0	12	0.629	0.008	-0.151	0.24	0.005	0.578	0.758	32.337	86.7	0.65	0.035	0.544	0.73	8.474	24.9
28	D8/R1/CR	4500	0.004	4	49	6	3.755	0.283	0.007	14	9.258	0.102	9	18.3	0.458	12	0.787	0.006	-0.183	0.22	0.006	0.781	0.88	39.88	110.4	0.59	0.019	0.668	0.796	17.304	49.5
29	D8/R1/RW	4500	0.004	4	85	6	6.567	0.554	0.009	17	32.576	1.909	9	65	3.95	30	0.962	0.001	-0.325	0.02	0.004	0.462	0.588	33.455	122.8	0.11	0.029	0.281	0.373	17.262	46.4
30	D8/R1/SP	4500	0.004	4	115	6	9.852	0.517	0.021	19	8.053	0.109	8	16.2	0.4	10	0.771	0.005	-0.314	0.05	0.006	0.508	0.618	32.001	108.9	0.51	0.052	0.532	0.687	9.516	29.2
31	D8/R2/CR	4500	0.004	4	26	6	2.291	0.217	0.005	21	8.548	0.038	11	17	0	16	0.744	0.005	-0.104	0.32	0.008	0.54	0.724	25.408	88.6	0.63	0.017	0.66	0.806	16.786	43.7
32	D8/R2/RW	4500	0.004	4	17	6	1.591	0.199	0.004	28	15.037	0.343	38	30.4	0.8	43	0.86	0.005	-0.075	0.34	0.009	0.524	0.707	30.911	96.7	0.54	0.019	0.451	0.575	21.251	59
33	D8/R2/SP	4500	0.004	4	80	6	7.368	0.359	0.01	14	7.737	0.029	10	15.4	0.49	12	0.724	0.005	-0.23	0.15	0.013	0.534	0.676	35.249	108.5	0.58	0.036	0.582	0.733	11.259	31.9
34	D8/R3/CR	4422	0.004	2	41	5.886	2.244	0.123	0.004	43	8.947	0.053	19	17.9	0.3	29	0.747	0.007	-0.098	0.32	0.005	0.55	0.73	25.54	86	0.63	0.017	0.621	0.755	16.761	43.9
35	D8/R3/RW	4500	0.004	4	15	6	1.655	0.063	0.002	2	10.755	0.187	3	21.4	0.49	4	0.786	0.004	-0.074	0.35	0.008	0.603	0.757	23.848	80.1	0.63	0.007	0.614	0.751	21.663	61
36	D8/R3/SP	4465.4	0.004	3	80	5.954	4.927	0.239	0.01	22	8.127	0.039	12	16	0	28	0.735	0.006	-0.133	0.23	0.011	0.582	0.732	32.316	100.7	0.62	0.027	0.554	0.709	12.947	37.4
37	D16/R1/CR	4500	0.004	4	32	6	3.069	0.268	0.007	10	16.67	0.276	8	33.6	0.8	10	0.842	0.003	-0.172	0.23	0.012	0.536	0.67	38.117	114.4	0.53	0.022	0.507	0.626	22.098	63.5
38	D16/R1/RW	4500	0.004	4	54	6	5.936	0.563	0.007	20	35.595	1.494	8	71.2	3.059	10	0.963	0.001	-0.341	0.03	0.004	0.586	0.695	48.766	153.7	0.09	0.013	0.228	0.341	16.013	43.2
39	D16/R1/SP	4499.5	0.004	4	57	5.999	6.424	0.471	0.01	12	15.185	0.205	8	30.7	0.458	12	0.834	0.003	-0.294	0.06	0.01	0.61	0.683	31.918	99.7	0.45	0.048	0.474	0.593	14.446	42.6
40	D16/R2/CR	4499.9	0.004	4	25	6	2.296	0.224	0.005	22	15.685	0.075	13	31.8	0.4	26	0.825	0.003	-0.125	0.29	0.013	0.661	0.751	34.184	94.3	0.56	0.022	0.582	0.707	24.098	68.1
41	D16/R2/RW	4500	0.004	4	15	6	1.576	0.23	0.005	18	26.424	0.824	28	53.8	1.661	43	0.904	0.005	-0.08	0.26	0.023	0.512	0.646	35.028	101.5	0.39	0.062	0.537	0.626	29.119	86
42	D16/R2/SP	4499.9	0.004	4	47	6	4.783	0.355	0.007	32	14.508	0.065	10	29	0	16	0.809	0.003	-0.219	0.16	0.014	0.573	0.7	40.956	119.1	0.53	0.042	0.507	0.627	16.386	47.1
43	D16/R3/CR	4083.8	0.004	2	37	5.445	2.613	0.237	0.006	38	17.42	0.367	33	35.5	0.922	41	0.864	0.002	-0.111	0.19	0.021	0.552	0.655	34.867	103.2	0.48	0.019	0.414	0.579	19.763	57.4
44	D16/R3/RW	4500	0.004	4	16	6	1.609	0.094	0.003	3	19.253	0.299	5	39	0.775	7	0.85	0.002	-0.072	0.34	0.007	0.527	0.683	36.227	113.1	0.54	0.023	0.492	0.62	25.168	73
45	D16/R3/SP	4228.1	0.004	2	43	5.637	4.025	0.319	0.007	41	15.694	0.152	22	32.2	0.4	27	0.838	0.002	-0.16	0.16	0.008	0.472	0.592	38.793	113.9	0.51	0.032	0.516	0.637	18.083	52.2

4.3.2. Malla 50 x 50 vértices cooperadores

Tabla 4.2: Resultados finales (primer conjunto) malla 50 × 50 vértices

			Fase de formación													Fase de degradación por ataques secuenciales dirigidos						Fase de degradación por fallas aleatorias					
Id	Configuración	$\|E\|$	d_G	$min(Sc)$	$max(Sc)$	$\langle\delta\rangle$	σ	C_G	σ	CE	LTP_G	σ	CE	D_G	σ	CE	M	σ	r	$\mu - A2TR$	σ	$A2TR(p)$	$\sigma(\rho)^A$	LTP_G^A	D_G^A	$\mu - A2TR$	σ	$A2TR(p)$	$\sigma(\rho)^F$	LTP_G^F	D_G^F
0	Topología inicial	4900	0.001568627	2	4	3.92		0			33.333			98			0.861202	0.001314	0.6504	0.334467	0.002979	0.744125	0.862412	53.530144	124.1	0.378727	0.01381	0.574413	0.730937	50.530777	134.6
1	D/R1/CR	9900	0.003	4	157	7.92	5.623	0.158	0.002	14	4.386	0.04	14	8	0	28	0.65	0.004	-0.017	0.38	0.011	0.522	0.708	28.471	92.4	0.73	0.011	0.618	0.786	8.303	22.6
2	D/R1/RW	9900	0.003	4	318	7.92	11.679	0.462	0.01	44	9.921	0.224	36	19.9	0.7	44	0.868	0.004	-0.131	0.35	0.02	0.556	0.731	59.161	177.6	0.59	0.027	0.549	0.702	19.366	55.9
3	D/R1/SP	9900	0.003	4	2499	7.92	52.226	0.443	0.022	16	2.002	0.012	7	2.4	0.49	10	0.546	0.031	-0.177	0.32	0.014	0.622	0.784	48.296	161.2	0.67	0.117	0.497	0.607	6.22	18.2
4	D/R2/CR	9900	0.003	4	25	7.92	1.971	0.093	0.002	7	4.474	0.011	6	7	0	10	0.531	0.005	0.052	0.48	0.003	0.67	0.818	18.412	49.6	0.74	0.007	0.646	0.803	10.162	26.3
5	D/R2/RW	9900	0.003	4	19	7.92	1.582	0.184	0.002	19	6.541	0.149	25	13.3	0.458	39	0.774	0.003	-0.029	0.47	0.005	0.573	0.753	24.946	79.3	0.71	0.005	0.703	0.838	17.678	47.2
6	D/R2/SP	9900	0.003	4	2499	7.92	49.824	0.296	0	16	1.998	0	8	2.9	0.3	11	0.372	0.003	-0.145	0.41	0.001	0.705	0.839	15.497	42.7	0.78	0.112	0.552	0.645	8.28	20.1
7	D/R3/CR	9900	0.003	4	19	7.92	1.604	0.007	0.001	1	4.163	0.003	1	6	0	1	0.377	0.003	0.026	0.48	0.003	0.694	0.833	14.462	35.4	0.76	0.005	0.65	0.806	9.333	22.4
8	D/R3/RW	9900	0.003	4	17	7.92	1.614	0.055	0.002	4	4.294	0.006	1	6.1	0.3	5	0.476	0.005	-0.03	0.48	0.004	0.706	0.84	15.29	39.4	0.75	0.005	0.656	0.81	10.13	26.4
9	D/R3/SP	9900	0.003	4	2499	7.92	49.828	0.296	0	29	1.997	0	14	2.7	0.458	26	0.373	0.004	-0.145	0.41	0.003	0.696	0.834	15.788	45.3	0.75	0.09	0.545	0.689	9.631	23.6
10	D2/R1/CR	9900	0.003	4	165	7.92	5.604	0.158	0.002	14	4.378	0.05	15	8	0	29	0.652	0.007	-0.014	0.38	0.016	0.544	0.724	26.768	83.5	0.72	0.011	0.634	0.795	8.564	24.2
11	D2/R1/RW	9900	0.003	4	332	7.92	11.708	0.464	0.011	45	9.548	0.484	37	19.1	0.943	43	0.868	0.004	-0.132	0.36	0.017	0.533	0.714	57.434	164	0.61	0.025	0.549	0.71	21.892	62.3
12	D2/R1/SP	9900	0.003	4	2481	7.92	51.055	0.442	0.021	14	2.05	0.037	7	3.9	0.3	9	0.55	0.037	-0.176	0.32	0.012	0.63	0.789	47.031	151.4	0.69	0.099	0.493	0.61	4.486	15.2
13	D2/R2/CR	9900	0.003	4	33	7.92	2.416	0.098	0.001	6	4.416	0.008	6	7	0	10	0.539	0.004	0.035	0.46	0.005	0.684	0.826	18.049	51.9	0.75	0.01	0.623	0.789	10.294	26.6
14	D2/R2/RW	9900	0.003	4	19	7.92	1.58	0.185	0.003	21	6.618	0.151	24	13.5	0.896	39	0.775	0.004	-0.029	0.46	0.004	0.652	0.805	20.747	66.6	0.7	0.012	0.711	0.842	17.308	46
15	D2/R2/SP	9900	0.003	4	2474	7.92	48.645	0.293	0.002	18	2.044	0.03	8	4	0	9	0.391	0.007	-0.141	0.41	0.005	0.661	0.813	17.152	50.7	0.77	0.09	0.561	0.653	6.732	18.1
16	D2/R3/CR	9900	0.003	4	21	7.92	2.066	0.009	0.001	1	4.146	0.003	1	6	0	2	0.392	0.003	0.075	0.47	0.004	0.692	0.832	14.9	41.1	0.75	0.007	0.647	0.804	9.11	22.7
17	D2/R3/RW	9900	0.003	4	18	7.92	1.637	0.057	0.002	4	4.339	0.004	1	7	0	3	0.502	0.005	-0.022	0.48	0.003	0.65	0.806	15.607	41.2	0.75	0.006	0.646	0.803	10.051	25.3
18	D2/R3/SP	9900	0.003	4	2468	7.92	47.569	0.278	0.017	25	2.086	0.083	13	4	0	14	0.386	0.004	-0.137	0.41	0.005	0.66	0.812	16.102	47.6	0.73	0.091	0.525	0.71	11.544	30.8
19	D4/R1/CR	9900	0.003	4	57	7.92	3.836	0.134	0.002	6	4.955	0.023	12	10	0	19	0.686	0.003	0.092	0.41	0.01	0.521	0.694	25.243	77.5	0.72	0.015	0.647	0.803	10.114	27
20	D4/R1/RW	9900	0.003	4	296	7.92	11.125	0.464	0.009	44	9.839	0.337	36	19.5	0.671	45	0.871	0.003	-0.138	0.36	0.014	0.513	0.694	55.277	167.5	0.6	0.02	0.587	0.743	20.165	54.4
21	D4/R1/SP	9900	0.003	4	948	7.92	21.083	0.332	0.01	10	3.362	0.06	6	7.1	0.7	8	0.728	0.011	-0.074	0.33	0.013	0.496	0.68	39.278	129.4	0.65	0.031	0.535	0.69	12.512	37.3
22	D4/R2/CR	9900	0.003	4	39	7.92	2.72	0.105	0.001	5	4.648	0.018	6	8.2	0.4	10	0.639	0.005	0.117	0.46	0.006	0.593	0.769	23.072	71	0.74	0.008	0.635	0.797	9.857	25.6
23	D4/R2/RW	9900	0.003	4	21	7.92	1.586	0.187	0.002	18	7.106	0.071	20	15.3	0.64	33	0.783	0.004	-0.031	0.46	0.008	0.602	0.771	20.793	65.5	0.7	0.011	0.692	0.822	18.257	48.4
24	D4/R2/SP	9900	0.003	4	948	7.92	23.676	0.245	0.012	27	3.026	0.03	7	5.7	0.458	10	0.617	0.004	-0.103	0.37	0.006	0.631	0.793	29.812	93.1	0.72	0.031	0.599	0.77	7.637	24.8
25	D4/R3/CR	9900	0.003	4	23	7.92	2.23	0.026	0.001	1	4.441	0.005	2	8	0	6	0.563	0.005	0.165	0.47	0.005	0.598	0.772	19.504	62	0.75	0.005	0.639	0.799	9.715	26.3
26	D4/R3/RW	9900	0.003	4	18	7.92	1.63	0.053	0.002	2	4.97	0.022	1	9.5	0.5	2	0.645	0.006	-0.025	0.47	0.004	0.664	0.814	16.716	51.2	0.75	0.007	0.659	0.811	11.268	27.6
27	D4/R3/SP	9900	0.003	4	948	7.92	21.35	0.164	0.012	35	3.111	0.029	10	6	0	13	0.574	0.012	-0.076	0.41	0.004	0.617	0.784	26.103	78.7	0.74	0.035	0.601	0.771	8.245	23.6
28	D8/R1/CR	9900	0.003	4	27	7.92	2.289	0.143	0.002	16	6.889	0.048	12	15.5	0.5	23	0.752	0.005	0.132	0.44	0.009	0.62	0.783	22.58	66.8	0.71	0.007	0.63	0.79	14.58	39
29	D8/R1/RW	9900	0.003	4	176	7.92	9.359	0.46	0.006	42	10.526	0.369	32	21.3	0.781	40	0.873	0.002	-0.151	0.35	0.015	0.537	0.712	49.831	156.6	0.6	0.019	0.591	0.732	18.921	51.6
30	D8/R1/SP	9900	0.003	4	240	7.92	7.38	0.26	0.005	9	5.356	0.073	8	11.3	0.458	12	0.794	0.004	-0.042	0.35	0.006	0.568	0.747	31.933	107.7	0.67	0.022	0.607	0.763	11.343	32.9
31	D8/R2/CR	9900	0.003	4	24	7.92	2.311	0.118	0.002	5	6.034	0.021	5	13.1	0.3	9	0.723	0.005	0.08	0.45	0.011	0.549	0.723	22.83	70.2	0.73	0.011	0.604	0.767	12.666	32.2
32	D8/R2/RW	9900	0.003	4	17	7.92	1.572	0.197	0.003	12	8.912	0.099	11	20.8	0.6	16	0.809	0.003	-0.032	0.46	0.006	0.695	0.774	23.47	73.3	0.68	0.013	0.684	0.823	21.286	56.8
33	D8/R2/SP	9900	0.003	4	233	7.92	9.477	0.218	0.006	28	4.694	0.026	6	9.9	0.3	12	0.72	0.006	-0.084	0.35	0.01	0.549	0.73	29.278	102	0.71	0.014	0.64	0.798	8.794	27.5
34	D8/R3/CR	9886.9	0.003	3	22	7.91	2.093	0.07	0.002	5	5.978	0.023	6	13.1	0.3	19	0.701	0.006	0.088	0.46	0.006	0.563	0.742	23.831	75.2	0.73	0.008	0.67	0.818	13.225	34.6
35	D8/R3/RW	9900	0.003	4	20	7.92	1.642	0.066	0.002	2	6.762	0.034	1	15	0	3	0.743	0.007	-0.023	0.46	0.008	0.632	0.792	18.892	56.6	0.73	0.007	0.638	0.792	15.616	40.8
36	D8/R3/SP	9892	0.003	3	223	7.914	7.477	0.152	0.006	36	4.868	0.031	8	10.6	0.49	18	0.7	0.006	-0.035	0.38	0.009	0.557	0.742	23.248	79.7	0.72	0.01	0.622	0.787	9.73	29.7
37	D16/R1/CR	9897.5	0.003	4	16	7.918	1.56	0.198	0.002	31	10.528	0.032	9	26.9	0.3	12	0.806	0.002	0.168	0.46	0.002	0.619	0.783	22.195	61.7	0.68	0.01	0.648	0.795	20.549	55.6
38	D16/R1/RW	9900	0.003	4	60	7.92	5.742	0.436	0.004	41	12.238	0.186	32	28.7	0.64	43	0.872	0.002	-0.173	0.35	0.01	0.538	0.702	41.377	129.9	0.6	0.012	0.575	0.74	24.301	61.9
39	D16/R1/SP	9897	0.003	4	59	7.918	2.886	0.241	0.003	11	9.26	0.113	9	20.3	0.458	17	0.824	0.003	0.022	0.42	0.007	0.629	0.789	29.586	83.7	0.66	0.015	0.564	0.728	18.352	49
40	D16/R2/CR	9900	0.003	4	20	7.92	1.951	0.145	0.002	10	9.23	0.033	5	23.1	0.3	10	0.785	0.003	0.027	0.44	0.003	0.644	0.798	23.139	67.5	0.7	0.01	0.677	0.82	19.655	51.3
41	D16/R2/RW	9900	0.003	4	17	7.92	1.544	0.24	0.004	8	12.257	0.113	5	30.4	0.49	7	0.834	0.002	-0.032	0.45	0.004	0.62	0.784	26.005	73.2	0.64	0.01	0.547	0.693	23.569	63.3
42	D16/R2/SP	9899.9	0.003	4	54	7.92	4.036	0.194	0.004	6	8.146	0.028	6	19.5	0.5	15	0.789	0.003	-0.053	0.37	0.005	0.573	0.748	28.378	87.6	0.68	0.02	0.621	0.775	17.802	47.5
43	D16/R3/CR	9471	0.003	2	18	7.577	1.856	0.15	0.002	21	9.756	0.056	17	25.2	0.748	32	0.794	0.001	0.058	0.43	0.006	0.58	0.756	24.372	74.2	0.68	0.009	0.648	0.797	19.531	52.5
44	D16/R3/RW	9900	0.003	4	16	7.92	1.6	0.135	0.002	4	10.608	0.063	3	25.9	0.539	5	0.811	0.003	-0.026	0.45	0.004	0.651	0.804	23.446	68.1	0.68	0.014	0.655	0.796	22.974	62.2
45	D16/R3/SP	9585.5	0.003	3	43	7.668	3.472	0.2	0.004	21	8.628	0.035	14	21.4	0.49	23	0.8	0.003	-0.035	0.38	0.004	0.536	0.721	28.562	91.6	0.67	0.017	0.611	0.765	16.915	44.9

4.3.3. Anillo 1500 vértices semi-cooperadores

Tabla 4.3: Resultados finales (segundo conjunto) anillo 1500 vértices

	Fase de formación																		**Fase de degradación por ataques secuenciales dirigidos**						**Fase de degradación por fallas aleatorias**					
Id	Configuración	$\|E\|$	d_G	$min(S_G)$	$max(S_G)$	$\langle\delta\rangle$	σ	C_G	σ	CE	LTP_G	σ	CE	D_G	σ	CE	M	σ	r	$\mu - A2TR$	σ	$A2TR(p)$	$\sigma(\rho)^A$	LTP_G^A	D_G^A	$\mu - A2TR$	σ	$A2TR(p)$	$\sigma(\rho)^E$	LTP_G^E	D_G^E
0	Topología inicial	1500	0.001334223	2	2	2	0	0			375.25			750			0.946884	0.00016	NAN	0.01	0.000559	1	1	375.250167	750	0.01	0.000559	1	1	375.250167	750
1	D/R1/SP/n16	4500	0.004	4	98	6	10.535	0.357	0.013	15	4.033	0.052	7	7.5	0.5	10	0.717	0.01	-0.268	0.06	0.006	0.485	0.598	31.639	92.1	0.62	0.065	0.471	0.647	5.662	16.2
2	D/R1/SP/n4	4500	0.004	4	379	6	19.751	0.428	0.027	11	3.055	0.074	8	5.7	0.458	9	0.622	0.027	-0.243	0.04	0.004	0.583	0.699	28.872	90.1	0.64	0.007	0.536	0.709	3.921	12.7
3	D/R1/SP/n64	4500	0.004	4	28	6	5.313	0.333	0.005	15	5.588	0.236	9	10.6	0.49	16	0.811	0.014	-0.185	0.08	0.009	0.452	0.578	21.778	65.3	0.59	0.027	0.544	0.723	9.529	26
4	D/R2/SP/n16	4500	0.004	4	98	6	9.713	0.201	0.007	20	3.486	0.02	10	5.4	0.49	12	0.506	0.009	-0.215	0.23	0.005	0.612	0.781	21.837	57.6	0.66	0.041	0.603	0.772	5.549	15.5
5	D/R2/SP/n4	4500	0.004	4	378	6	19.316	0.244	0.007	16	2.85	0.018	11	4.9	0.3	12	0.476	0.011	-0.224	0.25	0.009	0.61	0.779	19.939	51.4	0.62	0.06	0.57	0.743	5.323	19.3
6	D/R2/SP/n64	4500	0.004	4	28	6	4.737	0.19	0.005	23	4.106	0.008	12	6.2	0.4	16	0.531	0.005	-0.125	0.25	0.004	0.645	0.802	21.524	54.6	0.64	0.014	0.599	0.772	7.026	18.8
7	D/R3/SP/n16	4500	0.004	4	98	6	8.841	0.085	0.005	20	3.483	0.027	1	5.7	0.458	6	0.436	0.003	-0.157	0.29	0.008	0.744	0.861	16.321	42.6	0.68	0.032	0.606	0.776	5.585	16.1
8	D/R3/SP/n4	4500	0.004	4	379	6	17.231	0.128	0.024	25	2.914	0.089	8	4.9	0.3	9	0.409	0.008	-0.163	0.29	0.014	0.712	0.841	16.528	44.2	0.66	0.051	0.581	0.75	5.849	16.7
9	D/R3/SP/n64	4500	0.004	4	28	6	4.249	0.061	0.004	10	4.035	0.01	1	6	0	1	0.436	0.004	-0.097	0.31	0.004	0.641	0.8	20.584	52.9	0.68	0.017	0.582	0.762	7.111	19.3
10	D2/R1/SP/n16	4500	0.004	4	98	6	10.434	0.365	0.012	19	4.228	0.054	7	8	0	9	0.721	0.013	-0.264	0.06	0.005	0.551	0.688	36.465	104.3	0.61	0.047	0.512	0.694	7.309	21.4
11	D2/R1/SP/n64	4500	0.004	4	28	6	5.376	0.335	0.009	20	5.578	0.221	9	10.5	0.671	15	0.808	0.013	-0.193	0.08	0.008	0.608	0.709	28.865	84	0.59	0.035	0.552	0.706	9.794	26.7
12	D2/R2/SP/n16	4500	0.004	4	98	6	10.052	0.273	0.006	39	3.726	0.015	10	6.1	0.3	12	0.604	0.009	-0.234	0.2	0.007	0.633	0.794	25.341	72.3	0.65	0.034	0.561	0.746	6.232	18.7
13	D2/R2/SP/n64	4500	0.004	4	28	6	4.853	0.226	0.007	37	4.31	0.019	12	7	0	15	0.588	0.005	-0.139	0.24	0.006	0.655	0.808	25.517	64.5	0.64	0.027	0.603	0.776	7.724	20.7
14	D2/R3/SP/n16	4500	0.004	4	98	6	8.877	0.149	0.005	31	3.759	0.017	4	6.5	0.5	8	0.515	0.005	-0.165	0.27	0.006	0.634	0.795	21.135	55.3	0.66	0.032	0.57	0.75	6.473	18.4
15	D2/R3/SP/n64	4500	0.004	4	28	6	4.458	0.108	0.006	14	4.24	0.012	2	7	0	4	0.51	0.005	-0.125	0.28	0.007	0.673	0.819	21.076	53.2	0.67	0.016	0.62	0.786	7.486	18.5
16	D4/R1/SP/n64	4500	0.004	4	28	6	5.434	0.342	0.011	25	6.344	0.152	10	12.2	0.4	18	0.809	0.011	-0.208	0.08	0.009	0.425	0.519	23.871	67.2	0.59	0.033	0.526	0.689	11.734	30.6
17	D4/R2/SP/n64	4500	0.004	4	28	6	4.902	0.251	0.005	29	5.101	0.014	12	9	0	18	0.644	0.006	-0.169	0.22	0.01	0.595	0.745	34.969	94.3	0.63	0.023	0.626	0.791	8.62	21.2
18	D4/R3/SP/n64	4499.7	0.004	4	28	6	4.287	0.154	0.003	14	5.178	0.02	6	9.6	0.49	11	0.616	0.007	-0.127	0.27	0.007	0.635	0.792	25.703	69.7	0.66	0.024	0.589	0.765	9.4	24.5
19	D8/R1/SP/n64	4500	0.004	4	28	6	5.498	0.373	0.007	30	8.731	0.09	10	17.6	0.49	14	0.815	0.006	-0.243	0.08	0.007	0.586	0.681	27.251	81.4	0.55	0.019	0.483	0.623	11.868	34.3
20	D8/R2/SP/n64	4500	0.004	4	28	6	4.776	0.302	0.004	26	7.838	0.032	12	15.7	0.458	19	0.728	0.007	-0.201	0.19	0.009	0.539	0.71	34.142	104.1	0.61	0.023	0.583	0.744	13.709	36.5

