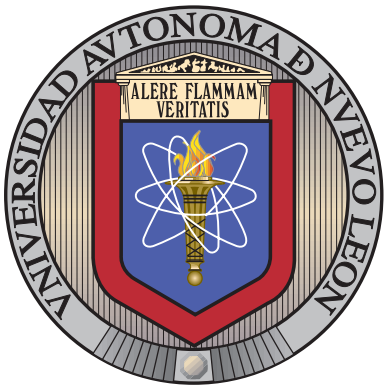
Universidad Autónoma de Nuevo León

Facultad de Ingeniería Mecánica y Eléctrica

División de Estudios de Posgrado



Análisis y optimización estructural de redes complejas

por

Perla Elizabeth Cantú Cerda

tesis

en opción al grado de

Maestro en Ciencias en Ingeniería de Sistemas

San Nicolás de los Garza, Nuevo León octubre 2009

Universidad Autónoma de Nuevo León

Facultad de Ingeniería Mecánica y Eléctrica

División de Estudios de Posgrado

Análisis y optimización estructural de redes complejas

por

Perla Elizabeth Cantú Cerda

tesis

en opción al grado de

Maestro en Ciencias en Ingeniería de Sistemas

San Nicolás de los Garza, Nuevo León octubre 2009

Universidad Autónoma de Nuevo León

Facultad de Ingeniería Mecánica y Eléctrica

División de Estudios de Posgrado

Los miembros del Comité de Tesis recomendamos que la Tesis «Análisis y optimización estructural de redes complejas», realizada por el alumno Perla Elizabeth Cantú Cerda, con número de matrícula 1052177, sea aceptada para su defensa como opción al grado de Maestro en Ciencias en Ingeniería de Sistemas.

El Comité de Tesis

Dra. Satu Elisa Schaeffer
Asesor

Dra. Laura Cruz Reyes
Revisor

Dr. Roger Z. Ríos Mercado
Revisor

Vo. Bo.

Dr. Moisés Hinojosa Rivera
Subdirector
División de Estudios de Posgrado

San Nicolás de los Garza, Nuevo León, octubre 2009

Este trabajo lo dedico a las personas que siempre han estado conmigo y que sin su confianza y apoyo incondicional no hubiera sido posible llegar hasta este momento.

A mis padres,

La Sra. Victoria Cerda Ramírez y

el Sr. José Antonio Cantú Aguilar,

y mis hermanos,

Victoria Cantú Cerda,

José Antonio Cantú Cerda y

Alejandro Cantú Cerda.

Índice general

Índice de figuras

Índice de tablas

Agradecimientos

Agradezco primeramente a Dios por darme la oportunidad de estar aquí y permitirme apreciar lo bello que es la vida.

A mis padres por la educación que me brindaron, por su sacrificio cotidiano, por sus regaños, sabios consejos y sobre todo por sus buenos ejemplos. Gracias a ellos soy una mejor persona cada día.

A mis hermanos que siempre han estado conmigo en todo momento y con quienes he creado mis más gratos recuerdos.

A la Dra. Satu Elisa Schaeffer, mi tutora y asesora de tesis, quien siempre me apoyó y dispuso de su tiempo para atender mis dudas y expresarme sus valiosos comentarios.

A los miembros del comité de tesis, la Dra. Laura Cruz y el Dr. Roger Z. Ríos, que estuvieron involucrados en el desarrollo de este trabajo. Gracias por su tiempo, comentarios y observaciones para la mejora del mismo.

A mis compañeros de generación del PISIS por haberme apoyado durante mi estancia en el programa.

A los profesores del PISIS por haberme orientado durante todo el tiempo que estuve como alumna y a la coordinación del programa por permitirme pertenecer a él.

A la Universidad Autónoma de Nuevo León y la Facultad de Ingeniería Mecánica y Eléctrica por haberme otorgado beca de inscripción y colegiatura. Al Consejo Nacional de Ciencia y Tecnología (CONACYT) por haberme otorgado una beca de manutención durante la duración de la maestría. Gracias por hacer posible mis estudios de maestría.

A los proyectos PAICYT CA1475-07, PROMEP 103,5/07/2523 y PROMEP 103,5/08/4804 a través de los cuales se obtuvo financiamento para copias, impresión de pósters, folletos y apoyo para asistencia a congresos nacionales. También agradezco a la Facultad de Ingeniería Mecánica y Eléctrica (FIME), por su apoyo para asistir al 6º Congreso Internacional sobre Innovación y Desarrollo Tecnológico (CIINDET).

A todos, gracias.

Resumen

Perla Elizabeth Cantú Cerda.

Candidato para el grado de Maestro en Ciencias en Ingeniería de Sistemas.

Universidad Autónoma de Nuevo León.
Facultad de Ingeniería Mecánica y Eléctrica.

Título del estudio:

Análisis y optimización estructural de redes complejas

Número de páginas: 123.