4.3.4. Malla 50 x 50 vértices semi-cooperadores

Tabla 4.4: Resultados finales (segundo conjunto) malla 50 × 50 vértices

						Fase de formación										**Fase de degradación por ataques secuenciales dirigidos**					**Fase de degradación por fallas aleatorias**				
Id	Configuración	$\|E\|$	d_G	$min(S_G)$	$max(S_G)$	$\langle\delta\rangle$	σ	C_G	σ	CE	LTP_G	σ	CE	D_G	σ	CE	M	σ	r	$\mu - A2TR$	σ	$A2TR(p)$	$\sigma(\rho)^4$	LTP_G^4	D_G^4	$\mu - A2TR$	σ	$A2TR(p)$	$\sigma(\rho)^F$	LTP_G^F	D_G^F
0	Topología inicial	4900	0.001568627	2	4	3.92		0			33.333			98			0.861262	0.001314	0.6504	0.334467	0.002979	0.744125	0.862412	33.530144	124.1	0.578727	0.01381	0.574413	0.739937	50.530777	134.6
1	D/R1/SP/n16	9900	0.003	4	163	7.92	11.389	0.283	0.011	9	3.841	0.088	6	6.4	0.49	9	0.766	0.017	-0.126	0.34	0.011	0.566	0.739	41.292	137.5	0.68	0.017	0.516	0.71	9.493	29.9
2	D/R1/SP/n4	9900	0.003	4	631	7.92	22.059	0.321	0.016	11	3.128	0.121	6	5.6	0.49	9	0.703	0.019	-0.131	0.33	0.013	0.583	0.754	40.987	129.9	0.69	0.05	0.56	0.715	6.2	23.8
3	D/R1/SP/n64	9900	0.003	4	46	7.92	6.206	0.249	0.006	3	4.691	0.05	5	8.2	0.4	9	0.789	0.008	-0.094	0.35	0.01	0.567	0.737	34.942	116.5	0.68	0.021	0.565	0.714	11.336	31.8
4	D/R2/SP/n16	9900	0.003	4	163	7.92	12.571	0.127	0.002	7	3.359	0.011	6	5	0	8	0.458	0.005	-0.152	0.4	0.003	0.661	0.813	20.812	61.4	0.74	0.031	0.554	0.736	6.301	18.8
5	D/R2/SP/n4	9900	0.003	4	624	7.92	24.913	0.148	0.004	10	2.792	0.011	7	4.7	0.458	8	0.412	0.006	-0.139	0.4	0.005	0.668	0.817	17.8	48.4	0.76	0.06	0.543	0.713	7.112	22.2
6	D/R2/SP/n64	9900	0.003	4	46	7.92	6.255	0.114	0.003	8	3.834	0.011	6	6	0	8	0.475	0.003	-0.09	0.4	0.004	0.688	0.829	20.826	62.5	0.74	0.007	0.613	0.782	6.988	18.4
7	D/R3/SP/n16	9900	0.003	4	163	7.92	11.141	0.057	0.002	17	3.401	0.013	1	5	0	2	0.405	0.004	-0.108	0.43	0.003	0.667	0.816	17.169	42.4	0.75	0.019	0.616	0.784	6.449	18.2
8	D/R3/SP/n4	9900	0.003	4	631	7.92	23.443	0.098	0.01	22	2.828	0.04	9	5	0	2	0.393	0.002	-0.132	0.42	0.007	0.644	0.802	17.446	47.2	0.71	0.033	0.618	0.783	8.526	23.7
9	D/R3/SP/n64	9900	0.003	4	46	7.92	5.477	0.044	0.002	8	3.835	0.012	1	6	0	1	0.416	0.003	-0.068	0.43	0.003	0.702	0.838	16.33	41.3	0.75	0.01	0.617	0.785	7.624	19.7
10	D2/R1/SP/n16	9900	0.003	4	163	7.92	11.352	0.285	0.012	8	3.837	0.053	5	6.6	0.49	9	0.765	0.011	-0.122	0.34	0.014	0.578	0.748	39.775	123.2	0.7	0.036	0.532	0.702	8.393	25.2
11	D2/R1/SP/n4	9900	0.003	4	631	7.92	22.159	0.324	0.014	9	3.118	0.145	6	5.8	0.4	8	0.703	0.033	-0.132	0.34	0.011	0.583	0.753	48.978	162.7	0.71	0.056	0.564	0.721	5.413	19.2
12	D2/R1/SP/n64	9900	0.003	4	46	7.92	6.26	0.248	0.01	3	4.673	0.12	6	8	0	11	0.785	0.013	-0.099	0.36	0.01	0.55	0.733	41.311	134.4	0.69	0.023	0.614	0.776	11.273	30.5
13	D2/R2/SP/n16	9900	0.003	4	163	7.92	12.692	0.133	0.004	12	3.35	0.014	6	5	0	8	0.473	0.008	-0.152	0.39	0.006	0.663	0.813	21.121	65.5	0.74	0.032	0.567	0.746	6.675	19
14	D2/R2/SP/n4	9900	0.003	4	630	7.92	25.028	0.163	0.006	14	2.796	0.017	7	4.6	0.49	9	0.433	0.006	-0.156	0.4	0.006	0.647	0.804	19.264	54	0.73	0.054	0.558	0.729	7.962	24.1
15	D2/R2/SP/n64	9900	0.003	4	46	7.92	6.393	0.118	0.002	7	3.818	0.013	6	6	0	8	0.487	0.004	-0.095	0.4	0.008	0.646	0.801	24.461	73.9	0.75	0.014	0.583	0.762	6.906	17.3
16	D2/R3/SP/n16	9900	0.003	4	163	7.92	11.687	0.065	0.003	16	3.363	0.02	1	5	0	6	0.418	0.002	-0.121	0.42	0.005	0.646	0.803	17.789	51.1	0.74	0.023	0.63	0.791	7.005	20.7
17	D2/R3/SP/n4	9900	0.003	4	631	7.92	23.459	0.101	0.011	23	2.833	0.042	9	5	0	5	0.404	0.005	-0.132	0.41	0.009	0.673	0.82	16.356	49.4	0.75	0.038	0.575	0.754	6.672	20.7
18	D2/R3/SP/n64	9900	0.003	4	46	7.92	5.799	0.051	0.001	8	3.805	0.012	1	6	0	2	0.429	0.003	-0.08	0.42	0.003	0.706	0.84	17.316	50.5	0.75	0.011	0.604	0.777	7.332	20.2
19	D4/R1/SP/n16	9900	0.003	4	163	7.92	10.714	0.291	0.009	8	4.01	0.069	6	7.5	0.5	8	0.766	0.017	-0.114	0.34	0.01	0.532	0.715	41.415	129.5	0.67	0.031	0.539	0.7	9.678	29.9
20	D4/R1/SP/n64	9900	0.003	4	46	7.92	6.16	0.252	0.005	3	4.822	0.102	6	8.7	0.64	12	0.789	0.009	-0.096	0.36	0.013	0.521	0.707	40.178	133.9	0.69	0.024	0.61	0.774	11.06	33.7
21	D4/R2/SP/n16	9900	0.003	4	163	7.92	12.493	0.173	0.004	16	3.512	0.017	6	6	0	8	0.602	0.008	-0.149	0.37	0.007	0.622	0.786	26.862	87.6	0.72	0.022	0.552	0.739	6.682	22.3
22	D4/R2/SP/n64	9900	0.003	4	46	7.92	6.489	0.14	0.004	5	3.972	0.008	6	6.8	0.4	9	0.599	0.006	-0.095	0.39	0.01	0.573	0.749	29.719	84.5	0.73	0.009	0.594	0.77	7.322	20.4
23	D4/R3/SP/n16	9900	0.003	4	163	7.92	10.564	0.105	0.003	30	3.578	0.015	4	6	0	11	0.573	0.011	-0.094	0.4	0.005	0.629	0.792	21.775	71.5	0.74	0.033	0.583	0.758	7.493	22.3
24	D4/R3/SP/n64	9900	0.003	4	46	7.92	5.873	0.085	0.002	17	3.965	0.008	2	7	0	6	0.565	0.006	-0.071	0.41	0.005	0.565	0.745	22.729	73.2	0.74	0.014	0.628	0.792	7.571	20.7
25	D8/R1/SP/n64	9900	0.003	4	46	7.92	5.276	0.243	0.004	4	5.693	0.086	7	11.6	0.49	14	0.795	0.005	-0.061	0.36	0.006	0.521	0.71	29.494	94.7	0.67	0.018	0.584	0.75	12.57	38.9
26	D8/R2/SP/n64	9900	0.003	4	46	7.92	5.814	0.181	0.003	13	4.942	0.016	6	10	0	11	0.715	0.005	-0.071	0.36	0.007	0.578	0.752	27.926	97.8	0.72	0.013	0.598	0.771	9.939	29.1
27	D8/R3/SP/n64	9890.8	0.003	3	46	7.913	5.039	0.133	0.002	13	5.048	0.021	7	10.4	0.49	21	0.702	0.003	-0.043	0.39	0.009	0.534	0.718	24.386	75.7	0.73	0.008	0.624	0.789	10.741	30.6

4.4. Discusión de resultados

A continuación, se muestra la discusión de los resultados obtenida después de realizar un análisis de los datos mostrados en las Tablas 4.1, 4.2, 4.3 y 4.4. Se usarán los identificadores de cada uno de los experimentos (mostrados en la primer columna de cada Tabla) para referenciarlos a lo largo de esta discusión.

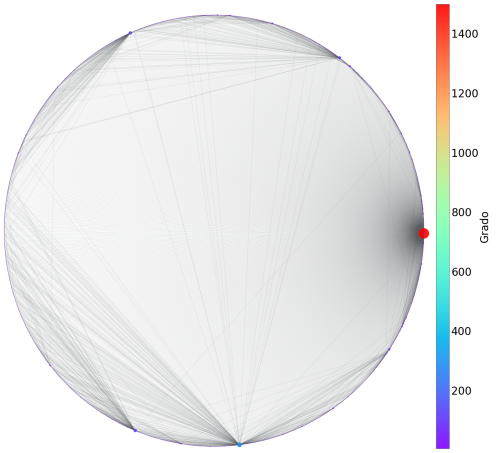
4.4.1. Anillo 1500 vértices cooperadores

Para la topología Anillo con 1500 vértices, en experimentos cooperadores:

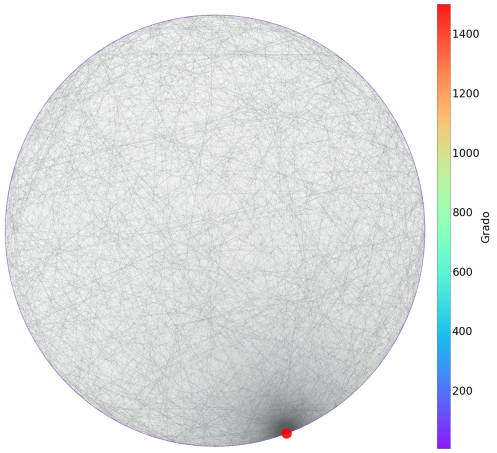
1. Las redes que resultan de los experimentos en los que se usa cualquier regla de conexión-reconexión y el algoritmo Shortest-Path, con tamaño de arista dinámica D (señalados con identificadores 3, 6 y 9, en la Tabla 4.1), tienden a que su diámetro se reduzca a dos unidades, indicando que todos los vértices conectan una de sus dos aristas dinámicas a un vértice concentrador de grado 1499. Para $R1$, $R2$ y $R3$, el diámetro se estabiliza en 2.6±0.49, 2.2±0.4 y 2.4±0.49 unidades, al cabo de 13, 20 y 25 ciclos de experimentación, respectivamente, lo cual, quiere decir que no en todos los experimentos fue necesario terminar los 50 ciclos, debido a que una vez que el diámetro está establecido en dos unidades y se usa Shortest-Path como algoritmo de encaminamiento, ya no se permite que los vértices reconecten sus aristas dinámicas, dado que todo el tráfico de paquetes pasa a través del concentrador. Sin embargo, esto no es necesariamente bueno, debido a que puede que el concentrador no tenga la capacidad de atender todas las peticiones de todos los vértices y, eventualmente, puede perder funcionalidad y salir de operación. Más aún, al atacar, de manera dirigida por grado a la red, el concentrador sería el primer vértice en salir de operación. El experimento donde se usa la regla $R1$ (señalado con identificador 3, en la Tabla 4.1), provoca la emergencia de un concentrador de grado 1499 y varios concentradores locales, produciendo el coeficiente de agrupamiento promedio más alto (0.62 ± 0.02) de todos los experimentos de esta sección, mientras

que con las reglas $R2$ y $R3$, solo emerge un concentrador de grado 1499, tal y como se muestra en las Figuras 4.1a, 4.1b y 4.1c.

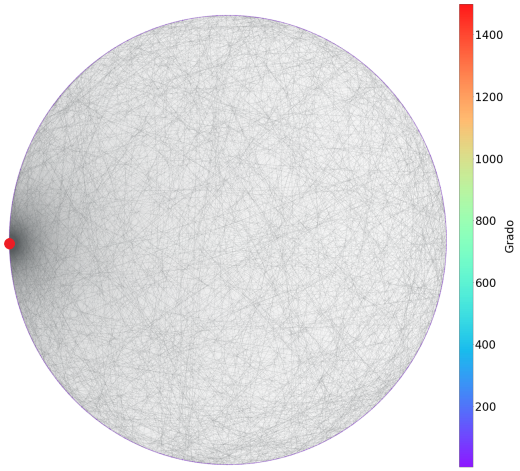
2. Las redes que resultan de los experimentos donde se usa en conjunto la regla $R1$ y el algoritmo Random-Walk (señalados con identificadores 38, 29, 2, 11 y 20, en la Tabla 4.1), cuentan con las más altas modularidades 0.96 ± 0 (lo cual, indica la emergencia de buenas particiones de comunidades), altos coeficientes de agrupamiento $[0.55, 0.56] \pm 0.01$, bajas asortatividades [-0.34,-0.31] (indicando que vértices de grado bajo tienden a conectarse con vértices de grado alto y que las redes son poco robustas), vértices concentradores locales, los más altos diámetros (en los experimentos de esta sección), los cuales, oscilan en $[63.7, 71.2] \pm 4.94$ y las redes menos robustas ante ataques secuenciales dirigidos y fallas aleatorias (con $\mu - A2TR$ ante ataques secuenciales dirigidos en $[0.02, 0.03] \pm 0.01$ y ante fallas aleatorias en $[0.09, 0.11] \pm 0.04$), en particular, se puede observar que, comparado con el $\mu - A2TR$ ante ataques secuenciales dirigidos y fallas aleatorias de la topología inicial (0.01 ± 0), no emergió gran mejora en la robustez de las redes, todo lo anterior, independiente del tamaño máximo de arista dinámica. Esto se da debido a que la regla $R1$ es "elitista", y en ella, los vértices siempre buscan conectarse con el vértice por el que pasaron más paquetes exploradores, provocando la emergencia de vértices concentradores locales, en este sentido, al usar Random-Walk como algoritmo de encaminamiento, los paquetes son libres de viajar en la red con una probabilidad uniforme, lo que origina que los vértices busquen conectarse con vértices cercanos a su entorno local (al ser por donde más veces pasan los paquetes), descartando aquellos que se encuentran en entornos lejanos. En la Figura 4.2a, se muestra la topología resultante del experimento $R1$, RW, D, donde puede observarse que, a pesar de que los vértices pueden alcanzar cualquier punto en la red, esto no se logró debido al efecto de "entroncarse" en regiones locales. En la Figura 4.2b, se muestra la partición óptima de comunidades (indicadas en diferentes colores), conformada por 37 elementos, de la red mostrada en la Figura 4.2a.



(a) Topología final $R1$, SP, D

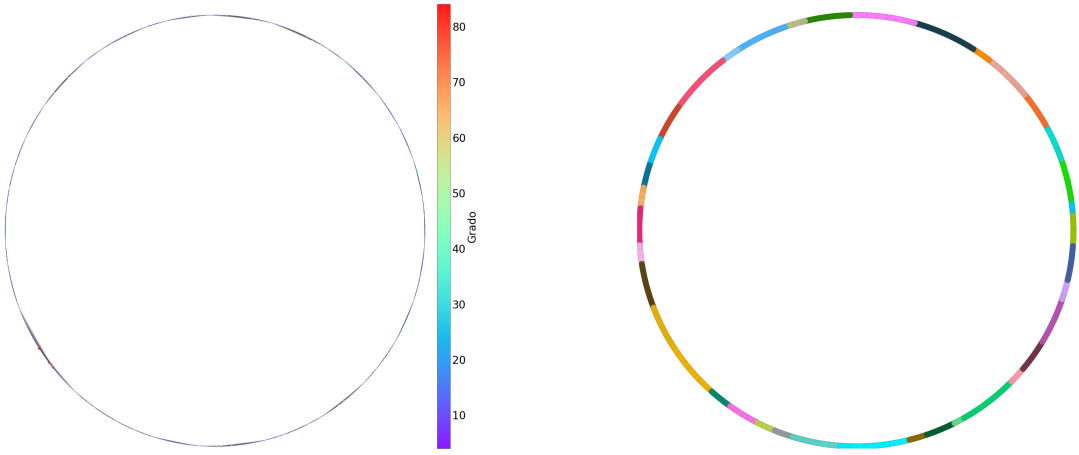


(b) Topología final $R2$, SP, D



(c) Topología final $R3$, SP, D

Figura 4.1: Topologías finales anillo SP, D



(a) Topología final $R1$, RW, D

(b) Partición óptima de comunidades

Figura 4.2: Topología y partición óptima de comunidades anillo $R1$, RW, D

3. Las redes que resultan de los experimentos donde se usa la regla $R3$, en conjunto con los algoritmos Random-Walk y Compass-Routing, y cualquier tamaño de arista dinámica, tienden a contar con los más bajos coeficientes de agrupamiento promedio (en los experimentos de esta sección), los cuales, oscilan en $[0.01, 0.12] \pm 0$ (señalados con identificadores 7, 16, 8, 26, 25, 17, 35, 44 y 34, en la Tabla 4.1), a excepción del experimento $R3$, CR, $D16$ (señalado con identificador 43, en la Tabla 4.1), donde el coeficiente de agrupamiento promedio resultó ser 0.24 ± 0.01, indicando que al usar el algoritmo Compass-Routing y un tamaño de arista dinámica $\frac{D}{16}$, se logra contrarrestar el efecto de emergencia de bajos coeficientes de agrupamiento promedio de la regla $R3$.

4. Las redes que resultan de los experimentos donde se usa la regla $R2$, en conjunto con los algoritmos Random-Walk y Compass-Routing, y cualquier tamaño de arista dinámica, tienden a contar con coeficientes de agrupamiento promedio que oscilan en $[0.14, 0.23] \pm 0.01$ (señalados con identificadores 4, 5, 14, 23, 13, 32, 22, 31, 40 y 41, en la Tabla 4.1).

5. Las redes que resultan de los experimentos donde se usa la regla $R1$, el algoritmo Compass-Routing y cualquier tamaño de arista dinámica, así como donde se usan las reglas $R2$ y $R3$, en conjunto con el algoritmo Shortest-Path y cualquier tamaño de arista dinámica, cuentan con coeficientes de agrupamiento promedio que oscilan en $[0.24, 0.42] \pm 0.02$ (señalados con identificadores 36, 27, 37, 1, 28, 19, 45, 18, 10, 42, 33, 24, 15, 6, y 9, en la Tabla 4.1).

6. Las redes que resultan de los experimentos donde se usa la regla $R1$, en conjunto con los algoritmos Random-Walk y Shortest-Path, y cualquier tamaño de arista dinámica, tienden a contar con los más altos coeficientes de agrupamiento promedio, los cuales, oscilan en $[0.47, 0.62] \pm 0.02$ (señalados con identificadores 39, 30, 21, 12, 20, 29, 11, 2, 38 y 3, en la Tabla 4.1).

7. Las redes que resultan de los experimentos donde se usa cualquier regla de conexión-reconexión, en conjunto con el algoritmo Shortest-Path y tamaños de arista dinámica D y $\frac{D}{2}$, tienden a contar con los más bajos diámetros, los cuales, oscilan en $[2.2, 6.1] \pm 0.49$ (señalados con identificadores 6, 9, 3, 12, 15 y 18, en la Tabla 4.1).

8. Las redes emergentes de los experimentos con configuraciones en las que intervienen las reglas $R2$ y $R3$, con los algoritmos Compass-Routing y Random-Walk y cualquier tamaño de arista dinámica, tienden a la no emergencia de vértices concentradores, en particular, el concentrador de menor grado (14) que emergió, se presentó en el experimento $R3$, Compass-Routing y tamaño de arista dinámica D (señalado con identificador 7, en la Tabla 4.1).

9. Las redes emergentes de los experimentos con configuraciones en las que interviene cualquier regla de conexión-reconexión, con el algoritmo Shortest-Path, tienden a la emergencia de vértices concentradores de grado en $[43, 1499]$, realizando un ordenamiento, primero, de manera ascendente de tamaño máximo de arista dinámica y, luego, de manera descendente de las tres reglas de conexión-reconexión. Más aún, experimen-

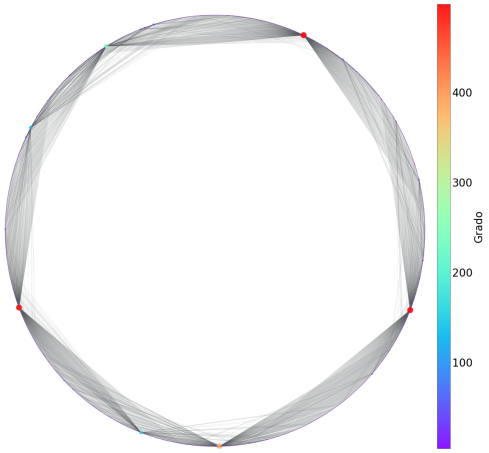


Figura 4.3: Polígono inscrito emergente en topología $R1$, SP, $\frac{D}{2}$

tos donde interviene la regla $R1$ y los algoritmos Compass-Routing y Random-Walk, con cualquier tamaño de arista dinámica, tienden, también, a la emergencia de vértices concentradores.

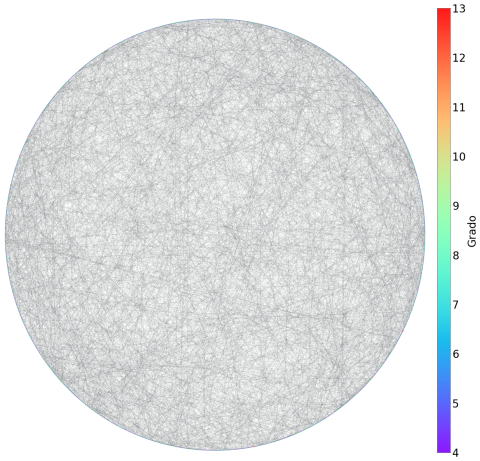
10. Las redes que resultan de los experimentos en los que se usa la regla $R3$, en combinación con el algoritmo Random-Walk y cualquier tamaño de arista dinámica (señalados con identificadores 17, 35, 8, 44 y 26, en la Tabla 4.1), así como donde se usa la regla $R3$, el algoritmo Compass-Routing y los tamaños de arista dinámica D y $\frac{D}{2}$ (señalados con identificadores 7 y 16, en la Tabla 4.1), tienden a estabilizar las medidas de sus propiedades estructurales en muy pocos ciclos de conexión-reconexión (a lo más 7).

11. Del experimento con configuración $R1$, Shortest-Path y tamaño de arista dinámica $\frac{D}{2}$ (señalado con identificador 12, en la Tabla 4.1), emergieron redes con un patrón de auto-organización muy bien definido, el cual, parece dirigirse a un polígono inscrito en la circunferencia que representa al anillo, formado con las aristas dinámicas de la red y estableciendo concentradores en los vértices de tal polígono. La topología final, se muestra en la Figura 4.3.

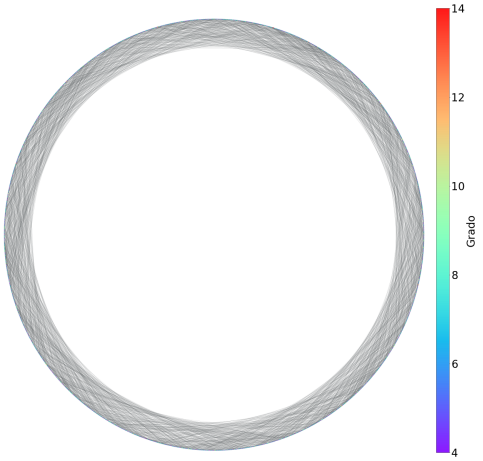
12. En todos los experimentos de esta sección, excepto en las configuraciones que usan la regla $R1$, el algoritmo Random-Walk y cualquier tamaño de arista dinámica, las desviaciones estándar de las medidas de las propiedades estructurales de las redes son muy pequeñas, al cabo de 10 ejecuciones de cada experimento, lo cual, indica que los resultados que se muestran en la Tabla 4.1, son confiables, y si se generaran ejecuciones de alguno de ellos, el resultado esperado se podrá verificar en dicha tabla.

13. Las redes más robustas ante ataques secuenciales dirigidos por grado, emergieron de los experimentos con configuraciones $R3$, Compass-Routing y tamaños de arista dinámica D y $\frac{D}{2}$ (señalados con identificadores 7 y 16, en la Tabla 4.1), con un $\mu - A2TR$ de 0.39 ± 0.01. Las redes, tienen un $A2TR(38) = 0.72$ y $A2TR(38) = 0.66$, respectivamente, lo cual, quiere decir que con tamaño de arista dinámica D, al cabo del 38 % de los vértices removidos, con base en ataques secuenciales dirigidos por grado, se tiene una probabilidad de comunicación del 72 % entre cualesquiera dos vértices que aún vivan en la red. Más aún, al usar la regla $R3$, la configuración que usa tamaño de arista dinámica D, estabiliza las medidas de las propiedades estructurales de las redes en solo 1 ciclo de conexión-reconexión, y en a lo más dos ciclos, en el caso del tamaño de arista dinámica $\frac{D}{2}$. En esta configuración, emergieron redes con un patrón de auto-organización muy bien definido, el cual, parece dirigirse a una corona circular, formada con las aristas dinámicas de la red. El tamaño de arista $\frac{D}{2}$, representa un muy buen compromiso entre alcance y las propiedades que emergen, esto puede aplicarse cuando el alcance implica un costo, por ejemplo, potencia de transmisión en redes de telecomunicaciones. Las topologías finales, se muestran en las Figuras 4.4a y 4.7a.

14. La red más robusta ante fallas aleatorias, emergió del experimento con configuración $R3$, Shortest-Path y tamaño de arista dinámica D (señalado con identificador 9, en la Tabla 4.1), con un $\mu - A2TR = 0.70 \pm 0.15$. La red, tiene un $A2TR(69) = 0.42$. Como se puede observar, a pesar de que la red es muy robusta ante fallas aleatorias, la desviación estándar de su $\mu - A2TR$ es alta, esto se da debido a que en dicha configuración emerge



(a) Topología final $R3$, CR, D



(b) Topología final $R3$, CR, $\frac{D}{2}$

Figura 4.4: Redes, en anillo, más robustas ante ataques secuenciales dirigidos por grado

un vértice concentrador de grado 1499, que mientras siga vivo en la red, permitirá la comunicación entre sus vértices, pero, cuando falla, la red se fragmenta fácilmente y sus vértices pierden comunicación tal y como puede observarse con la medida de su $A2TR(p)$, la cual, dice que la probabilidad de que cualesquiera dos vértices de la red, establezcan comunicación, en promedio, es 42 %. La topología final, se muestra en la Figura 4.1c. Las redes que siguen, en el sentido de mayor robustez ante fallas aleatorias, emergen de los experimentos con configuraciones $R3$, el algoritmo Compass-Routing y los tamaños de arista dinámica $\frac{D}{2}$, D y $\frac{D}{4}$, así como con la configuración $R2$, Shortest-Path y arista dinámica D (señalados con identificadores 16, 7, 25 y 6, en la Tabla 4.1), con $\mu - A2TR's = 0.69 \pm 0.02$, 0.68 ± 0.01, 0.68 ± 0.01 y 0.69 ± 0.18, respectivamente, no obstante, la red que emerge con $R2$, presenta las mismas condiciones que la referenciada con identificador 9, por lo que se considera que la red producida con $R3$, Compass-Routing y arista dinámica $\frac{D}{2}$, es la más robusta ante fallas aleatorias de esta sección.

15. Cada vértice $v \in V$, en la topología inicial, cuenta con las siguientes medidas:

a) Centralidad de intermediación $BC(v)$ (normalizada): 0.249833

b) Centralidad de cercanía $CC(v)$: 0.002664

c) Centralidad de grado $\delta(v)$: 2

Lo cual, indica que todos los vértices se encuentran en las mismas posibilidades de ser concentradores, no dando prioridades a alguno(s) de ellos. En particular, en los experimentos donde no emergen vértices super-concentradores (los cuales, usan las reglas $R2$ y $R3$ sin el algoritmo Shortest-Path), los vértices de mayor grado, se turnan en cada ciclo de conexión-reconexión, induciendo un mecanismo de control de congestión en la red.

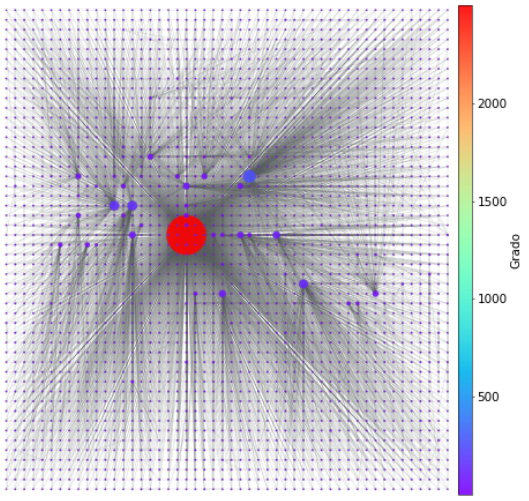
4.4.2. Malla 50 x 50 vértices cooperadores

Para la topología Malla 50 x 50 vértices, en experimentos cooperadores:

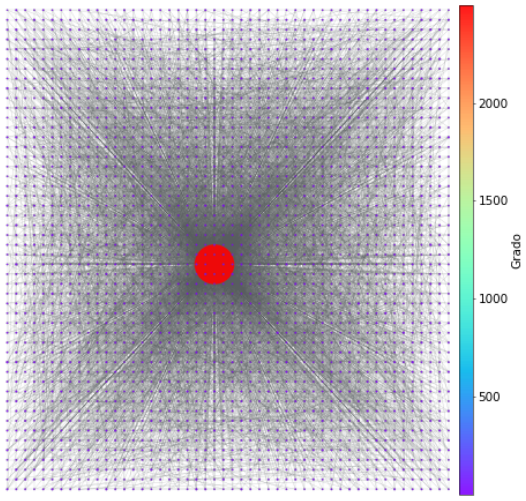
1. Las redes que resultan de los experimentos en los que se usa cualquier regla de conexión-reconexión y el algoritmo Shortest-Path, con tamaño de arista dinámica D (señalados con identificadores 3, 6 y 9, en la Tabla 4.2), tienden a que su diámetro se reduzca a dos unidades, indicando que todos los vértices conectan una de sus dos aristas dinámicas a un vértice concentrador de grado 2499. Para $R1$, $R2$ y $R3$, el diámetro se estabiliza en 2.4 ± 0.49, 2.9 ± 0.3 y 2.7 ± 0.46 unidades, al cabo de 10, 11 y 26 ciclos de experimentación, respectivamente, lo cual, quiere decir que no en todos los experimentos fue necesario terminar los 50 ciclos, debido a que una vez que el diámetro está establecido en dos unidades y se usa Shortest-Path como algoritmo de encaminamiento, ya no se permite que los vértices reconecten sus aristas dinámicas, dado que todo el tráfico de paquetes pasa a través del concentrador. Sin embargo, esto no es necesariamente bueno, debido a que puede que el concentrador, no tenga la capacidad de atender todas las peticiones de todos los vértices y, eventualmente, puede perder funcionalidad y salir de operación.

Más aún, al atacar, de manera dirigida por grado a la red, el concentrador sería el primer vértice en salir de operación. El experimento donde se usa la regla $R1$ (señalado con identificador 3, en la Tabla 4.2), provoca la emergencia de redes con un concentrador de grado 2499 y varios concentradores locales, produciendo un coeficiente de agrupamiento promedio alto (0.44 ± 0.02), mientras que con las reglas $R2$ y $R3$, solo emerge un concentrador de grado 2499, tal y como se muestra en las Figuras 4.5a, 4.5b y 4.5c.

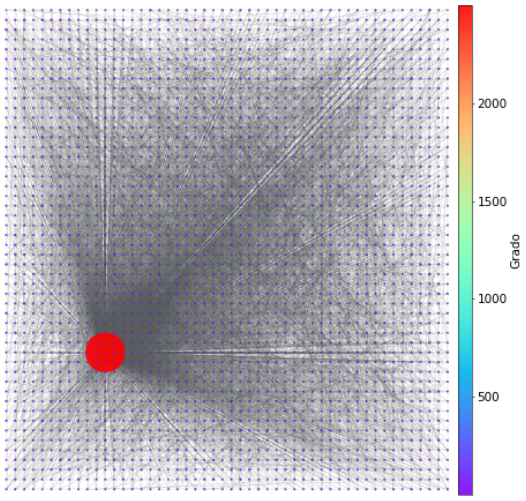
2. Las redes que resultan de los experimentos en los que se usa en conjunto la regla $R1$ y el algoritmo Random-Walk (señalados con identificadores 38, 29, 2, 11 y 20, en la Tabla 4.2), cuentan con las más altas modularidades 0.87 ± 0 (lo cual, indica la emergencia de buenas particiones de comunidades), los más altos coeficientes de agrupamiento (en los experimentos de esta sección), los cuales, oscilan en $[0.44, 0.46] \pm 0.01$, asortatividades negativas [-0.17,-0.13] (indicando que vértices de grado bajo tienden a conectarse con vértices de grado alto), vértices concentradores locales, diámetros que oscilan en $[19.1, 28.7] \pm 0.94$, un $\mu - A2TR$ ante ataques secuenciales dirigidos en $[0.35, 0.36] \pm 0.02$, el cual, comparado con el $\mu - A2TR$ de la topología inicial (0.33 ± 0), indica que no emergió gran mejora en la robustez de las redes ante ataques secuenciales dirigidos, así como los más bajos $\mu - A2TR's$ ante fallas aleatorias, los cuales, viven en $[0.59, 0.61] \pm 0.03$, todo lo anterior, independiente del tamaño máximo de arista dinámica. Esto se da debido a que la regla $R1$ es "elitista", y en ella, los vértices siempre buscan conectarse con el vértice por el que pasaron más paquetes exploradores, provocando la emergencia de vértices concentradores locales, en este sentido, al usar Random-Walk como algoritmo de encaminamiento, los paquetes son libres de viajar en la red con una probabilidad uniforme, lo que origina que los vértices busquen conectarse con vértices cercanos a su entorno local (al ser por donde más veces pasan los paquetes), descartando aquellos que se encuentran en entornos lejanos. En la Figura 4.6a, se muestra la topología de red resultante del experimento $R1$, RW, D, donde puede observarse que, a pesar de que los vértices pueden alcanzar cualquier punto en la red, esto no se logró debido al efecto de "entroncarse" en regiones locales. En la Figura 4.6b, se muestra la partición óptima



(a) Topología final $R1$, SP, D

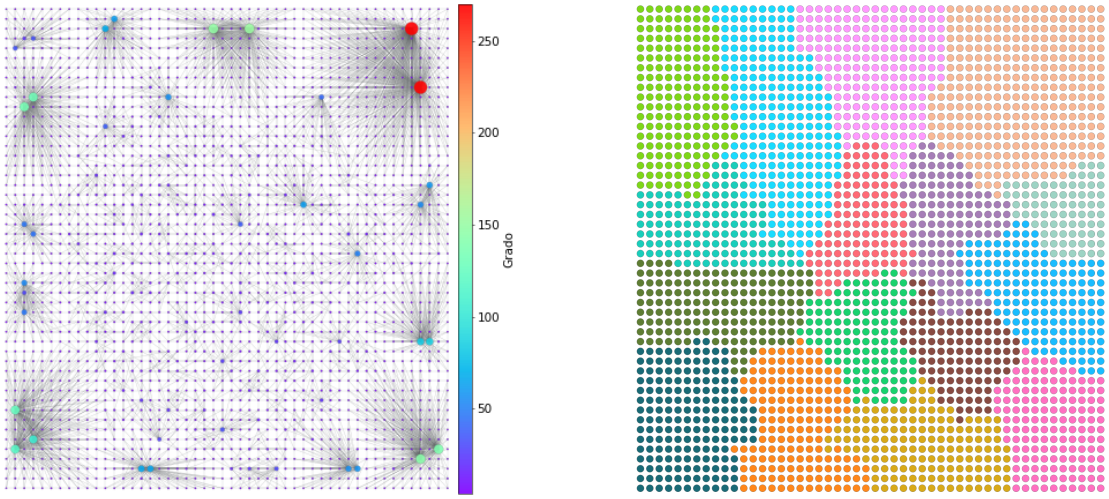


(b) Topología final $R2$, SP, D



(c) Topología final $R3$, SP, D

Figura 4.5: Topologías finales malla SP, D



(a) Topología final $R1$, RW, D

(b) Partición óptima de comunidades

Figura 4.6: Topología y partición óptima de comunidades malla $R1$, RW, D

de comunidades (indicadas en diferentes colores), conformada por 16 elementos, de la red mostrada en la Figura 4.6a.

3. Las redes que resultan de los experimentos en los que se usa la regla $R3$, en conjunto con los algoritmos Random-Walk y Compass-Routing, y tamaños de arista dinámica D, $\frac{D}{2}$, $\frac{D}{4}$ y $\frac{D}{8}$, tienden a contar con los más bajos coeficientes de agrupamiento promedio (en los experimentos de esta sección), los cuales, oscilan en $[0.01, 0.07] \pm 0$ (señalados con identificadores 7, 16, 8, 26, 25, 17, 35 y 34, en la Tabla 4.2), a excepción de los experimentos con configuraciones $R3$, $D16$ y algoritmos de encaminamiento Compass-Routing y Random-Walk (señalados con identificadores 43 y 44, en la Tabla 4.2), donde el coeficiente de agrupamiento promedio resultó ser 0.15 ± 0 y 0.14 ± 0, respectivamente, indicando que al usar dichas configuraciones, se logra contrarrestar el efecto de producción de bajos coeficientes de agrupamiento promedio de la regla $R3$.

4. Las redes que resultan de los experimentos donde se usa la regla $R1$, el algoritmo Random-Walk y cualquier tamaño de arista dinámica, en conjunto con aquellos donde se usa la regla $R1$ con el algoritmo Shortest-Path, y tamaños de arista dinámica D y $\frac{D}{2}$ (señalados con identificadores 20, 11, 2, 29, 38, 3 y 12, en la Tabla 4.2), tienden a contar con los más altos coeficientes de agrupamiento promedio, los cuales, oscilan en $[0.44, 0.46] \pm 0.02$.

5. Las redes que resultan de los experimentos donde se usa cualquier regla de conexión-reconexión, en conjunto con el algoritmo Shortest-Path y tamaños de arista dinámica D y $\frac{D}{2}$, tienden a contar con los más bajos diámetros, los cuales, oscilan en $[2.4, 4] \pm 0.49$ (señalados con identificadores 6, 9, 3, 12, 15 y 18, en la Tabla 4.2).

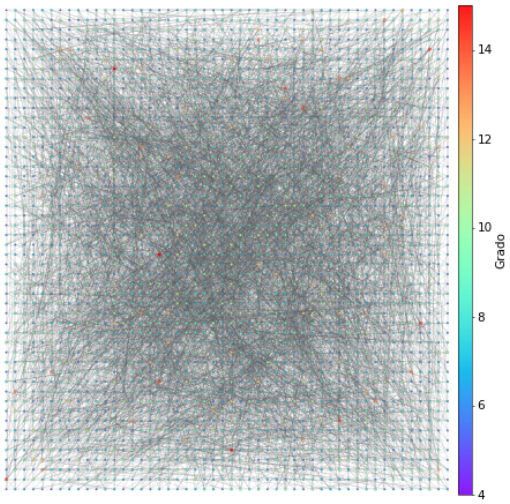
6. Las redes emergentes de los experimentos con configuraciones en las que intervienen las reglas $R2$ y $R3$, con los algoritmos Compass-Routing y Random-Walk y cualquier tamaño de arista dinámica, así como aquellos con configuraciones $R1$, Compass-Routing y tamaños de arista dinámica $\frac{D}{8}$ y $\frac{D}{16}$, tienden a la no emergencia de vértices concentradores, en particular, el concentrador de menor grado (16) que emergió, se presentó en los experimentos con configuraciones $R3$, Random-Walk y tamaño de arista dinámica $\frac{D}{16}$ y $R1$, Compass-Routing y tamaño de arista dinámica $\frac{D}{16}$ (señalados con identificadores 37 y 44, en la Tabla 4.2).

7. Las redes emergentes de los experimentos con configuraciones en las que interviene cualquier regla de conexión-reconexión, con el algoritmo Shortest-Path, tienden a la emergencia de vértices concentradores de grado en $[43, 2499]$, realizando un ordenamiento, primero, de manera ascendente de tamaño máximo de arista dinámica y, luego, de manera descendente de las tres reglas de conexión-reconexión. Más aún, experimentos donde interviene la regla $R1$, el algoritmo Random-Walk, con cualquier tamaño de arista dinámica y la regla $R1$, el algoritmo Compass-Routing y tamaños de arista dinámica D, $\frac{D}{2}$ y $\frac{D}{4}$, tienden, también, a la emergencia de vértices concentradores.

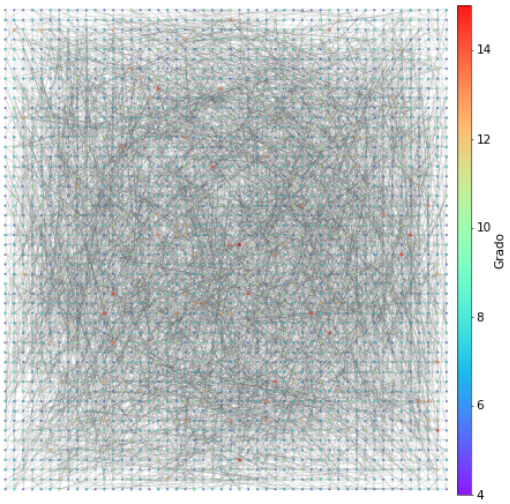
8. Las redes que resultan de los experimentos en los que se usa la regla $R3$, en combinación con el algoritmo Random-Walk y cualquier tamaño de arista dinámica (señalados con identificadores 17, 35, 8, 44 y 26, en la Tabla 4.2), así como donde se usa la regla $R3$, el algoritmo Compass-Routing y los tamaños de arista dinámica D, $\frac{D}{2}$ y $\frac{D}{4}$ (señalados con identificadores 7, 16 y 25, en la Tabla 4.2), estabilizan las medidas de sus propiedades estructurales en muy pocos ciclos de conexión-reconexión (a lo más 6).