Objetivos y método de estudio: El objetivo del presente trabajo de tesis es la creación de una herramienta de software que optimice la calidad estructural de una red compleja. Existen varios modelos de redes complejas como las Aleatorias, Libre de escala y de Mundo pequeño que representan sistemas del mundo real, por lo cual se pretende que la herramienta pueda trabajar con cualquiera de ellas.

Para mejorar la calidad estructural de una red la herramienta de software realiza un análisis a la estructura topológica y a través de un algoritmo genético realiza su optimización. La herramienta de software desarrollada no tiene definida una calidad estructural ya que las redes funcionan y se conforman de diferentes

maneras para cumplir con su funcionamiento. Así la calidad estructural de una red la define el usuario en base a la aplicación de la red y las propiedades estructurales deseadas.

Como objetivo secundario se busca encontrar los parámetros recomendados para el algoritmo genético que producen el conjunto de redes alternativas de mejor calidad y sin un costo excesivo. Los parámetros que se buscan determinar son el tamaño de la población inicial, la cantidad de aristas mutadas y la cantidad de iteraciones durante las cuales se recomienda ejecutar la herramienta de software.

CONTRIBUCIONES Y CONCLUSIONES: La contribución de este trabajo es la creación de una herramienta de software de fácil uso y código abierto que opera en ambiente Windows y ambientes basados en Unix.

Se realizó un estudio sobre los tiempos de ejecución que toma la herramienta de software en optimizar la calidad en redes de diferentes tamaños y se identificó la propiedad estructural de la herramienta de software que toma un mayor tiempo de optimizar.

Se observó que para las instancias de prueba desarrolladas en este trabajo, la metodología utilizada brinda soluciones robustas de buena calidad en un tiempo de cómputo razonable. La herramienta desarrollada arroja como soluciones un conjunto de redes alternativas con una mayor calidad y con el menor costo que encontró el algoritmo genético durante su ejecución. Se decidió mostrar un conjunto de soluciones para otorgar libertad al tomador de decisiones de elegir la solución que más le agrade en base a criterios como el costo y el porcentaje de mejora obtenido.

Firma del asesor: _____

Dra. Satu Elisa Schaeffer

Capítulo 1

Introducción

Vivimos en un mundo de *sistemas*, de hecho somos parte de un sistema y estamos formado por sistemas. Un sistema se puede definir como la interacción de elementos para lograr un objetivo en común [4]. Ejemplos de sistemas son la Internet, las neuronas de nuestro cerebro, circuitos electrónicos, redes de computadoras, redes de comunicaciones, redes de transporte y hasta el metabolismo de una célula [10]. Estos sistemas y muchos otros más se pueden representar a través de grafos, ya que un *grafo* es una representación abstracta de situaciones formadas por elementos (*vértices*) y las interacciones entre ellos (*aristas*) [23]. Un ejemplo de esto es la Internet que se puede modelar como un grafo donde las computadoras son representadas por vértices y el cableado de las computadoras por aristas.

Existe una clase de sistemas denominada *redes complejas* [53], las cuales son el tema principal de este trabajo. A continuación se describe el problema que se aborda.

1.1 Descripción del problema

Xu [59] menciona que existen una gran cantidad de problemas en la ciencia, la ingeniería, la medicina, la tecnología y otras áreas donde el resolver ciertos problemas requiere un gran tiempo de cómputo y en la mayoría de las ocasiones se requiere una respuesta en un tiempo pequeño, por ejemplo, el procesamiento de imágenes, el procesamiento de señales de un radar, el pronóstico del tiempo, el control rápido

de reacciones químicas o nucleares y simulaciones aerodinámicas. Para resolver tales problemas es necesario tener un *sistema de múltiples procesadores* (MPS) el cual sea rápidamente capaz de ejecutar algoritmos en paralelo, sin embargo; la rapidez de los procesadores está limitada por su mecanismo, por lo cual se deben construir mejores supercomputadoras que logren incrementar la cantidad de procesamiento de información [59].

Para construir supercomputadoras se necesita de diseñadores que construyan sistemas confiables, altamente efectivos y económicos [59]. Para que tales sistemas sean económicos deben tener un buen diseño en la estructura de las interconexiones de sus componentes. La estructura topológica de la conexión de los componentes de una red juega el papel más importante en el diseño de los supersistemas debido a que cada conexión entre los componentes está asociado a un costo y a un determinado funcionamiento. Un alto rendimiento y un bajo costo son las principales metas en el desarrollo o diseño de cualquier red [59].

Existen redes en las cuales la estructura topológica de sus componentes no es trivial y además están formadas por un gran número de elementos, estas redes son conocidas como *redes complejas*. La importancia de estas redes radica en que a través de ellas se modelan sistemas tecnológicos, biológicos y sociales del mundo real.