9. En todos los experimentos de esta sección, las desviaciones estándar de las medidas de las propiedades estructurales de las redes son muy pequeñas, al cabo de 10 ejecuciones de cada experimento, lo cual, indica que los resultados que se muestran en la Tabla 4.2, son confiables, y si se generaran ejecuciones de alguno de ellos, el resultado esperado se podrá verificar en dicha tabla.

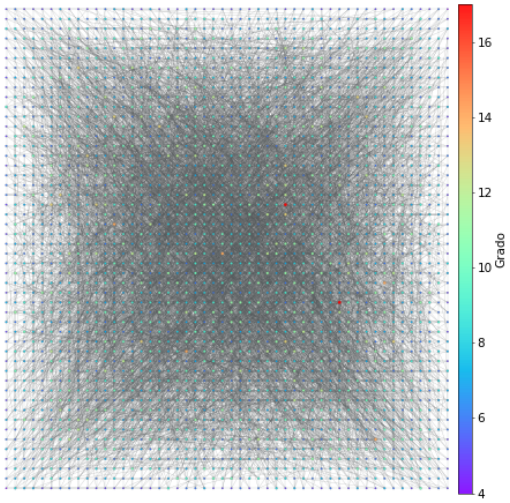
10. Las redes más robustas ante ataques secuenciales dirigidos por grado, emergieron de los experimentos con configuraciones $R3$, RW y arista dinámica D; $R3$, RW y arista dinámica $\frac{D}{2}$; $R3$, CR y arista dinámica D, así como $R2$, CR y arista dinámica D (señalados con identificadores 8, 17, 7 y 4, en la Tabla 4.2), con un $\mu - A2TR$ de 0.48 ± 0. Las redes, tienen $A2TR's(47) = 0.71, 0.65, 0.69y0.67$, respectivamente, lo cual, por ejemplo, para la configuración $R3$, RW y arista dinámica D, quiere decir que, al cabo del 47 % de los vértices removidos, con base en ataques secuenciales dirigidos por grado, se tiene una probabilidad de comunicación del 71 % entre cualesquiera dos vértices que aún vivan en la red. Note que la red con mayor $A2TR(p)$, justo al alcanzar el umbral $\mu - A2TR(47)$, resulta ser aquella con configuración $R3$, RW y arista dinámica D. La red, en segundo lugar, con respecto a su $A2TR(p)$, resulta ser aquella con configuración $R3$, CR y arista dinámica D al contar con $A2TR(47) = 0.69$. Más aún, esta configuración, estabiliza las medidas de sus propiedades estructurales en solo 1 ciclo de conexión-reconexión. Las topologías finales, de las cuatro configuraciones, se muestran en las Figuras 4.7a, 4.7b, 4.7c y 4.7d.



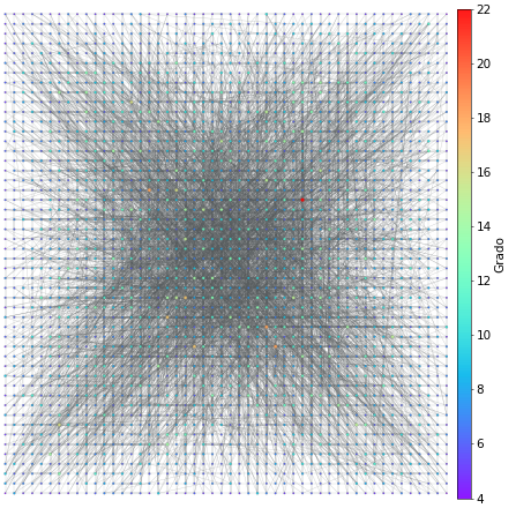
(a) Topología final $R3$, RW, D



(b) Topología final $R3$, RW, $\frac{D}{2}$



(c) Topología final $R3$, CR, D



(d) Topología final $R2$, CR, D

Figura 4.7: Redes, en malla, más robustas ante ataques secuenciales dirigidos por grado

11. Las redes más robustas ante fallas aleatorias, emergieron de los experimentos con configuraciones $R3$, CR y tamaño de arista dinámica D; $R2$, SP y tamaño de arista dinámica D, así como $R2$, SP y tamaño de arista dinámica $\frac{D}{2}$, (señalados con identificadores 7, 6 y 15, en la Tabla 4.2), con $\mu - A2TR's = 0.76 \pm 0.01, 0.78 \pm 0.11$ y 0.77 ± 0.09, respectivamente. Sus $A2TR's$, resultaron ser $0.65, 0.55$ y 0.56, respectivamente. Como se puede observar, a pesar de que las redes identificadas con identificadores 6 y 15 son muy robustas ante fallas aleatorias, la desviación estándar de su $\mu - A2TR$ es alta, esto se da debido a que en dichas configuraciones, emerge un vértice concentrador de grado 2499, para el caso de tamaño de arista dinámica D, y de grado 2474, para el caso de tamaño de arista dinámica $\frac{D}{2}$, que mientras sigan vivos en la red, permitirán la comunicación entre sus vértices, pero, cuando fallan, la red se fragmenta fácilmente y sus vértices pierden comunicación, por lo que se considera que la red producida con $R3$, Compass-Routing y arista dinámica D, es la más robusta ante fallas aleatorias de esta sección.

12. Las redes menos robustas ante ataques secuenciales dirigidos, se dan con las configuraciones $R1$, SP y tamaños de arista dinámica D y $\frac{D}{2}$ (señalados con identificadores 3 y 12, en la Tabla 4.2). Su $\mu - A2TR = 0.32 \pm 0.01$, que, comparado con el $\mu - A2TR$ de la red inicial (0.33 ± 0), disminuyó al producir vértices concentradores locales, los cuales, representan puntos clave de la red, y al removerlos, esta pierde robustez.

13. La topología malla, no resulta ser homogénea en términos de sus métricas de centralidad, tal y como se muestra en las Figuras 4.8 y 4.9. Lo cual, indica que los vértices no se encuentran en las mismas posibilidades de ser concentradores al usar el algoritmo Shortest-Path, dando prioridades a algunos de ellos, en particular, a los que se encuentran en el centro y su periferia, sin embargo, no todos los experimentos ejecutados con los tres algoritmo de encaminamiento, necesariamente dan tales prioridades. Observe, por ejemplo, las redes de las Figuras 4.5c y 4.6a.

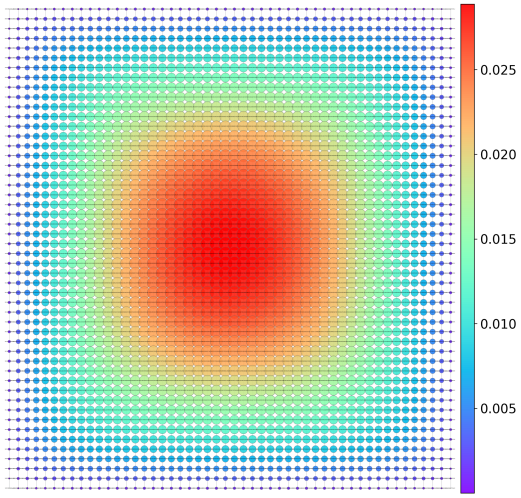


Figura 4.8: Centralidad de intermediación $BC(v)$ (normalizada)

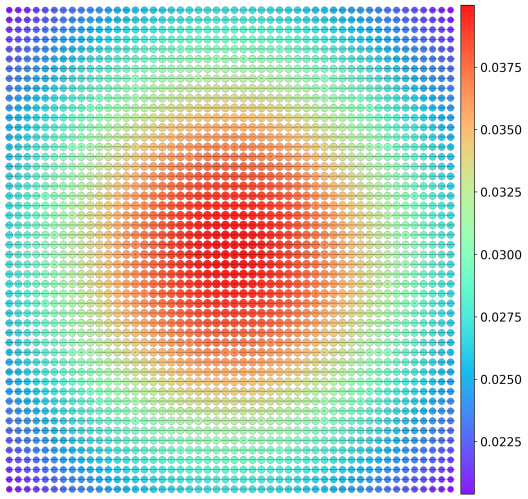


Figura 4.9: Centralidad de cercanía $CC(v)$

4.4.3. Anillo 1500 vértices semi-cooperadores

Para la topología Anillo con 1500 vértices, en experimentos semi-cooperadores:

1. En todos los experimentos de esta sección, los vértices de las redes hacen uso de sus capacidades al máximo, al aceptar todas las posibles conexiones entrantes de acuerdo a las restricciones $\frac{n}{4}$, $\frac{n}{16}$ y $\frac{n}{64}$.

2. Los coeficientes de agrupamiento promedio, marcan la misma tendencia que con los experimentos cooperadores, $R3$ crea los más bajos [0.06,0.15]±0.02, después $R2$ los incrementa un poco [0.19,0.3]±0.01, y los más altos emergen con $R1$ [0.33,0.43]±0.03.

3. La red más robusta ante ataques secuenciales dirigidos por grado, emergió del experimento con configuración $R3$, SP, tamaño máximo de arista dinámica D y restricción $\frac{n}{64}$ (señalado con identificador 9 en la Tabla 4.3), con un $\mu - A2TR = 0.31 \pm 0$ y un $A2TR(30) = 0.64$. Su topología final, se muestra en la Figura 4.10.

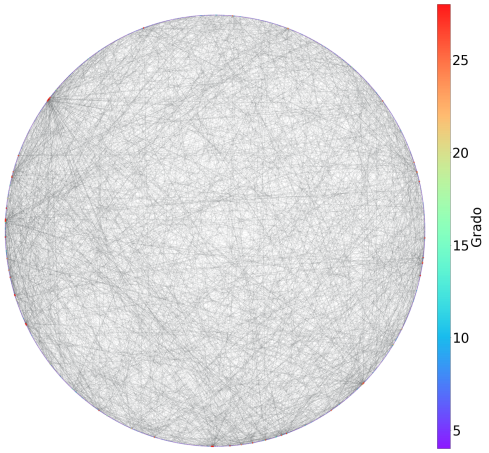


Figura 4.10: Red, en anillo, más robusta ante ataques secuenciales dirigidos por grado, segundo conjunto

4. Las redes más robustas ante fallas aleatorias, emergieron de los experimentos con configuraciones $R3$, SP, tamaño máximo de arista dinámica D y restricciones $\frac{n}{16}$ y $\frac{n}{64}$ (señalados con identificadores 7 y 9 en la Tabla 4.3), con un $\mu - A2TR = 0.68 \pm 0.03$ y $A2TR's(67) = 0.61$ y 0.58, respectivamente.

5. Al definir las restricciones $\frac{n}{4}$, $\frac{n}{16}$ y $\frac{n}{64}$, ninguno de los experimentos de esta sección fue capaz de producir redes tales que su diámetro sea reducido a dos unidades, debido a que esto solo se da si todos los vértices se conectan a un super-concentrador, lo cual, resulta imposible al definir tales restricciones.

6. Todos los experimentos de esta sección, producen redes con asortatividades en [-0.27,-0.1], lo cual, indica que al usar el algoritmo Shortest-Path, vértices de grado bajo tienden a conectarse con vértices de grado alto, en este caso, los concentradores que caracterizan al algoritmo Shortest-Path.

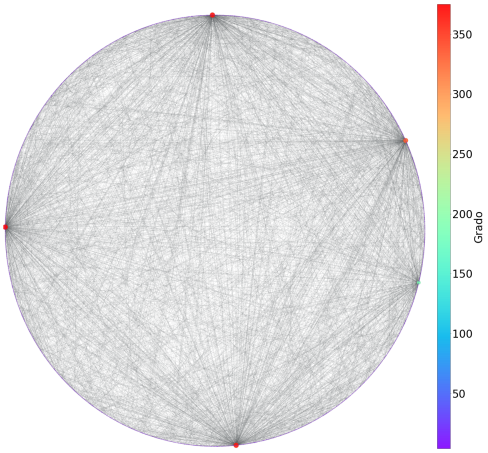
7. En todos los experimentos de esta sección donde se usan las reglas $R1$ y $R3$, con tamaño de arista dinámica D, el número máximo de concentradores emergentes en las redes es, a lo más, el denominador de las restricciones $\frac{n}{4}$, $\frac{n}{16}$ y $\frac{n}{64}$, tal y como se muestra en las Figuras 4.12 y 4.11. Al usar la regla $R2$, lo anterior no ocurre, ya que en la mayoría de los experimentos, el número de vértices concentradores emergentes es mayor al denominador de las tres restricciones, respectivamente.

8. En todos los experimentos de esta sección, las desviaciones estándar de las medidas de las propiedades estructurales son muy pequeñas, al cabo de 10 ejecuciones de cada experimento, lo cual, indica que los resultados que se muestran en la Tabla 4.3, son confiables, y si se generaran ejecuciones de alguno de ellos, el resultado esperado se podrá verificar en dicha tabla.

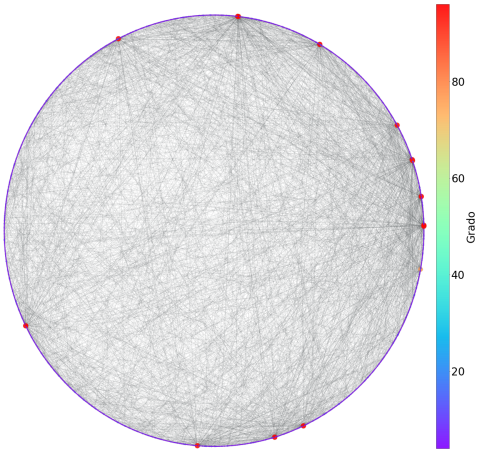
4.4.4. Malla 50 x 50 vértices semi-cooperadores

Para la topología Malla 50 x 50 vértices, en experimentos semi-cooperadores:

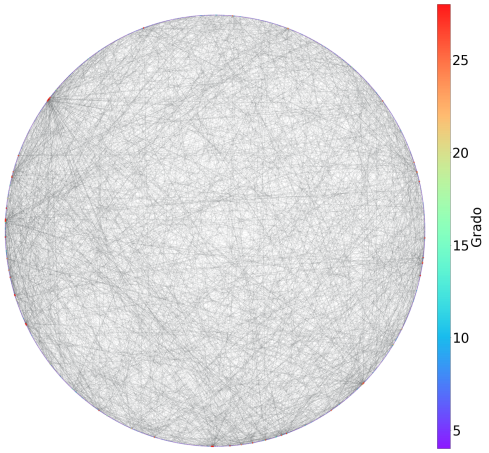
1. En todos los experimentos de esta sección, los vértices de las redes hacen uso de sus capacidades al máximo, al aceptar todas las posibles conexiones entrantes de acuerdo a las restricciones $\frac{n}{4}$, $\frac{n}{16}$ y $\frac{n}{64}$.



(a) Topología final $R3$, SP, D, $\frac{n}{4}$



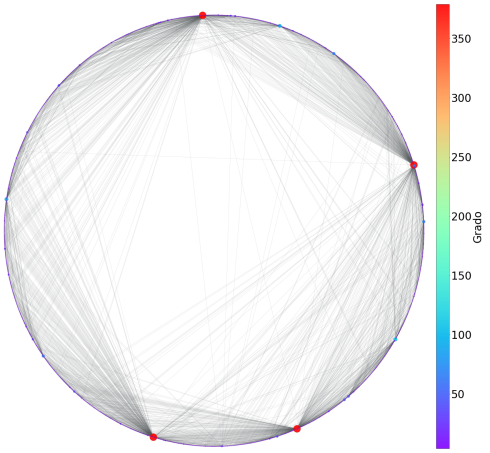
(b) Topología final $R3$, SP, D, $\frac{n}{16}$



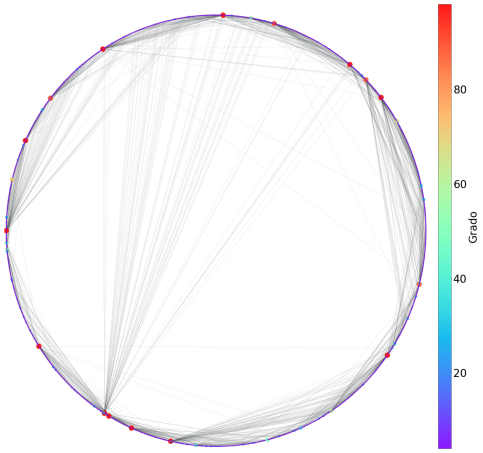
(c) Topología final $R3$, SP, D, $\frac{n}{64}$

Figura 4.11: Topologías finales anillo $R3$, SP, D, segundo conjunto

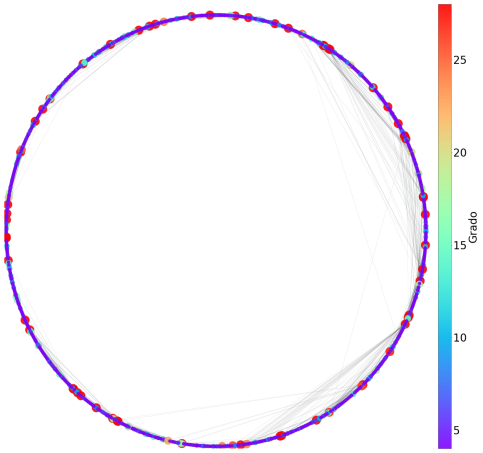
2. Los coeficientes de agrupamiento promedio, marcan la misma tendencia que con los experimentos cooperadores, $R3$ crea los más bajos [0.04,0.13]±0.01, después $R2$ los incrementa un poco [0.11,0.18]±0.01, y los más altos emergen con $R1$ [0.24,0.32]±0.02.



(a) Topología final $R1$, SP, D, $\frac{n}{4}$



(b) Topología final $R1$, SP, D, $\frac{n}{16}$

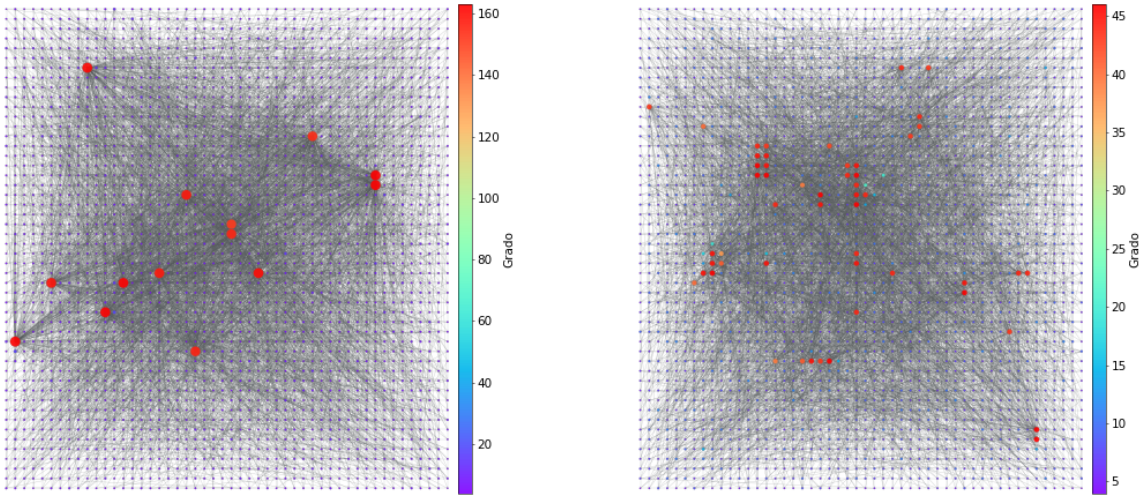


(c) Topología final $R1$, SP, D, $\frac{n}{64}$

Figura 4.12: Topologías finales anillo $R1$, SP, D, segundo conjunto

3. Las redes más robustas ante ataques secuenciales dirigidos por grado, emergieron de los experimentos con configuraciones $R3$, SP, tamaño máximo de arista dinámica D y restricciones $\frac{n}{16}$ y $\frac{n}{64}$ (señalados con identificadores 7 y 9 en la Tabla 4.4), con un

$\mu - A2TR = 0.43 \pm 0$ y $A2TR's(42) = 0.67$ y 0.7, respectivamente. Sus topologías finales, se muestran en la Figuras 4.13a y 4.13b.



(a) Topología final $R3$, SP, D, $\frac{n}{16}$

(b) Topología final $R3$, SP, D, $\frac{n}{64}$

Figura 4.13: Redes, en malla, más robustas ante ataques secuenciales dirigidos por grado, segundo conjunto

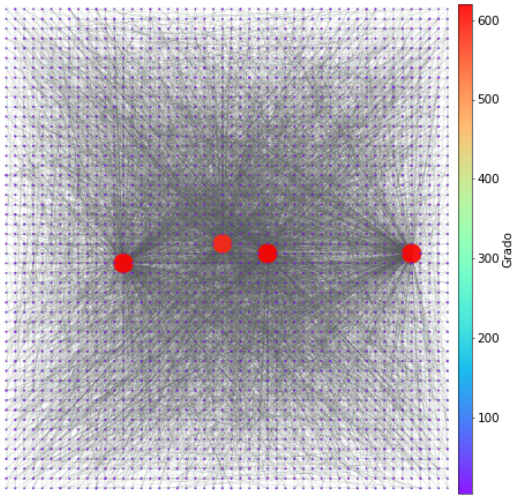
4. Al definir las restricciones $\frac{n}{4}$, $\frac{n}{16}$ y $\frac{n}{64}$, ninguno de los experimentos de esta sección fue capaz de producir redes tales que su diámetro sea reducido a dos unidades, debido a que esto solo se da si todos los vértices se conectan a un super-concentrador, lo cual, resulta imposible al definir tales restricciones.

5. Todos los experimentos de esta sección, producen redes con asortatividades en [-0.16,-0.04], lo cual, indica que al usar el algoritmo Shortest-Path, vértices de grado bajo tienden a conectarse con vértices de grado alto, en este caso, los concentradores que caracterizan al algoritmo Shortest-Path.

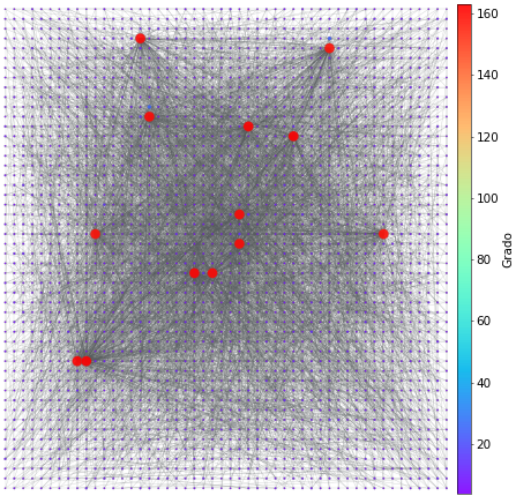
6. En todos los experimentos de esta sección donde se usan las reglas $R1$ y $R3$, con

tamaño de arista dinámica D, el número máximo de concentradores emergentes en las redes es, a lo más, el denominador de las restricciones $\frac{n}{4}$, $\frac{n}{16}$ y $\frac{n}{64}$, tal y como se muestra en las Figuras 4.14 y 4.15. Al usar la regla $R2$, lo anterior no ocurre, ya que en la mayoría de los experimentos, el número de vértices concentradores emergentes es mayor al denominador de las tres restricciones.

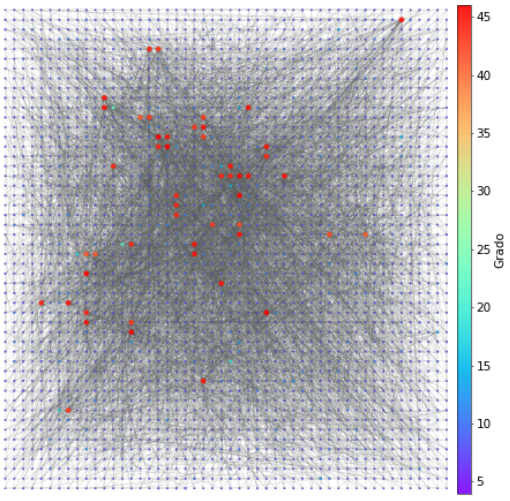
7. En todos los experimentos de esta sección, las desviaciones estándar de las medidas de las propiedades estructurales son muy pequeñas, al cabo de 10 ejecuciones de cada experimento, lo cual, indica que los resultados que se muestran en la Tabla 4.4, son confiables, y si se generaran ejecuciones de alguno de ellos, el resultado esperado se podrá verificar en dicha tabla.



(a) Topología final $R3$, SP, D, $\frac{n}{4}$

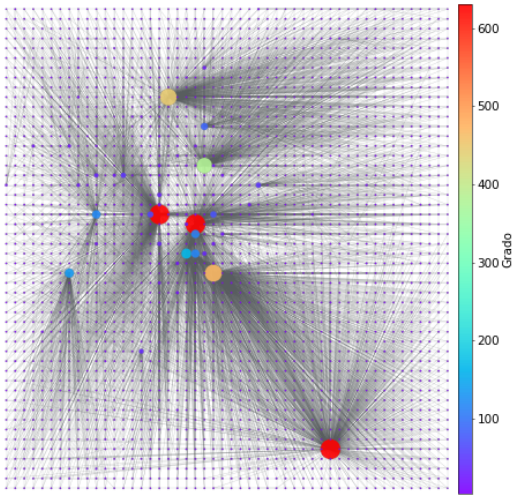


(b) Topología final $R3$, SP, D, $\frac{n}{16}$

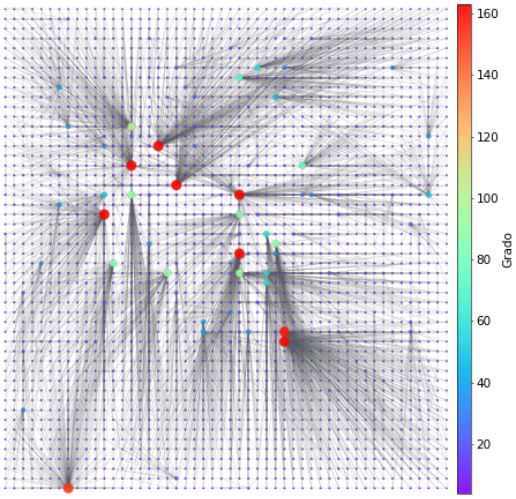


(c) Topología final $R3$, SP, D, $\frac{n}{64}$

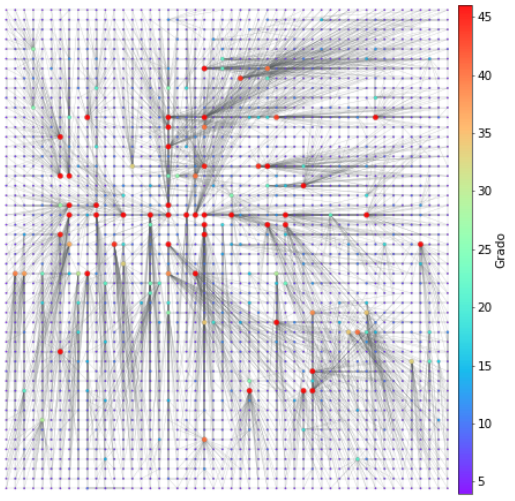
Figura 4.14: Topologías finales malla $R3$, SP, D, segundo conjunto



(a) Topología final $R1$, SP, D, $\frac{n}{4}$



(b) Topología final $R1$, SP, D, $\frac{n}{16}$



(c) Topología final $R1$, SP, D, $\frac{n}{64}$

Figura 4.15: Topologías finales malla $R1$, SP, D, segundo conjunto

Capítulo 5

Conclusiones y Trabajo Futuro

"Pienso que el siguiente siglo [XXI], será el siglo de la complejidad" -Stephen Hawking

5.1. Objetivos

El objetivo general de la realización de esta investigación, fue estudiar los efectos que emergen de la cooperación en la evolución estructural de un conjunto de redes, las cuales cuentan con topologías de redes regulares de malla y anillo, mediante un conjunto de experimentos en los que se configuraron diversos parámetros locales en cada uno de sus vértices, fueron capaces de adquirir diversas características presentes en los sistemas complejos. Bajo este contexto, se plantearon una serie de diversos objetivos particulares, los cuales, fueron cubiertos cabalmente durante el desarrollo de esta investigación, como sigue:

1) *Reconocer las principales propiedades estructurales que caracterizan el estado de una red.* Se abordó este objetivo, reconociendo las siguientes propiedades estructurales de las redes: modularidad, diámetro, longitud de trayectoria promedio, coeficiente de agrupamiento promedio, orden relativo del componente más grande, asortatividad y robustez ante fallas aleatorias y ante ataques secuenciales dirigidos, haciendo uso de ellas en ambas fases de la investigación.

2) *Reconocer parámetros locales de los que pueden emerger propiedades globales comúnmente encontradas en las redes complejas.* Se abordó este objetivo, reconociendo los parámetros locales denominados: algoritmo de encaminamiento, reglas locales de conexión-reconexión de aristas dinámicas, tamaño máximo de alcance de aristas dinámicas y limitaciones en las conexiones entrantes en cada vértice. Más aún, se probaron dichos parámetros en dos topologías de red distintas, una malla y un anillo (el cual, representa una gráfica simétrica).

3) *Proponer, al menos, un mecanismo que permita describir la formación de una red en el que intervengan vértices cooperadores y semi-cooperadores.* Se abordó este objetivo, definiendo un conjunto de experimentos semi-cooperadores, en los cuales, se tienen tres diferentes restricciones acerca del número máximo de conexiones que los vértices de las redes pueden aceptar.

4) *Desarrollar una herramienta de simulación sobre la cual se ejecutarán los experimentos que permitirán caracterizar la evolución estructural de las redes propuestas.* Se abordó este objetivo, escribiendo un simulador de eventos discretos, el cual, con base en diversos scripts del lenguaje de programación python, permite la emergencia de un conjunto de redes con características análogas a las encontradas en las redes complejas.

5) *Caracterizar la evolución estructural de un conjunto de redes mientras son sometidas a diversos procesos de conexión y reconexión de aristas.* Se abordó este objetivo, definiendo dos diferentes conjuntos de experimentos: cooperadores y semi-cooperadores, los cuales, fueron ejecutados llevando a cabo la conexión y reconexión de las aristas de cada vértice en las redes durante cincuenta ciclos de experimentación, considerando diez ejecuciones de cada uno de ellos.

6) *Evaluar la robustez de las redes obtenidas mientras son sometidas a diversos procesos de degradación.* Se abordó este objetivo, llevando a cabo la degradación de cada una de las redes resultantes en la etapa de formación de redes complejas, considerando diez

ejecuciones de cada proceso de degradación para cada una de ellas. Más aún, para llevar a cabo la evaluación de la robustez, se usaron tres métricas distintas: orden relativo del componente más grande, fiabilidad promedio entre dos terminales y media de la fiabilidad promedio entre dos terminales.

5.2. Aportaciones

Las aportaciones que nacen a partir de esta investigación, son definidas a continuación:

1. Un simulador de eventos discretos que la comunidad interesada puede usar libremente en búsqueda de pruebas y mejoras.

2. Una serie de mecanismos de formación de redes complejas, que se pueden aplicar a diversas redes de telecomunicaciones, por ejemplo, redes de sensores inalámbricos o redes par a par, no descartando que dichos mecanismos puedan ser aplicables a diversas áreas del conocimiento.

5.3. La gran sorpresa

En los experimentos propuestos en esta investigación, *una **exploración** desinformada (donde interviene completamente el azar), o semi-informada, y una **selección** desinformada o basada totalmente en azar; ambas totalmente distribuidas, producen los mejores resultados en términos de robustez y, por otro lado, muy buenas propiedades estructurales de las redes emergentes (sin la emergencia de super-concentradores) tal que son análogas a las encontradas en las redes complejas; comparadas con una exploración y selección informadas.*

5.4. Trabajo futuro

Esta investigación, abre puertas a distintos trabajos futuros, entre ellos:

1. Determinar modelos matemáticos para las diversas propiedades de las redes (coeficiente de agrupamiento promedio, longitud de trayectoria promedio, diámetro) con base en los distintos parámetros considerados en los experimentos, así como validar si sus distribuciones de grados se ajustan a alguna función de distribución de probabilidad conocida.

2. Determinar cotas de complejidad para cada uno de los algoritmos correspondientes a las fases de exploración, negociación y conexión-reconexión.

3. Determinar el número óptimo de paquetes exploradores que se deben enviar en la fase de exploración, con base en el orden de las topologías subyacentes en las redes.

4. Ejecutar los mecanismos de formación de redes complejas combinando en los experimentos diversos algoritmos de encaminamiento.

5. Ejecutar los mecanismos de formación de redes complejas combinando en los experimentos diversas reglas de conexión-reconexión de aristas dinámicas.

6. Ejecutar los mecanismos de formación de redes complejas combinando en los experimentos diversas longitudes de aristas dinámicas.

7. En esta investigación, se trabajó con topologías de red que se pueden perfectamente empotrar en el espacio Euclidiano, no obstante, se podrían realizar las modificaciones correspondientes al simulador de eventos discretos para poder trabajar con topologías que vivan en otro espacio, por ejemplo $\mathbb{R}^3$.

8. Degradar a las redes obtenidas en la fase de formación de redes complejas con base en diversas métricas de centralidad, por ejemplo, centralidad de intermediación y centralidad de cercanía ya que, en esta investigación, fueron degradadas con base en fallas aleatorias y ataques secuenciales dirigidos por grado.

Apéndice A

Conceptos básicos de teoría de gráficas

A.1. Introducción

El estudio de las redes, en la forma de teoría matemática de gráficas, es uno de los pilares fundamentales de las matemáticas discretas. La teoría de gráficas, tiene sus raíces en 1736. En esa época, la ciudad de Königsberg, Rusia, tenía 7 puertos conectando las orillas del río Pregel con dos grandes islas. El problema de los siete puentes de Königsberg, consistía en decidir si es posible seguir un camino que cruce cada puente solo una vez y volver al punto de origen del camino. Leonhard Euler, mapeó el problema en una gráfica, dando origen a lo que, actualmente, se conoce como teoría de gráficas, demostrando que no existe tal *ciclo Euleriano* [7]. La solución a este problema es, a menudo, citada como la primera prueba real en la teoría de gráficas y, durante el siglo XX, dicha teoría se ha convertido en un importante cuerpo de conocimientos [13]. En México, la teoría de gráficas fue impulsada por Víctor Neumann-Lara, quien no solo fue pionero en esta rama, sino que organizó el Coloquio de Teoría de las Gráficas, Combinatoria y sus Aplicaciones. En México, los investigadores llaman gráficas a los grafos, ya que Víctor Neuman-Lara los describía como "unos objetos de una belleza tal que debían tener un nombre femenino". Esto ha generado cierta confusión en el país ya que la palabra gráfica también nos refiere a la representación de datos o funciones [35]. La teoría de gráficas, resulta útil para analizar, comprender y cuantificar las propiedades de las redes,

ya que, a través de las gráficas, se pueden representar a las redes y, además, las interacciones existentes entre sus componentes.

A.2. Definiciones básicas

Esta sección, introduce notación, terminología y definiciones de la teoría de gráficas que son frecuentemente usadas en la descripción de las redes [1, 3, 4, 5, 7].

Definición A.2.1: Una *gráfica (o gráfica no dirigida)* G, es un objeto matemático constituido por un conjunto V de *vértices (o nodos)* y un conjunto $E \subset (V \times V)$ de *aristas (o arcos)*. Cada arista $e \in E$ se asocia con una pareja no ordenada de vértices. Si existe una arista única e asociada con los vértices $u, v \in V$, se escribe $e = (u, v)$ o $e = (v, u)$. Cada arista en una gráfica no dirigida, representa una relación bidireccional entre los vértices que asocia.

Definición A.2.2: Una *gráfica dirigida (o digráfica)* G, es un objeto matemático constituido por un conjunto V de *vértices (o nodos)* y un conjunto $E \subset (V \times V)$ de *aristas (o arcos)*. Cada arista $e \in E$ está asociada con una pareja ordenada de vértices $(u, v) \in V$. Si existe una arista única e asociada con la pareja ordenada (u, v) de vértices, se escribe $e = (u, v)$, lo cual, denota una arista que va de u a v, estableciendo una relación unidireccional entre ambos vértices.

En las definiciones A.2.1 y A.2.2, se asume que los conjuntos V y E son no vacíos y, además, son *conjuntos finitos*, es decir, son tales que pueden ponerse en correspondencia biunívoca con el conjunto $\{1, 2, 3, ..., i\}$, para algún $i \in \mathbb{N}$.

Definición A.2.3: Sea $e = (u, v)$ una arista en una gráfica no dirigida. Se dice que e es *incidente sobre* u y v, u y v son *incidentes sobre* e y, además, u y v son *vértices adyacentes*. Por otro lado, sea $e = (u, v)$ una arista en una gráfica dirigida. Se dice que e es *incidente*

sobre v, v es *incidente sobre* e y, además, v es *adyacente* a u.

En general, las gráficas se ilustran como imágenes en el plano, donde sus vértices, son representados como círculos con un identificador $id \in \mathbb{N}$ dentro de ellos y sus aristas, como líneas que unen parejas de vértices adyacentes. En una gráfica dirigida, las aristas se representan como flechas.

Ejemplo A.2.1: La Figura A.1 muestra una gráfica no dirigida G consistente en el conjunto de vértices:

$$V = \{1, 2, 3, 4, 5, 6\}$$

y en el conjunto de aristas:

$$E = \{e_1, e_2, e_3, e_4, e_5, e_6, e_7, e_8, e_9\}$$

La arista e_1, se asocia con la pareja no ordenada de vértices $(1, 2)$, dicha arista se denota por $e_1 = (1, 2)$ o $e_1 = (2, 1)$, además se dice que la arista e_1 es incidente sobre los vértices 1 y 2 y que ellos son adyacentes.

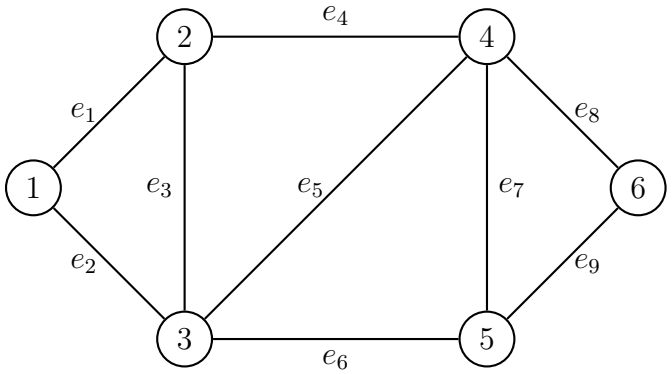


Figura A.1: Gráfica no dirigida

Ejemplo A.2.2: La Figura A.2 muestra una gráfica dirigida G. La arista e_1 se asocia con la pareja ordenada de vértices $(1, 2)$, dicha arista se denota por $e_1 = (1, 2)$.

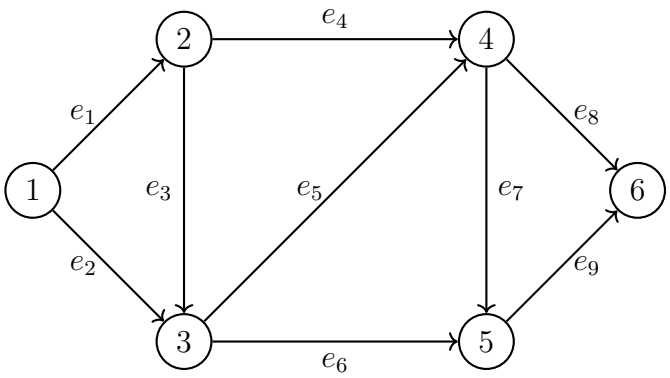


Figura A.2: Gráfica dirigida o digráfica

Definición A.2.4: Si G es una gráfica (no dirigida o dirigida) con vértices V y aristas E, se denota $G = (V, E)$.

Definición A.2.5: Sean $u, v \in V$. Una *arista simple*, es aquella que une a u con v tal que $u \neq v$.

Definición A.2.6: Sean $u, v \in V : u \neq v$. Si existe más de una arista $e = (u, v) \in E$, ellas se llaman *aristas múltiples*.

Ejemplo A.2.3: La Figura A.3, muestra una gráfica G, consistente en el conjunto de vértices $V = \{1, 2\}$, tal que los vértices contenidos en V están unidos por un par de aristas múltiples e_1 y e_2.

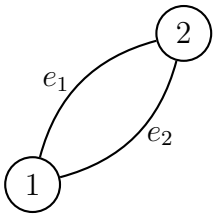


Figura A.3: Ejemplo de aristas múltiples

Definición A.2.7: Sea $v \in V$. Un *lazo o bucle*, es aquella arista de la forma $e = (v, v)$.

Ejemplo A.2.4: La Figura A.4 muestra un vértice que se conecta a el mismo a través

de un *lazo o bucle* $e_1 = (1, 1)$.

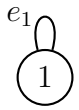


Figura A.4: Ejemplo de un lazo o bucle

Definición A.2.8: Se dice que G es una *gráfica simple* si no contiene aristas múltiples ni lazos.

Definición A.2.9: Si $V = \emptyset$, G se llama *gráfica nula*. Por otro lado, si $V \neq \emptyset$ y $E = \emptyset$, G se llama *gráfica vacía*.

Definición A.2.10: El número de vértices de G, denotado $n = |V|$, es denominado *orden de la gráfica*. De la Figura A.1, $n = 6$.

Definición A.2.11: El número de aristas de G, denotado $m = |E|$, es denominado *tamaño de la gráfica*. De la Figura A.1, $m = 9$.

Definición A.2.12: Sea $v \in V$. Se dice que v es un *vértice aislado* si no existe ninguna arista en la que v sea incidente.

Definición A.2.13: Sea G una gráfica no dirigida. Se define al *grado de un vértice* $v \in V$, denotado $\delta(v)$, como el número de aristas incidentes en v. De la Figura A.1, $\delta(4) = 4$.

Definición A.2.14: Sea G una gráfica no dirigida. Se define a la *secuencia de grados* de G como: $S_G = \{\delta(v_1), \delta(v_2), \delta(v_3), ..., \delta(v_n) \mid v_i \in V\}$. De la Figura A.1, $S_G = \{2, 3, 4, 4, 3, 2\}$.

Definición A.2.15: Dada la secuencia de grados de una gráfica no dirigida G, se define al *grado medio* de G, denotado $\langle\delta\rangle$, como sigue:

$$\langle \delta \rangle = \frac{1}{n} \sum_{i=1}^{n} \delta(v_i) = \frac{2m}{n} \tag{A.1}$$

De la Figura A.1, $\langle \delta \rangle = \frac{18}{6} = 3$.

Definición A.2.16: Para una gráfica no dirigida G, se define:

$$P(k) = \frac{N_k}{n} \tag{A.2}$$

Donde N_k representa el número de vértices en G que tienen grado k, como la probabilidad de que un vértice elegido al azar en G tenga grado k. Debido a que $P(k)$ es una probabilidad, debe de ser normalizada como sigue:

$$\sum_{k=1}^{\infty} P(k) = 1 \tag{A.3}$$

Ejemplo A.2.5: La Tabla A.1 muestra la distribución de grados de la Figura A.1. Como se puede observar, $\sum_{k=1}^{\infty} P(k) = 1$, lo cual, indica que dicha distribución está normalizada.

Tabla A.1: Distribución de grados de la gráfica mostrada en la Figura A.1

k	Frecuencia absoluta	$P(k)$
2	2	$\frac{2}{6}$
3	2	$\frac{2}{6}$
4	2	$\frac{2}{6}$

Definición A.2.17: Sea G una gráfica dirigida. Se define al *grado entrante* de un vértice $v \in V$, denotado $\delta_{in}(v)$, como el número de aristas entrantes en v. De la Figura A.2, $\delta_{in}(4) = 2$.

Definición A.2.18: Sea G una gráfica dirigida. Se define al *grado saliente* de un vértice $v \in V$, denotado $\delta_{out}(v)$, como el número de aristas salientes de v. De la Figura A.2, $\delta_{out}(4) = 2$.

Definición A.2.19: Sean $i \in \mathbb{N}$, $\{v_1, v_2, ..., v_i\} \subset V$ y $\{e_1, e_2, ..., e_{i-1}\} \subset E$. Una secuencia de vértices y aristas de la forma $v_1, e_1, v_2, e_2, ..., e_{i-1}, v_i$ denota un *camino* que va del vértice v_1 al vértice v_i tal que la arista e_i es incidente sobre los vértices v_i, v_{i+1}.

Definición A.2.20: Sea $S = \{v_1, e_1, v_2, e_2, ..., e_{i-1}, e_i\}$ un camino. Se dice que S es un *camino simple* si todas sus aristas son distintas.

Definición A.2.21: Sea $S = \{v_1, e_1, v_2, e_2, ..., e_{i-1}, e_i\}$ un camino simple. Se dice que S es una *trayectoria o ruta* si todos sus vértices son distintos.

Definición A.2.22: Sea $v \in V$. El conjunto de vértices vecinos de v, denotado $N(v)$, se define como $N(v) = \{u \in V \mid u \neq v, \exists e \in E : e = (v, u)\}$.

Definición A.2.23: Se dice que G es una *gráfica conexa* si $\forall\ (u, v) \in V, u \neq v$ existe un camino que va de u a v.

Definición A.2.24: Se dice que G es una *gráfica conectada* si es conexa.

Definición A.2.25: Se dice que G es una *gráfica $p-$regular* si $\forall\ v \in V, \delta(v) = p$.

Definición A.2.26: Sean G una gráfica no dirigida, $z \in \mathbb{N}$. Se dice que G es una *gráfica completa*, denotada K_z, si $n = z$ y, además, cada pareja de vértices en V es adyacente.

Definición A.2.27: Si G es una gráfica completa, su tamaño m está dado por:

$$m = \frac{n(n-1)}{2} \tag{A.4}$$

Ejemplo A.2.5: La Figura A.5 muestra una gráfica completa K_8 con $m = 28$.

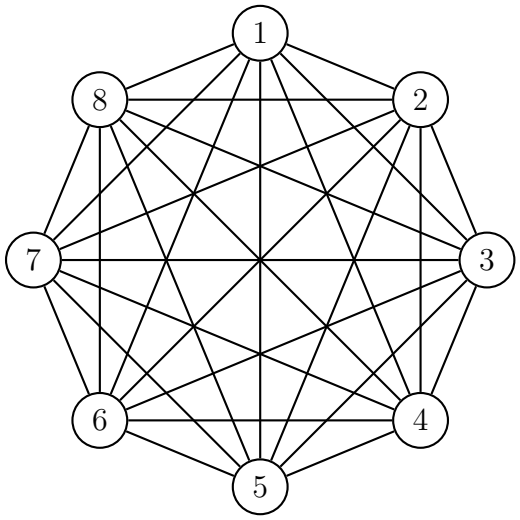


Figura A.5: Gráfica completa K_8

Definición A.2.29: Se dice que G es una *gráfica ponderada* si existe una función $w : E \to \mathbb{R}^+$ tal que asigna a cada arista $e \in E$ un valor numérico llamado *peso de la arista*. Si dicha función no está definida, se asume que se trata de una *gráfica no ponderada*.

Definición A.2.30: En una gráfica no ponderada, la *longitud* de un camino, camino simple o una trayectoria es igual a su número de aristas.

Definición A.2.31: En una gráfica ponderada, la *longitud* de un camino, camino simple o una trayectoria es igual a la suma de los pesos de sus aristas.

Definición A.2.32: Sean $u, v \in V$. La distancia entre u y v, denotada $d(u, v)$, se define como la longitud de la trayectoria más corta entre ellos.

Definición A.2.33: Un *árbol* $T = (V, E)$ es una gráfica simple y conectada en la que existe un solo camino entre cualesquiera dos de sus vértices.

Definición A.2.34: Sea G una gráfica no dirigida. El *diámetro* de G, denotado D_G, se define como: $D_G = max\{d(u, v) \mid u, v \in V, u \neq v\}$.

Definición A.2.35: Sea G una gráfica no dirigida. La *longitud de trayectoria promedio* de G, denotada LTP_G, está dada por:

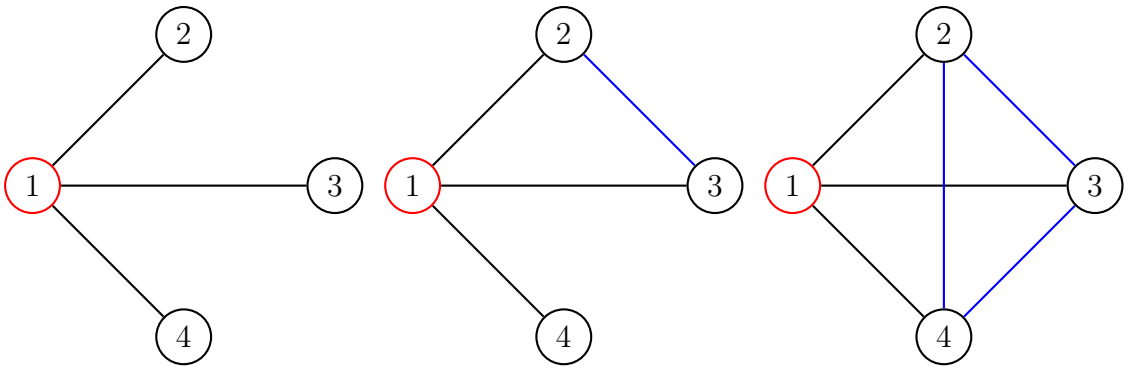
$$LTP_G = \sum_{u,v \in V, u \neq v} \frac{d(u,v)}{n(n-1)} \tag{A.5}$$

Definición A.2.36: Sean G una gráfica no dirigida, $v \in V$. El *coeficiente de agrupamiento* de v, denotado c_v, está dado por:

$$c_v = \frac{2T(v)}{\delta(v)(\delta(v)-1)} \tag{A.6}$$

donde $T(v) = |\{e = (u,w) \mid u, w \in N(v), u \neq w\}|$.

Ejemplo A.2.6: La Figura A.6 muestra tres posibles cálculos del coeficiente de agrupamiento del vértice con identificador 1, para tres diferentes gráficas.



(a) $c_1 = 0$ (b) $c_1 = \frac{1}{3}$ (c) $c_1 = 1$

Figura A.6: Ejemplo del cálculo del coeficiente de agrupamiento de un vértice en tres gráficas distintas

Definición A.2.37: Sea G una gráfica no dirigida. El *coeficiente de agrupamiento promedio* de G, denotado C_G, está dado por:

$$C_G = \frac{1}{n} \sum_{v \in V} c_v \tag{A.7}$$

Definición A.2.38: Se define a una *subgráfica* $G' = (V', E')$ de una gráfica $G = (V, E)$, cuando $V' \subset V, V' \neq \emptyset$ y $E' \subset E$. Si G' contiene a todas las aristas que unen a dos vértices de V', se dice que G' es una *subgráfica inducida* por V' y se denota como $G' = G[V']$.

Definición A.2.39: Sea G una gráfica no dirigida. Considere una partición de V en conjuntos no vacíos $V_1, V_2, ..., V_i$ tal que los vértices u y v están conectados si y solo si ambos pertenecen al mismo conjunto V_i. Las subgráficas $G[V_1], ..., G[V_i]$ se denominan los *componentes* de G.

Definición A.2.40: Sea $R = \{G[V_1], G[V_2], ..., G[V_i]\}$ el conjunto de los componentes de una gráfica no dirigida G. Se define al *componente más grande* de G como el componente de mayor orden contenido en R.

Ejemplo A.2.7: La Figura A.7 muestra una gráfica no dirigida G consistente en el conjunto de vértices:

$$V = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13\}$$

y en el conjunto de aristas:

$$E = \{e_1, e_2, e_3, e_4, e_5, e_6, e_7, e_8, e_9, e_{10}, e_{11}\}$$

Tal que su conjunto de componentes es:

$$R = \{\{1\}, \{2, 4, 7\}, \{3, 5, 6, 8, 9\}, \{10, 11, 12, 13\}\}$$

y su componente más grande está dado por el conjunto $\{3, 5, 6, 8, 9\} \subset R$.

Definición A.2.41: Sea G una gráfica no dirigida. La *densidad* de G, denotada d_G, se define como sigue:

$$d_G = \frac{2m}{n(n-1)} \tag{A.8}$$

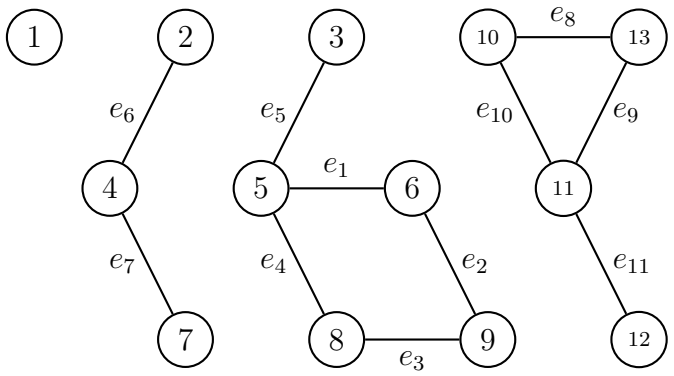


Figura A.7: Gráfica no dirigida dividida en componentes

Definición A.2.42: Sea G una gráfica no dirigida. Se define a una *comunidad*, como una subgráfica conexa de G, con vértices densamente conectados entre sí, y escasamente conectados con vértices de otra comunidad. La densidad de una comunidad debe ser mayor que la densidad de G.

Ejemplo A.2.8: La Figura A.8 muestra una gráfica no dirigida G consistente en el conjunto de vértices:

$$V = \{1, 2, 3, 4, 5, 6, 7, 8\}$$

y en el conjunto de aristas:

$$E = \{e_1, e_2, e_3, e_4, e_5, e_6, e_7, e_8, e_9, e_{10}, e_{11}\}$$

Particionada de una manera tal que los vértices en diferentes colores de G denotan diferentes comunidades.

La densidad de ambas comunidades es $\frac{5}{6}$, mientras que la densidad de G es $\frac{11}{28}$, como se puede observar, la densidad de ambas comunidades es mayor que la densidad de G y ambas comunidades son subgráficas conexas contenidas en G.

Se sabe que las redes aleatorias tienen una distribución de grados Binomial y, por lo

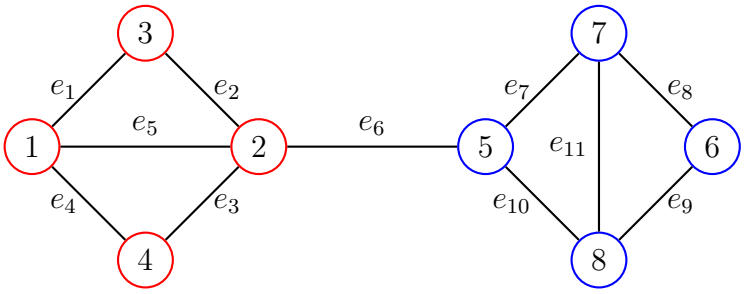


Figura A.8: Gráfica no dirigida dividida en comunidades

tanto, estas redes no exhiben una estructura de comunidades. En contraste, una comunidad única y bien definida, es una subgráfica localmente densa con distribución de grados similar a la distribución de una red aleatoria. Por lo tanto, la calidad de una partición de comunidades puede ser medida al comparar la densidad de cada comunidad con la densidad de una gráfica aleatoria hipotética con el mismo número de vértices. Esta medida es llamada *modularidad*.

Definición A.2.43: La *modularidad* de una comunidad, denotada M_c, mide su calidad al comparar su densidad con la densidad de la gráfica G que la contiene en su versión aleatoria. Su ecuación es definida como sigue:

$$M_c = \frac{L_c}{m} - \left(\frac{K_c}{2m}\right)^2 \tag{A.9}$$

Donde:

- L_c representa el número de enlaces que interconectan a la comunidad.
- m representa el tamaño de G.
- K_c representa la suma de los grados de los vértices de la comunidad.

De la Figura A.8, M_c para ambas comunidades es $\frac{9}{44}$.

Definición A.2.44: Sea G una gráfica no dirigida particionada en comunidades. La *modularidad* de dicha partición de G, denotada M, se define como sigue:

$$M = \sum_c M_c \tag{A.10}$$

M oscila en el intervalo $[-0.5, 1]$. Más aún, se pueden presentar tres casos posibles:

1) Si $M = 0$, la gráfica G es en sí misma la única comunidad dado que $L_c = m$ y $K_c = 2m$, lo cual indica que la gráfica tiene la misma cantidad de aristas que tendría en su versión aleatoria.

2) Si $M < 0$, cada vértice de G, representa una comunidad dado que $L_c = 0$, lo cual indica que la gráfica tiene menos enlaces que los que tendría en su versión aleatoria.

3) Si $M > 0$, la gráfica tiene más enlaces que los que tendría en su versión aleatoria.

El método más popular de optimización de la modularidad, es el *algoritmo de Louvain*[1]. Este algoritmo, tiene una complejidad $O(n)$, lo que permite la optimización de la modularidad en redes con millones de vértices. El algoritmo trabaja en iteraciones, cada una de las cuales, se divide en dos pasos:

1. Se itera sobre los vértices de la red: Se elige un vértice y se mide el cambio de modularidad al agregarlo a las comunidades de cada uno de sus vecinos. Se agrega el vértice a la comunidad donde se obtiene la mayor ganancia de modularidad positiva.

2. Cada comunidad se contrae en un super-vértice con un lazo, cuyo peso corresponde al número de aristas que unen a la comunidad. Las aristas de la nueva red, tienen un peso que es análogo al número de aristas entre las comunidades.

Las iteraciones, se computan hasta que no haya incrementos de modularidad en la red.

Definición A.2.45: Sea G una gráfica conexa no dirigida. La *centralidad de cercanía* de un vértice $v \in V$, denotada $CC(v)$, se define como el inverso de la suma de las distancias de v hacia los demás vértices de la red. Su ecuación, se define como sigue:

$$CC(v) = \frac{n-1}{\sum_{v=1}^{n-1} d(u,v)} \tag{A.11}$$

Donde:

- $d(u,v)$ representa la distancia más corta entre los vértices u y v.
- $n-1$ representa el número de vértices alcanzables por v.

Ejemplo A.2.9: La Tabla A.2 muestra el cálculo de la centralidad de cercanía de todos los vértices de la gráfica mostrada en la Figura A.1.

Tabla A.2: Centralidad de cercanía de los vértices de la gráfica mostrada en la Figura A.1

v	1	2	3	4	5	6	$CC(v)$
1	0	1	1	2	2	3	$\frac{5}{9}$
2	1	0	1	1	2	2	$\frac{5}{7}$
3	1	1	0	1	1	2	$\frac{5}{6}$
4	2	1	1	0	1	1	$\frac{5}{6}$
5	2	2	1	1	0	1	$\frac{5}{7}$
6	3	2	2	1	1	0	$\frac{5}{9}$

Definición A.2.46: Sea G una gráfica no dirigida. La *centralidad de intermediación* de un vértice $v \in V$, denotada $BC(v)$, mide la proporción de trayectorias más cortas que pasan por un vértice. Su ecuación, se define como sigue:

$$BC(v) = \sum_{s,t \in V, s \neq t} \frac{\sigma(s,t \mid v)}{\sigma(s,t)} \tag{A.12}$$

Donde:

- $\sigma(s,t)$ representa el número de trayectorias más cortas entre los vértices s y t.
- $\sigma(s,t \mid v)$ representa el número de trayectorias contenidas en $\sigma(s,t)$ tales que pasan a través del vértice v distinto de s y t.

Destacando dos aspectos importantes:

1. Si $s = t \Rightarrow \sigma(s,t) = 1$.
2. Si $v = s \vee v = t \Rightarrow \sigma(s,t \mid v) = 0$.

Los valores de la centralidad de intermediación pueden normalizarse para que oscilen en el intervalo $[0,1]$, como sigue:

$$BC(v) = \frac{2BC(v)}{(n-1)(n-2)} \tag{A.13}$$

Ejemplo A.2.10: La Tabla A.3 muestra el cálculo de la centralidad de intermediación de todos los vértices de la gráfica mostrada en la Figura A.1.

Definición A.2.47: Sea G una gráfica no dirigida. El *coeficiente de asortatividad* de G, denotado r, mide la asociación de vértices con propiedades similares, en particular, la propiedad que se ejemplificará es el grado de los vértices. El coeficiente de asortatividad, se puede interpretar como una medida de *homofilia* en una gráfica. Dicho coeficiente, se calcula a través del coeficiente de correlación de Pearson, evaluado en los grados de los vértices incidentes sobre cada arista $e = (u,v) : u,v \in V$. Redes altamente asortativas, tienden a

Tabla A.3: Centralidad de intermediación de los vértices de la gráfica mostrada en la Figura A.1

$(s-t)$	Caminos más cortos	$v=1$	$v=2$	$v=3$	$v=4$	$v=5$	$v=6$
$(1-2)$	-	-	-	-	-	-	-
$(1-3)$	-	-	-	-	-	-	-
$(1-4)$	$1-2-4$ $1-3-4$	0	$\frac{1}{2}$	$\frac{1}{2}$	0	0	0
$(1-5)$	$1-3-5$	0	0	1	0	0	0
$(1-6)$	$1-2-4-6$ $1-3-4-6$ $1-3-5-6$	0	$\frac{1}{3}$	$\frac{2}{3}$	$\frac{2}{3}$	$\frac{1}{3}$	0
$(2-3)$	-	-	-	-	-	-	-
$(2-4)$	-	-	-	-	-	-	-
$(2-5)$	$2-3-5$ $2-4-5$	0	0	$\frac{1}{2}$	$\frac{1}{2}$	0	0
$(2-6)$	$2-4-6$	0	0	0	1	0	0
$(3-4)$	-	-	-	-	-	-	-
$(3-5)$	-	-	-	-	-	-	-
$(3-6)$	$3-5-6$ $3-4-6$	0	0	0	$\frac{1}{2}$	$\frac{1}{2}$	0
$(4-5)$	-	-	-	-	-	-	-
$(4-6)$	-	-	-	-	-	-	-
$(5-6)$	-	-	-	-	-	-	-
$BC(v)$:		0	$\frac{5}{6}$	$\frac{8}{3}$	$\frac{8}{3}$	$\frac{5}{6}$	0
$BC(v)$ normalizada:		0	$\frac{1}{12}$	$\frac{4}{15}$	$\frac{4}{15}$	$\frac{1}{12}$	0

ser más robustas ante eventos destructivos, como la pérdida de sus vértices, sin embargo, debido a su concentración, pueden ser ineficientes en términos de su flujo de información.

El coeficiente de asortatividad, oscila en [-1,1], justo como un coeficiente de correlación. Su ecuación, se define como sigue:

$$r = \frac{\sum_{i=1}^{m}(X_i - \bar{X})(Y_i - \bar{Y})}{\sqrt{\sum_{i=1}^{m}(X_i - \bar{X})^2}\sqrt{\sum_{i=1}^{m}(Y_i - \bar{Y})^2}} \tag{A.14}$$

Donde:

- X_i representa el grado del vértice u en la i-ésima arista de G.
- Y_i representa el grado del vértice v en la i-ésima arista de G.
- $\bar{X}$ representa la media aritmética de los grados de los vértices u en las aristas de G.
- $\bar{Y}$ representa la media aritmética de los grados de los vértices v en las aristas de G.

Se presentan tres casos posibles, con respecto al coeficiente de asortatividad r:

1) Valores positivos de r, señalan que G es asortativa, lo cual, indica que vértices de grado similar tienden a conectarse entre sí. Si $r = 1$, se dice que G es perfectamente asortativa, es decir, vértices con mismo grado son adyacentes.

2) Valores negativos de r, señalan que G es disortativa, lo cual, indica que vértices de grado no similar tienden a conectarse entre ellos, sobre conectarse con vértices de grado similar.

3) Valores de r cercanos a 0, señalan que G no es ni asortativa ni disortativa, es decir, no existe correlación entre los grados de vértices adyacentes, por ende, se considera que G es neutral.

Ejemplo A.2.11: La Tabla A.4 muestra el cálculo del coeficiente de asortatividad de

la gráfica mostrada en la Figura A.1.

Tabla A.4: Cálculo del coeficiente de asortatividad de la gráfica mostrada en la Figura A.1

Aristas	X	Y	$X-\bar{X}$	$(X-\bar{X})^2$	$Y-\bar{Y}$	$(Y-\bar{Y})^2$	$(X-\bar{X})(Y-\bar{Y})$
$e_1=(1,2)$	2	3	$-\frac{11}{9}$	$-\frac{2}{9}$	$\frac{121}{81}$	$\frac{4}{81}$	$\frac{22}{81}$
$e_2=(1,3)$	2	4	$-\frac{11}{9}$	$\frac{7}{9}$	$\frac{121}{81}$	$\frac{49}{81}$	$-\frac{77}{81}$
$e_3=(2,3)$	3	4	$-\frac{2}{9}$	$\frac{7}{9}$	$\frac{4}{81}$	$\frac{49}{81}$	$-\frac{14}{81}$
$e_4=(2,4)$	3	4	$-\frac{2}{9}$	$\frac{7}{9}$	$\frac{4}{81}$	$\frac{49}{81}$	$-\frac{14}{81}$
$e_5=(3,4)$	4	4	$\frac{7}{9}$	$\frac{7}{9}$	$\frac{49}{81}$	$\frac{49}{81}$	$\frac{49}{81}$
$e_6=(3,5)$	4	3	$\frac{7}{9}$	$-\frac{2}{9}$	$\frac{49}{81}$	$\frac{4}{81}$	$-\frac{14}{81}$
$e_7=(4,5)$	4	3	$\frac{7}{9}$	$-\frac{2}{9}$	$\frac{49}{81}$	$\frac{4}{81}$	$-\frac{14}{81}$
$e_8=(4,6)$	4	2	$\frac{7}{9}$	$-\frac{11}{9}$	$\frac{49}{81}$	$\frac{121}{81}$	$-\frac{77}{81}$
$e_9=(5,6)$	3	2	$-\frac{2}{9}$	$-\frac{11}{9}$	$\frac{4}{81}$	$\frac{121}{81}$	$\frac{22}{81}$

Al efectuar los cálculos correspondientes, se obtiene:

$$\bar{X}=\tfrac{29}{9}$$

$$\bar{Y}=\tfrac{29}{9}$$

$$\sqrt{\textstyle\sum_{i=1}^{m}(X_i-\bar{X})^2}=\tfrac{\sqrt{450}}{9}$$

$$\sqrt{\textstyle\sum_{i=1}^{m}(Y_i-\bar{Y})^2}=\tfrac{\sqrt{450}}{9}$$

$$\textstyle\sum_{i=1}^{m}(X_i-\bar{X})(Y_i-\bar{Y})=-\tfrac{117}{81}$$

Sustituyendo los datos, se obtiene:

$$r=-\tfrac{117}{450}$$

Definición A.2.48: Una red multicapa, está dada por una tupla $M=(G,C)$, donde

G es un conjunto de gráficas $G_i = (V_i, E_i)$, tal que cada G_i denota la i-ésima capa de M y, por lo tanto, E_i denota el conjunto de aristas que conectan vértices en la misma capa G_i; estas aristas son denominadas *aristas intracapa*. Mientras tanto, C es el conjunto de aristas que conectan vértices de diferentes capas, las cuales son denominadas *aristas entre capas* [46].

Definición A.2.49: Sea $M = (G, C)$ una red multicapa, donde G es el conjunto de capas $G_i = (V_i, E_i)$ y C es el conjunto de capas internas. La i-ésima red de proyección de M está dada por la unión de capas consecutivas de G_1 a G_i, y es definida como sigue [46]:

$$P_i(M) = (V_i, E_i), \text{ donde } V_i = \bigcup_{1 \leq j \leq i} V_j \text{ y } E_i = \bigcup_{1 \leq j \leq i} E_j.$$

Definición A.2.50: Se dice que una gráfica G, es *vértice-transitiva*, si en términos generales, todos sus vértices tienen la misma perspectiva de toda la gráfica y, por lo tanto, no pueden ser distinguidos de cualquier otro con base en sus vecindarios. Esta propiedad, permite ejecutar procesos distribuidos usando los mismos algoritmos en cada vértice. Por ejemplo, el encaminamiento distribuido puede ser ejecutado usando los mismos algoritmos de encaminamiento en cada vértice [47].

Definición A.2.51: Se dice que una gráfica G, es *arista-transitiva*, si todas sus aristas tienen la misma perspectiva de toda la gráfica y, por lo tanto, no pueden ser distinguidas de cualquier otra con base en los vértices y aristas que las rodean [47].

Definición A.2.52: Se dice que una gráfica G, es *simétrica*, si es vértice-transitiva y arista-transitiva. En general, las gráficas que son vértice-transitivas y/o arista-transitivas, tienen un grado regular [47].

A.3. Representación de gráficas

Debido a que en esta investigación se trabajó únicamente con gráficas simples y no dirigidas, a continuación se muestran cuatro posibles formas de representar a tales gráficas:

1) *Matriz de Incidencias*: La matriz de incidencias I de una gráfica simple y no dirigida G con $N = n$ y $L = m$, es una matriz de N renglones por L columnas cuyos elementos (denotados $I_{i,j}$, donde i representa el i-ésimo renglón y j la j-ésima columna) son:

$$I_{i,j} = \begin{cases} 1 & \text{si el el vértice } i \text{ es incidente en la arista } e_j \\ 0 & \text{en otro caso} \end{cases} \tag{A.15}$$

Ejemplo A.2.1: A continuación, se muestra la matriz de incidencias correspondiente a la gráfica mostrada en la Figura A.1.

$$I = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix}$$

La matriz de incidencias, no suele ser una representación óptima en términos de espacio, debido a que cada columna de dicha matriz solo contiene dos elementos establecidos en 1, los cuales, representan a los vértices incidentes en la j-ésima arista, teniendo así, demasiadas entradas establecidas en 0, que hacen que al almacenar la matriz en una computadora, ocupe una cantidad excesiva de memoria. El peor de los casos se presenta al intentar representar a una gráfica completa K_z de orden z mediante su matriz de incidencias, lo cual, daría como resultado una matriz I de z renglones por $\frac{z(z-1)}{2}$ columnas, con un total de $\frac{z^3-z^2}{2}$ entradas, de las cuales solo $z^2 - z$ estarían establecidas en 1 y el restante en 0.

2) *Matriz de Adyacencias*: La matriz de adyacencias A de una gráfica simple y no dirigida G con $N = n$, es una matriz cuadrada con N renglones por N columnas cuyos elementos

(denotados $A_{i,j}$, donde i representa el i-ésimo renglón y j la j-ésima columna) son:

$$A_{i,j} = \begin{cases} 1 & \text{si el vértice } i \text{ es adyacente al vértice } j \\ 0 & \text{en otro caso} \end{cases} \tag{A.16}$$

Ejemplo A.2.2: A continuación, se muestra la matriz de adyacencias correspondiente a la gráfica mostrada en la Figura A.1.

$$A = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 \end{bmatrix}$$

Al tratarse de una gráfica no dirigida, se tiene que $A = A^T$, es decir, es igual a su transpuesta y, por lo tanto, es simétrica, siendo 0 todos los elementos de la diagonal principal, ya que no se tienen bucles o lazos. La matriz de adyacencias, al igual que la matriz de incidencias, no suele ser una representación óptima en términos de espacio, debido a que dicha matriz crece de manera cuadrática con respecto al orden de la gráfica que representa y, al ser simétrica, se estará teniendo demasiada redundancia de información y, por lo tanto, al ser almacenada en una computadora, ocuparía demasiada memoria innecesariamente.

3) *Lista de Adyacencias*: La lista de adyacencias de una gráfica simple y no dirigida G, está formada por n renglones, donde el i-esimo renglón contiene la lista de identificadores de los vértices con los que el vértice i es adyacente.

Ejemplo A.2.3: A continuación, se muestra la lista de adyacencias correspondiente a la gráfica mostrada en la Figura A.1.

2, 3

1, 3, 4

1, 2, 4, 5

2, 3, 5, 6

3, 4, 6

4, 5

Se puede observar que la lista de adyacencias de una gráfica G contiene redundancias, debido a que si el vértice i es adyacente al vértice j, j aparece en la lista de i y, a su vez, i aparece en la lista de j.

4) *Lista de Aristas*: La lista de aristas de una gráfica simple y no dirigida G, está formada por m parejas de vértices (u, v), donde la i-ésima pareja índica que el vértice u es adyacente al vértice v. La lista de aristas, no admite redundancias, es decir, si existe la pareja de vértices (u, v), se asume que la pareja (v, u) está también incluida y no debe de escribirse explícitamente.

Ejemplo A.2.4: A continuación, se muestra la lista de aristas correspondiente a la gráfica mostrada en la Figura A.1.

(1,2), (1,3), (2,3), (2,4), (3,4), (3,5), (4,5), (4,6), (5,6)

Se puede observar que esta representación de gráficas, resulta ser óptima en términos de espacio (memoria), comparado con la matriz de adyacencias, lista de incidencias y lista de adyacencias, ya que no acepta redundancias y utiliza la menor cantidad de espacio al ser almacenada en una computadora.

Apéndice B

Análisis semi-cooperadores

Tabla B.1: Análisis con tamaño máximo de arista D y $\frac{D}{2}$

(a) Análisis con tamaño máximo de arista D

Experimento	$\bar{\delta}(v)$	$\frac{n}{4}$	$\frac{n}{16}$	$\frac{n}{64}$
Malla 50 x 50 vértices/Compass-Routing/R1	108.2	NO	NO	NO
Malla 50 x 50 vértices/Random-Walk/R1	247.8	NO	NO	SI
Malla 50 x 50 vértices/Shortest-Path/R1	2482.2	SI	SI	SI
Malla 50 x 50 vértices/Compass-Routing/R2	20.4	NO	NO	NO
Malla 50 x 50 vértices/Random-Walk/R2	16.1	NO	NO	NO
Malla 50 x 50 vértices/Shortest-Path/R2	2498.1	SI	SI	SI
Malla 50 x 50 vértices/Compass-Routing/R3	16.5	NO	NO	NO
Malla 50 x 50 vértices/Random-Walk/R3	15.6	NO	NO	NO
Malla 50 x 50 vértices/Shortest-Path/R3	2498.3	SI	SI	SI
Anillo 1500 vértices/Compass-Routing/R1	170.2	NO	NO	SI
Anillo 1500 vértices/Random-Walk/R1	65	NO	NO	NO
Anillo 1500 vértices/Shortest-Path/R1	1468.1	SI	SI	SI
Anillo 1500 vértices/Compass-Routing/R2	14.5	NO	NO	NO
Anillo 1500 vértices/Random-Walk/R2	14.1	NO	NO	NO
Anillo 1500 vértices/Shortest-Path/R2	1498.8	SI	SI	SI
Anillo 1500 vértices/Compass-Routing/R3	12.9	NO	NO	NO
Anillo 1500 vértices/Random-Walk/R3	14.3	NO	NO	NO
Anillo 1500 vértices/Shortest-Path/R3	1498.6	SI	SI	SI

(b) Análisis con tamaño máximo de arista $\frac{D}{2}$

Experimento	$\bar{\delta}(v)$	$\frac{n}{4}$	$\frac{n}{16}$	$\frac{n}{64}$
Malla 50 x 50 vértices/Compass-Routing/R1	92.4	NO	NO	NO
Malla 50 x 50 vértices/Random-Walk/R1	249	NO	NO	SI
Malla 50 x 50 vértices/Shortest-Path/R1	2379.1	SI	SI	SI
Malla 50 x 50 vértices/Compass-Routing/R2	27.1	NO	NO	NO
Malla 50 x 50 vértices/Random-Walk/R2	15.8	NO	NO	NO
Malla 50 x 50 vértices/Shortest-Path/R2	2436.1	SI	SI	SI
Malla 50 x 50 vértices/Compass-Routing/R3	19.3	NO	NO	NO
Malla 50 x 50 vértices/Random-Walk/R3	16.2	NO	NO	NO
Malla 50 x 50 vértices/Shortest-Path/R3	2384.9	SI	SI	SI
Anillo 1500 vértices/Compass-Routing/R1	116.6	NO	NO	SI
Anillo 1500 vértices/Random-Walk/R1	64	NO	NO	NO
Anillo 1500 vértices/Shortest-Path/R1	484.6	NO	SI	SI
Anillo 1500 vértices/Compass-Routing/R2	17.4	NO	NO	NO
Anillo 1500 vértices/Random-Walk/R2	14.2	NO	NO	NO
Anillo 1500 vértices/Shortest-Path/R2	479.2	NO	SI	SI
Anillo 1500 vértices/Compass-Routing/R3	13.4	NO	NO	NO
Anillo 1500 vértices/Random-Walk/R3	14.3	NO	NO	NO
Anillo 1500 vértices/Shortest-Path/R3	457.5	NO	SI	SI

Tabla B.2: Análisis con tamaño máximo de arista $\frac{D}{4}$, $\frac{D}{8}$ y $\frac{D}{16}$

(a) Análisis con tamaño máximo de arista $\frac{D}{4}$

Experimento	$\bar{\delta}(v)$	$\frac{n}{4}$	$\frac{n}{16}$	$\frac{n}{64}$
Malla 50 x 50 vértices/Compass-Routing/R1	46.5	NO	NO	NO
Malla 50 x 50 vértices/Random-Walk/R1	223.9	NO	NO	SI
Malla 50 x 50 vértices/Shortest-Path/R1	948	NO	SI	SI
Malla 50 x 50 vértices/Compass-Routing/R2	31.8	NO	NO	NO
Malla 50 x 50 vértices/Random-Walk/R2	17.1	NO	NO	NO
Malla 50 x 50 vértices/Shortest-Path/R2	944.7	NO	SI	SI
Malla 50 x 50 vértices/Compass-Routing/R3	20.6	NO	NO	NO
Malla 50 x 50 vértices/Random-Walk/R3	16.1	NO	NO	NO
Malla 50 x 50 vértices/Shortest-Path/R3	931.2	NO	SI	SI
Anillo 1500 vértices/Compass-Routing/R1	58.2	NO	NO	NO
Anillo 1500 vértices/Random-Walk/R1	65.5	NO	NO	NO
Anillo 1500 vértices/Shortest-Path/R1	213.6	NO	NO	SI
Anillo 1500 vértices/Compass-Routing/R2	20.8	NO	NO	NO
Anillo 1500 vértices/Random-Walk/R2	14.6	NO	NO	NO
Anillo 1500 vértices/Shortest-Path/R2	194.7	NO	NO	SI
Anillo 1500 vértices/Compass-Routing/R3	14.4	NO	NO	NO
Anillo 1500 vértices/Random-Walk/R3	14.2	NO	NO	NO
Anillo 1500 vértices/Shortest-Path/R3	178.3	NO	NO	SI

(b) Análisis con tamaño máximo de arista $\frac{D}{8}$

Experimento	$\bar{\delta}(v)$	$\frac{n}{4}$	$\frac{n}{16}$	$\frac{n}{64}$
Malla 50 x 50 vértices/Compass-Routing/R1	23.1	NO	NO	NO
Malla 50 x 50 vértices/Random-Walk/R1	153.6	NO	NO	SI
Malla 50 x 50 vértices/Shortest-Path/R1	224.7	NO	NO	SI
Malla 50 x 50 vértices/Compass-Routing/R2	21.4	NO	NO	NO
Malla 50 x 50 vértices/Random-Walk/R2	15.7	NO	NO	NO
Malla 50 x 50 vértices/Shortest-Path/R2	217	NO	NO	SI
Malla 50 x 50 vértices/Compass-Routing/R3	19.4	NO	NO	NO
Malla 50 x 50 vértices/Random-Walk/R3	16.5	NO	NO	NO
Malla 50 x 50 vértices/Shortest-Path/R3	178	NO	NO	SI
Anillo 1500 vértices/Compass-Routing/R1	38.4	NO	NO	NO
Anillo 1500 vértices/Random-Walk/R1	62.5	NO	NO	NO
Anillo 1500 vértices/Shortest-Path/R1	104.2	NO	NO	SI
Anillo 1500 vértices/Compass-Routing/R2	21.2	NO	NO	NO
Anillo 1500 vértices/Random-Walk/R2	14.5	NO	NO	NO
Anillo 1500 vértices/Shortest-Path/R2	76.1	NO	NO	SI
Anillo 1500 vértices/Compass-Routing/R3	25.3	NO	NO	NO
Anillo 1500 vértices/Random-Walk/R3	13.9	NO	NO	NO
Anillo 1500 vértices/Shortest-Path/R3	63.2	NO	NO	NO

(c) Análisis con tamaño máximo de arista $\frac{D}{16}$

Experimento	$\bar{\delta}(v)$	$\frac{n}{4}$	$\frac{n}{16}$	$\frac{n}{64}$
Malla 50 x 50 vértices/Compass-Routing/R1	14.2	NO	NO	NO
Malla 50 x 50 vértices/Random-Walk/R1	57.9	NO	NO	NO
Malla 50 x 50 vértices/Shortest-Path/R1	49.3	NO	NO	NO
Malla 50 x 50 vértices/Compass-Routing/R2	17.6	NO	NO	NO
Malla 50 x 50 vértices/Random-Walk/R2	15.2	NO	NO	NO
Malla 50 x 50 vértices/Shortest-Path/R2	47.7	NO	NO	NO
Malla 50 x 50 vértices/Compass-Routing/R3	16.7	NO	NO	NO
Malla 50 x 50 vértices/Random-Walk/R3	15.1	NO	NO	NO
Malla 50 x 50 vértices/Shortest-Path/R3	39.3	NO	NO	NO
Anillo 1500 vértices/Compass-Routing/R1	28.5	NO	NO	NO
Anillo 1500 vértices/Random-Walk/R1	49.9	NO	NO	NO
Anillo 1500 vértices/Shortest-Path/R1	53.6	NO	NO	NO
Anillo 1500 vértices/Compass-Routing/R2	21.3	NO	NO	NO
Anillo 1500 vértices/Random-Walk/R2	13.3	NO	NO	NO
Anillo 1500 vértices/Shortest-Path/R2	39.9	NO	NO	NO
Anillo 1500 vértices/Compass-Routing/R3	30.7	NO	NO	NO
Anillo 1500 vértices/Random-Walk/R3	13.3	NO	NO	NO
Anillo 1500 vértices/Shortest-Path/R3	39.2	NO	NO	NO

Apéndice C

Ejecución de los experimentos

C.1. Introducción

El simulador de eventos discretos escrito para llevar a cabo la ejecución de los experimentos propuestos en esta investigación, se compone de las siguientes clases[1]:

1. Main, referente al script "main.py".
2. ComplexNetwork, referente al script "complexNetwork.py".
3. Paquete, referente al script "paquete.py".
4. Enlace, referente al script "enlace.py".
5. Encaminamiento, referente al script "encaminamiento.py".
6. Event, referente al script "event.py".
7. Process, referente al script "process.py".

[1]Todos los scripts de código referidos en este Apéndice, se encuentran disponibles en un repositorio público de GitHub, accesible a través de: *https://github.com/Jorge27873/ComplexNetworks.git*

8. Model, referente al script “model.py”.

9. Simulation, referente al script “simulation.py”.

10. Simulator, referente al script “simulator.py”.

Más aún, el simulador, también hace uso de un script denominado “extractData.py”, el cual, se encarga de calcular las medidas de las propiedades estructurales, ciclo a ciclo de experimentación, de las redes emergentes.

C.2. Fase 1: Formación de redes complejas

Para una correcta y clara organización de los archivos que arroja cada una de las ejecuciones de los experimentos de la fase de formación, se creó un árbol de directorios[2] para cada uno de los cinco tamaños posibles de arista dinámica. En la Figura C.1, se muestra el árbol general de directorios para la fase de formación.

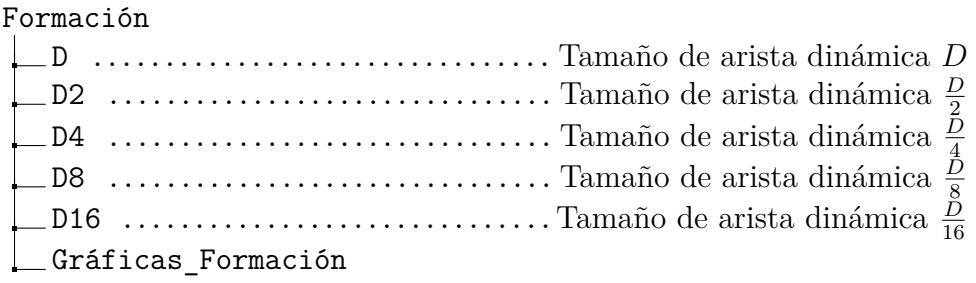


Figura C.1: Árbol de directorios general fase de formación

En la Figura C.2, se muestra que dentro del directorio “D”, se encuentra el directorio “ExperimentosCooperadores”, y dentro de él; cada una de sus posibles configuraciones para tal tamaño de arista dinámica. Más aún, los directorios D2, D4, D8 y D16, se definen de manera análoga al directorio D.

[2]Los árboles de directorios referidos en este Apéndice, se encuentran en un archivo con extensión “rar” disponible en: *https://drive.google.com/file/d/1aMg09MBAGl1M3cw6n8v6FmYhVx7nyC-n/view?usp=sharing*

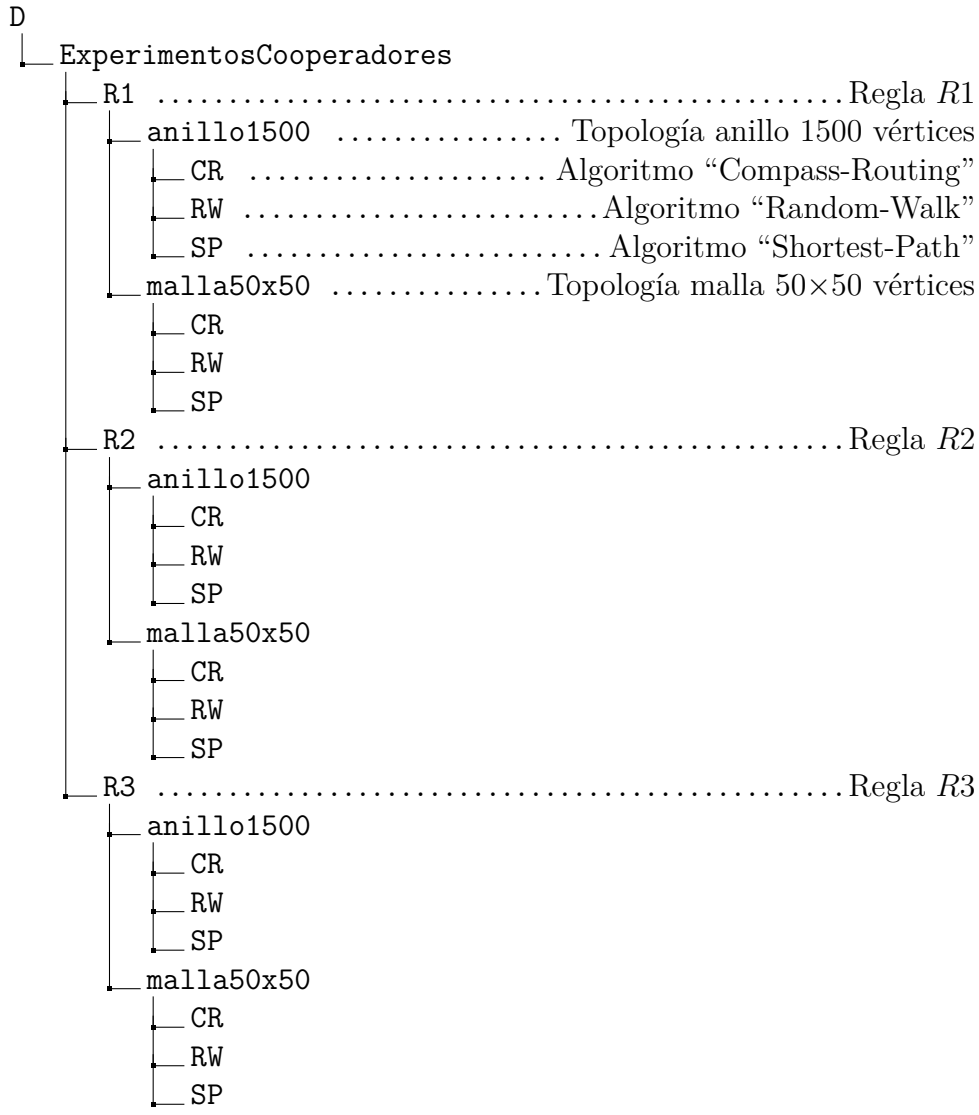


Figura C.2: Árbol de directorios dentro de "D", cooperadores

Por otro lado, para automatizar la ejecución de los experimentos, se escribió un script de bash denominado "formación.sh", el cual, se ejecutó en el sistema operativo Ubuntu, Linux, creando una copia de este script para cada uno de los 18 experimentos "base" cooperadores, con sus respectivas rutas de acceso a los directorios correspondientes de acuerdo a la configuración de cada experimento, los cuales, se ejecutaron en paralelo en una computadora con varios núcleos de procesamiento, esto permitió reducir considerablemente los tiempos de ejecución de los experimentos. Para realizar la ejecución de los scripts desde una terminal, se requiere que dentro de cada uno de los directorios hoja del árbol mostrado en la Figura C.2, estén presentes las 10 clases del simulador (la clase main.py, debe estar ajustada de acuerdo a los parámetros de configuración de cada experimento), el script "extractData.py", así como el anillo de 1500 vértices o la malla de 2500 vértices con extensión "adjlist" (según corresponda), los cuales, se pueden crear con la ayuda de la biblioteca NetworkX de python. Para automatizar la copia de dichos archivos a los directorios hoja del árbol mostrado en la Figura C.2, se escribió el script de python denominado "0creaCopiaSimulador.py", el cual, copia todos los archivos mencionados anteriormente, a excepción de la clase Main, la cual, se debe copiar manualmente con sus parámetros ya ajustados de acuerdo a la configuración del experimento en cuestión. En el repositorio de GitHub, se encuentra un ejemplo de script "formación.sh", el cual, fue ideado para el experimento "base" cooperador con configuración: Anillo 1500 vértices/Compass-Routing/R1.

La idea es que, una vez que se hayan ejecutado los experimentos cooperadores de la fase de formación, dentro de cada uno de los directorios hoja del árbol mostrado en la Figura C.2, se habrán creado 10 directorios nombrados mediante números $w : w \in \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$, donde, dentro de cada uno de ellos, se encontrarán los archivos que arroja cada ejecución de los experimentos.

Una vez obtenidos los resultados del conjunto de experimentos referente a la fase de formación, se escribió el script de python "1creaPromediosFormacion.py", el cual, automatiza el cálculo de los promedios y desviaciones estándar, ciclo a ciclo de experimentación, de las

medidas de las propiedades estructurales de las redes. Por otro lado, se escribió el script de python "2creaPromediosDistGradosFormacion.py", el cual, automatiza el cálculo de los promedios de los histogramas de grados de los experimentos. Ambos scripts, escriben los resultados en archivos con extensión "csv" (archivo de valores separados por comas, por sus siglas en inglés). Más aún, se escribió el script de python "7creaGraficasFormacion.py", el cual, genera una serie de 18 archivos con extensión "csv" (uno por cada experimento "base") en el directorio "Gráficas_Formación" de la Figura C.1, estos archivos, permiten graficar las medidas de las propiedades estructurales de las redes a través de los 50 ciclos de experimentación, así como sus distribuciones de grados.

Para definir el conjunto de experimentos semi-cooperadores, se escribió el script de python "5creaSemiCooperadores.py", el cual, a través de un archivo con extensión "csv", entrega las configuraciones de los experimentos que se considerarán en este segundo conjunto de experimentos. Los datos mostrados en las Tablas B.1a, B.1b, B.2a, B.2b, y B.2c, fueron obtenidos con dicho script.

De manera análoga a los experimentos cooperadores, se siguió la idea de creación de un árbol de directorios para una clara y correcta organización de archivos. En la Figura C.3, se muestra que dentro del directorio "D" del árbol mostrado en la Figura C.2, se anexó el directorio "ExperimentosSemi-Cooperadores", y dentro de él; cada una de sus posibles configuraciones para tal tamaño de arista dinámica. Los directorios D2, D4, D8 y D16, se definen de manera análoga al árbol de directorios de los experimentos cooperadores, con la restricción de que no para todos los tamaños posibles de arista dinámica, aplican las configuraciones $\frac{n}{4}$, $\frac{n}{16}$ y $\frac{n}{64}$. Por ejemplo, en la Tabla B.2c, se puede observar que para el tamaño de arista dinámica $\frac{D}{16}$, no aplica ninguna de las tres configuraciones, por lo tanto, el directorio "ExperimentosSemi-Cooperadores", no estará definido para el directorio "D16".

Se crearon 18 copias más del script "formación.sh", con sus respectivas rutas de acceso a los directorios correspondientes de acuerdo a la configuración de cada experimento semi-

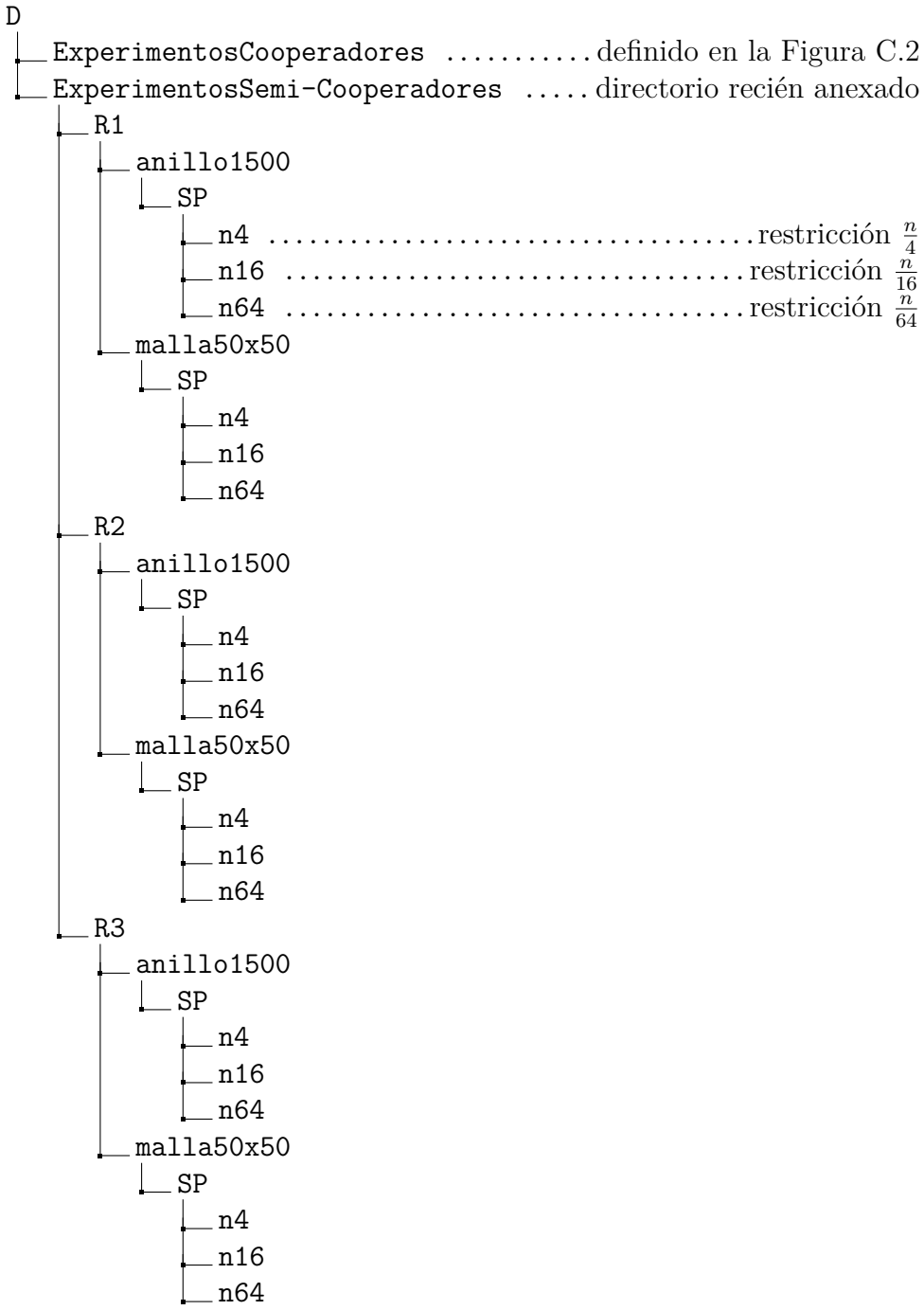


Figura C.3: Árbol de directorios dentro de "D", semi-cooperadores

cooperador (considerando los 6 experimentos que usan Shortest-Path, como algoritmo de encaminamiento, contenidos en los 18 experimentos "base", multiplicado por las tres restricciones $\frac{n}{4}$, $\frac{n}{16}$ y $\frac{n}{64}$). Su ejecución, creación de promedios y desviaciones estándar, promedios de histogramas de grados, así como los datos requeridos para graficar las medidas de las propiedades estructurales de los 50 ciclos de experimentación, se ejecutan de manera análoga al conjunto de experimentos cooperadores.

C.3. Fase 2: Degradación de redes complejas

Para esta fase, se creó el árbol de directorios mostrado en la Figura C.4. Los directorios D, D2, D4, D8 y D16, se definen de manera análoga al árbol de directorios de la fase de formación.

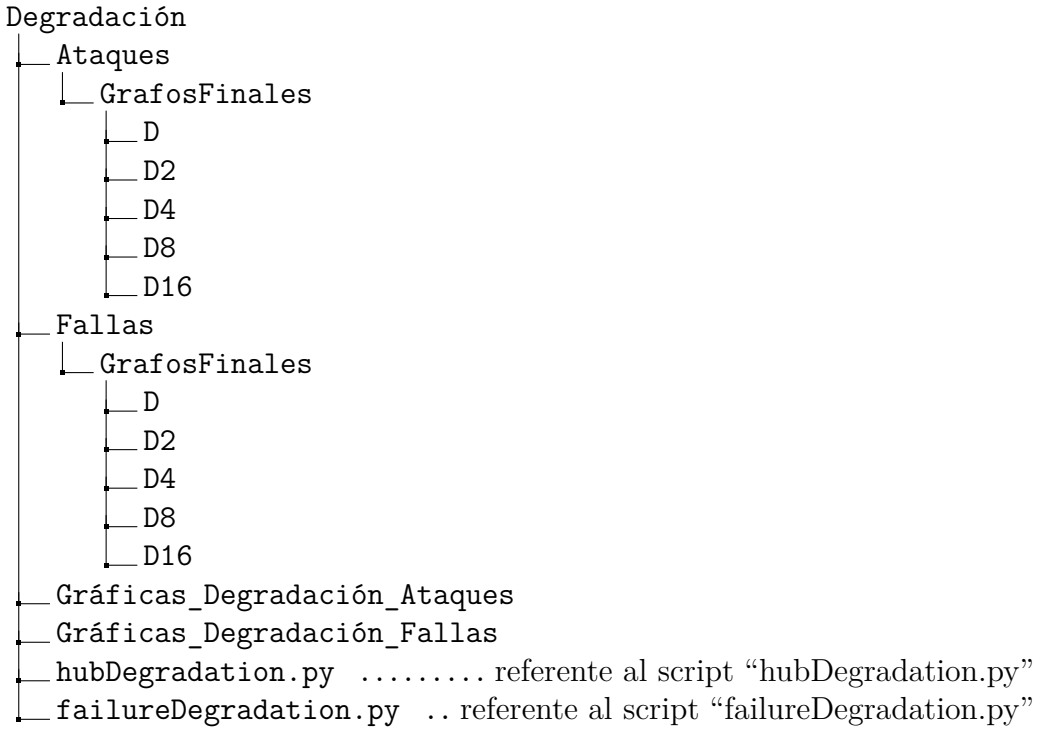


Figura C.4: Árbol de directorios general fase de degradación

Para automatizar la ejecución de los experimentos de esta fase, se decidió crear un script llamado "degradación.sh" para cada experimento "base" cooperador (18 en total), y uno pa-

ra cada experimento semi-cooperador (18 en total), pero, en esta ocasión, se consideraron los dos procesos de degradación utilizados en esta investigación, lo que dio por resultado, 72 scripts (18 cooperadores-ataques, 18 cooperadores-fallas, 18 semi-cooperadores-ataques, 18 semi-cooperadores-fallas), los cuales, se ejecutaron en paralelo, en una computadora con varios núcleos de procesamiento, permitiendo reducir considerablemente los tiempos de ejecución. Para realizar la ejecución de los scripts desde una terminal, se requiere que en cada directorio hoja del árbol de directorios de la fase de degradación (es decir, los directorios nombrados con números $w : w \in \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$), se encuentre única y exclusivamente el archivo correspondiente a la gráfica del ciclo final de experimentación con extensión "adjlist". Los scripts de python "failureDegradation.py" y "hubDegradation.py", se encargan de degradar a las redes por fallas aleatorias y ataques secuenciales dirigidos, respectivamente. En total, se tendrían 2740 gráficas con extensión "adjlist" de ciclos finales, 1370 para fallas aleatorias y 1370 para ataques secuenciales dirigidos, almacenadas en sus respectivos directorios. En el repositorio de GitHub, se encuentra un ejemplo de script "degradación.sh", el cual, fue ideado para el experimento "base" cooperador con configuración: Ataques/Anillo 1500 vértices/Compass-Routing/R1.

Una vez obtenidos los resultados de este conjunto de experimentos, se escribió el script de python "6creaPromediosDegradacion.py", el cual, automatiza el cálculo de promedios y desviaciones estándar de las propiedades estructurales, así como de diversas métricas de las redes degradadas. Por otro lado, se escribió el script de python "8creaGraficasDegradacion.py", el cual, genera una serie de 36 archivos con extensión "csv" (uno por cada experimento "base" cooperador y uno por cada experimento semi-cooperador) en el directorio "Gráficas_Degradación_Ataques" de la Figura C.4, estos archivos, permiten graficar las medidas de las propiedades estructurales y diversas métricas cada 1 % de vértices removidos de las redes mediante ataques secuenciales dirigidos. Además, también genera una serie de 36 archivos con extensión "csv" (uno por cada experimento "base" cooperador y uno por cada experimento semi-cooperador) en el directorio "Gráficas_Degradación_Fallas" de la Figura C.4, estos archivos, permiten graficar las medidas de las propiedades estructurales y

diversas métricas cada 1 % de vértices removidos de las redes mediante fallas aleatorias.

C.4. Generales

1. Para generar imágenes que ilustren las topologías finales emergentes en cada uno de los experimentos, se escribió el script de python "3creaImagenesPNG.py", y para obtener su respectiva representación con extensión "graphml" (la cual, permite realizar análisis en software especializado en gráficas, por ejemplo, Gephi [45]), se escribió el script de python "4creaGraphML.py".

2. Los datos mostrados en las Tablas 4.1, 4.2, 4.3 y 4.4, fueron obtenidos a través del script de python "10creaTablasFinales.py", el cual, a través de diversos cálculos, realiza las operaciones estadísticas y entrega los datos en archivos con extensión "csv".

3. La Figura 4.2b, fue creada, primero, haciendo uso de la herramienta "Gephi", la cual permite calcular la modularidad de una gráfica con extensión "graphml", y ajustar el color de sus vértices con base en la partición óptima de comunidades encontrada. Una vez realizado esto, basta con exportar la gráfica con la misma extensión. Se escribieron los scripts de python "vizRingCommunities.py" (para el caso de la topología anillo) y "vizGridCommunities.py" (para el caso de la topología malla), los cuales, leen los atributos "r", "g" y "b" de los vértices de la gráfica con extensión "graphml" y los dibuja en una imagen con extensión "png".

4. Se escribieron los scripts de python "vizGridBetwenness.py" (para el caso de la centralidad de intermediación) y "vizGridCloseness.py" (para el caso de la centralidad de cercanía), los cuales, reciben una gráfica, cuya topología subyacente es una malla cuadrada, con extensión "adjlist", calculan las métricas para cada uno de sus vértices y los dibujan en una imagen con extensión "png". Las Figuras 4.8 y 4.9, fueron creadas con dichos scripts, respectivamente.

Referencias

[1] Filippo Menczer, Santo Fortunato and Clayton A. Davis: "A First Course in Network Science". *Cambridge University Press*, 2020.

[2] Duncan J. Watts and Steven H. Strogatz: "Collective dynamics of "small-world" networks". *Nature 393: 440-442*, 1998.

[3] Ricardo Marcelín Jiménez, María Elena Melgar Estrada: "Introducción a los algoritmos distribuidos". *Universidad Autónoma Metropolitana Unidad Iztapalapa*, 2014.

[4] Albert-László Barabási: "Network Science". *Cambridge University Press*, 2016.

[5] Michele Coscia: "The Atlas for the Aspiring Network Scientist". *Michele Coscia*, 2021.

[6] Duncan J. Watts: "Six Degrees: The Science of a Connected Age". *W. W. Norton & Company*, 2003.

[7] Richard Johnsonbaugh: "Matemáticas Discretas", Sexta Edición. *Pearson Educación, México*, 2005.

[8] Paul Erdös and Alfréd Rényi: "On random graphs". *Publicationes Mathematicae, 6:290-297*, 1959.

[9] Mark S. Granovetter: "The strenght of weak ties". *American Journal of Sociology, 78:1360-1380*, 1973.

[10] M. E. J. Newman: "Networks, An Introduction". *OXFORD UNIVERSITY PRESS*, 2010.

[11] Nino Boccara: "Modeling Complex Systems", Second Edition. *Springer*, 2010.

[12] Magali Alexander López Chavira, Ricardo Marcelín Jiménez: "Distributed rewiring model for complex networking: The effect of local rewiring rules on final structural properties". *PLOS ONE*, 2017.

[13] M. E. J. Newman: "The Structure and Function of Complex Networks". *Society for Industrial and Applied Mathematics, Vol.45, No. 2, pp. 167-256*, 2003.

[14] Joaquín J. Torres Agudo: "Apuntes de Física de Redes Complejas y Aplicaciones Interdisciplinarias". *Universidad de Granada*, 2021.

[15] Stanley Milgram: "The Small-World Problem". *Psichology Today, Vol. 2, pp. 61-67*, 1967.

[16] Loredana B., Luca M., Marianna La Rocca, Dominique D., Angela L., Tomasso M., Alfonso M., Sabina T., Nicola A., Roberto B.: "Predicting brain age with complex networks: From adolescence to adulthood". *NeuroImage, Vol. 225*, 2021.

[17] Albert Lázló Barabási, Réka Albert: "Emergence of Scaling in Random Networks". *Science, Vol. 286, pp. 509-512*, 1999.

[18] Gerardo A. Laguna Sánchez, Ricardo Marcelín Jiménez, Geraldine A. Patrick Encina, Gerardo Vázquez Hernández: "Complejidad y Sistemas Complejos: Un Acercamiento Multidimensional". *EditoraC3 y CopIt-arXives*, 2016.

[19] Jeff Alstott, Ed Bullmore, Dietmar Plenz: "powerlaw: A Python Package for Analysis of Heavy-Tailed Distributions". *PLOS ONE, Vol. 9, issue 1*, 2014.

[20] Martha A. Centeno, Germán Méndez Giraldo, Felipe Baesler Abufarde, Lindsay Álvarez Pomar: "Introducción a la simulación discreta". *Universidad Distrital Francisco José de Caldas, Editorial UD*, 2015.

[21] Raj Jain: "The Art of Computer Systems Performance Analysis: Techniques for experimental design, measurements, simulation and modeling". *Wiley Computer Publishing, John Wiley & sons, Inc.*, 1990.

[22] Ricardo Marcelín Jiménez, Rolando Esquivel Villafaña, Sergio Rajsbaum: "A Flexible Simulator for Distributed Algorithms". *4th Mexican International Conference on Computer Science (ENC 2003), Apizaco, México*, 2003.

[23] Jayadev Misra: "Distributed Discrete-Event Simulation". *Computing Surveys, Vol. 18, No. 1*, 1986.

[24] Jerry Banks, John S. Carson II, Barry L. Nelson, David M. Nicol: "Discrete-Event System Simulation, Fourth Edition". *Pearson*, 2005.

[25] M. De Domenico, D. Brockmann, C. Camargo, C. Gershenson, D. Goldsmith, S. Jeschonnek, L. Kay, S. Nichele, J.R. Nicolás, T. Schmickl, M. Stella, J. Brandoff, A.J. Martínez Salinas, H. Sayama: "Complexity Explained". *DOI 10.17605/OSF.IO/TQGNW*, 2019.

[26] Carlos Gershenson: "COMPLEXITY". *Instituto de Investigaciones en Matemáticas Aplicadas y Sistemas, UNAM, México, D.F.*, 2011.

[27] Carlos Gershenson: "A General Methodology for Designing Self-Organizing Systems". *Vrije Universiteit Brussel*, 2006.

[28] Mitchel Resnick: "Turtles. termites, and traffic jams: explorations in massive parallel microworlds". *MIT Press*, 1997.

[29] Von Neumman J.: "Theory of Self-Reproducing Automata". *University of Illinois Press*, 1966.

[30] Ricardo Marcelín Jiménez, Salvador González Arellano: "Apuntes de la U.E.A. "Simulación Discreta"". *Universidad Autónoma Metropolitana Unidad Iztapalapa*, 2020.

[31] Adrian Segall: "Distributed Network Protocols". *IEEE Transactions on Information Theory, Vol. IT-29, No. 1*, 1983.

[32] Evangelos Kranakis, Harvinder Singh and Jorge Urrutia: "Compass Routing on Geometric Networks". *IN PROC. 11 TH CANADIAN CONFERENCE ON COMPUTATIONAL GEOMETRY, pp. 51-54*, 1999.

[33] Baruch Awerbuch: "Complexity of Network Synchronization". *Journal of tge Association for Computing Machinery, Vol. 32, No. 4, pp. 804-823*, 1985.

[34] Ricardo Marcelín Jiménez, Rolando Esquivel Villafaña and Sergio Rajsbaum: "A Flexible Simulator for Distributed Algorithms". *Computer Science. ENC 2003, Proceedings of the Fourth Mexican International Conference, pp. 176-181, IEEE*, 2003.

[35] María Montserrat López, Tamayo Huelgas, Paloma Zubieta López: "PUENTES DE KÖNINGSBERG COMO PRIMER ACERCAMIENTO A LA TEORÍA DE GRAFOS EN MÉXICO". *Núcleo temático: IX. Comunicación y Divulgación Matemática*, 2017.

[36] Daniela Aguirre Guerrero, Mañosa A, Ll. Fàbrega, and P. Vilà: "Evaluation of Cayley Graphs for Parallel Computer Systems". *2018 International Conference on Computer, Information and Telecommunication Systems (CITS). Colmar, France*, 2018.

[37] M. Manzano, J. L. Marzo*, E. Calle, A. Manolova: "Robustness Analysis of Real Network Topologies Under Multiple Failure Scenarios". *17th European Conference on Networks and Optical Communications*, 2012.

[38] Swami Iyer, Timothy Killingback, Bala Sundaram, Zhen Wang: "Attack Robustness and Centrality of Complex Networks". *PLoS ONE 8(4): e59613. doi:10.1371/journal.pone.0059613*, 2013.

[39] Edgar Nelson Gilbert: "Random graphs". *The Annals of Mathematical Statistics, 30:1141-1144*, 1959.

[40] Juan Segovia S.: "Robustness against Large-Scale Failures in Communications Networks". *Department of Computer Architecture and Technology University of Girona, Girona, Spain*, 2011.

[41] Diego F. Rueda, Eusebi Calle, Jose L. Marzo: "Robustness Comparison of 15 Real Telecommunication Networks: Structural and Centrality Measurements". *J Netw Syst Manage DOI 10.1007/s10922-016-9391-y*, 2016.

[42] Marc Manzano, Kashif Bilal, Eusebi Calle, and Samee U. Khan: "On the Connectivity of Data Center Networks". *IEEE Communications Letters (Volume: 17, Issue: 11, November)*, 2013.

[43] Jocelyn O. Padallan: "Internet & Distributed Systems". *Arcler Press*, 2010.

[44] K. Mani Chandy, Leslie Lamport: "Distributed Snapshots: Determining Global States of Distributed Systems". *ACM Transactions on Computer Systems, Vol. 3, No. 1, pp. 63-75*, 1985.

[45] Bastian M., Heymann S., Jacomy M.: "Gephi: an open source software for exploring and manipulating networks". *AAAI Conference on Weblogs and Social Media*, 2009.

[46] Daniela Aguirre Guerrero, Roberto Bernal Jaquez: “A Methodology for the Analysis of Collaboration Networks with Higher-Order Interactions”. *Mathematics 2023, 11, 2265. https://doi.org/10.3390/math11102265*, 2023.

[47] Daniela Aguirre Guerrero: “WORD-PROCESSING-BASED ROUTING FOR CAYLEY GRAPHS”. *Doctoral Thesis, http://hdl.handle.net/10803/667410*, 2019.

Casa abierta al tiempo
UNIVERSIDAD AUTÓNOMA METROPOLITANA

ACTA DE EXAMEN DE GRADO

No. 00108
Matrícula: 2212801451

ESTUDIO DE LOS EFECTOS DE LA COOPERACÓN EN LA EVOLUCIÓN ESTRUCTURAL DE REDES COMPLEJAS.

En la Ciudad de México, se presentaron a las 14:00 horas del día 11 del mes de octubre del año 2023 en la Unidad Iztapalapa de la Universidad Autónoma Metropolitana, los suscritos miembros del jurado:

DR. ROBERTO BERNAL JAQUEZ
DR. ARMANDO CASTAÑEDA ROJANO
DRA. DANIELA AGUIRRE GUERRERO

Bajo la Presidencia del primero y con carácter de Secretaria la última, se reunieron para proceder al Examen de Grado cuya denominación aparece al margen, para la obtención del grado de:

MAESTRO EN CIENCIAS (CIENCIAS Y TECNOLOGÍAS DE LA INFORMACIÓN)

DE: JORGE ANTONIO MUÑOZ GARCIA

y de acuerdo con el artículo 78 fracción III del Reglamento de Estudios Superiores de la Universidad Autónoma Metropolitana, los miembros del jurado resolvieron:

APROBAR

Acto continuo, el presidente del jurado comunicó al interesado el resultado de la evaluación y, en caso aprobatorio, le fue tomada la protesta.

JORGE ANTONIO MUÑOZ GARCIA
ALUMNO

REVISÓ

MTRA. ROSALIA SERRANO DE LA PAZ
DIRECTORA DE SISTEMAS ESCOLARES

DIRECTOR DE LA DIVISIÓN DE CBI

Roman Linares Romero
DR. ROMAN LINARES ROMERO

PRESIDENTE

Roberto Bernal J.
DR. ROBERTO BERNAL JAQUEZ

VOCAL

DR. ARMANDO CASTAÑEDA ROJANO

SECRETARIA

DRA. DANIELA AGUIRRE GUERRERO

Con base en la facultad que me otorga el Acuerdo 9/2002 del Rector General que formaliza la creación, estructura orgánica funciones de la Dirección de Sistemas Escolares, apartado Tercero, incisos k), l) y s), y de conformidad con la información y documentos escolares que integran el expediente de **JORGE ANTONIO MUÑOZ GARCIA** con matrícula **2212801451**, la denominación correcta de la idónea comunicación de resultados es **"ESTUDIO DE LOS EFECTOS DE LA COOPERACIÓN EN LA EVOLUCIÓN ESTRUCTURAL DE REDES COMPLEJAS."**, lo cual se aclara para los efectos legales ha que haya lugar.

Mtra. Rosalía Serrano de la Paz
Directora de Sistemas Escolares