Actualmente existen herramientas de software que analizan y visualizan redes complejas como *Network Workbench* [7] y *Pajek* [9], pero no otorgan recomendaciones al tomador de decisiones sobre que acciones se pueden realizar para mejorar la calidad estructural de la red. Estas herramientas son de ayuda, pero existen situaciones en las que se requieren respuestas precisas, como por ejemplo determinar si una red es robusta, conocer si la estructura es óptima y determinar si es factible mejorar el desempeño de la red.

Al revisar la literatura se ha detectado el problema de no encontrar herramientas de software que optimicen la estructura de redes complejas en base a las necesidades o requerimientos del usuario, ya que en los últimos años se han dedicado

esfuerzos para la creación de herramientas que analizan y visualizan las conexiones de las redes complejas y éstas han demostrado no realizar alguna optimización a la estructura de la red (este tema se aborda más adelante en la Sección 2.4.2).

1.2 Antecedentes

A finales de la década de los cincuentas y principios de la decada de los sesentas, el estudio de la estructura de redes con conexiones aleatorias (*teoría de grafos aleatorios*) fue un tema de gran interés para investigadores de diversas ciencias como las matemáticas, ciencias de la computación, comunicaciones, biología, sociología y economía entre otras [24]. Desafortunadamente, en ese entonces no pudo ser comprobada en su totalidad la teoría que se había desarrollado.

Fue hasta los finales de los noventas que el estudio de la evolución y la estructura topológica de las redes llegó a ser un nuevo campo de estudio para la física estadística. Esto se debió en medida al crecimiento impactante en la arquitectura de las redes de comunicaciones, la Internet y la World Wide Web (WWW) [24]. Es a partir de este crecimiento que se han desarrollado diferentes modelos de redes complejas, tales como las *Aleatorias* [29], *Exponenciales* [3], *Libre de escala* [11] y *Mundo pequeño* [56]. Cada uno de estos modelos tiene propiedades estructurales y una distribución del grado que las caracteriza.

La clasificación de redes complejas dentro de un modelo ayuda a conocer y entender su comportamiento, pero no otorga un ánalisis de la calidad estructural en base a la aplicación de la red.

1.3 Objetivo

En este trabajo de tesis el objetivo es desarrollar una herramienta de software que optimice las propiedades estructurales de una red compleja al realizar *cambios en su topología*. De esta manera se otorga al usuario el conjunto de las mejores redes

alternativas construidas, las cuales se caracterizan por tener una mayor calidad que la red original sin un *costo* excesivo.

Como objetivos secundarios se pretende que la herramienta de software funcione para redes complejas Aleatorias, Libre de escala y Mundo pequeño, además de establecer los parámetros iniciales bajo los cuales la herramienta de software encuentre las mejores soluciones posibles.

1.4 Motivación y justificación

En el mundo, cualquier sistema se puede visualizar como un grafo y por tanto se observa qué elementos están relacionados, qué elementos son los más importantes y qué pasaría si dejaran de existir algunos componentes [16]. Claro está que lo anterior es posible cuando el sistema es de una magnitud pequeña ya que si el sistema resulta ser una red compleja entonces se vuelve difícil el poder identificar algunos elementos y sus propiedades básicas.

Estudiar la topología de las redes es de importancia ya que conociendo sus características y debilidades se pueden manejar de la mejor forma diversas situaciones, por ejemplo, aprovechar las conexiones de un vértice para realizar y evitar ataques, evitar la propagación de una epidemia en una red social, actuar a tiempo en desastres naturales al crear rutas de evacuación y otras más. Al conocer las propiedades estructurales de una red se podría responder a cualquier situación con un costo bajo y con la seguridad de haber tomado la mejor decisión dada la estructura de la red.

Un ejemplo de un sistema que necesita ser mejorado y optimizado es el *drenaje pluvial* en donde se puede representar cada alcantarilla como un vértice y cada calle como una arista. La red pluvial representada como un grafo es frágil ante un ataque, que bien es la cantidad de lluvia que la mayoría de las veces supera la capacidad de la red. Una solución a esta situación podría ser colocar más alcantarillas en los lugares donde se cree que son los mejores, pero esta acción no soluciona del todo

el problema ya que sería una solución temporal y con el tiempo ineficiente debido a diversos factores como el crecimiento de la población y la creación de nuevas viviendas. Por tanto la solución radica en estudiar el problema y atacarlo de manera estratégica en el presente y a largo plazo a través de la teoría de grafos.

El desarrollo de este trabajo fue motivado por la necesidad de mejorar redes con una gran cantidad de elementos, ya que solo habían sido desarrolladas herramientas que analizaban y visualizaban las conexiones de una red, por ejemplo, Network Workbench [7] y Pajek [9], pero hasta donde se tiene conocimiento éstas no otorgan una asesoría para optimizar el funcionamiento de la red.

1.5 Hipótesis

La hipótesis de este trabajo es que la herramienta de software construye un conjunto de redes que satisfacen las necesidades de un usuario al mejorar las propiedades estructurales de una red existente.

1.6 Estructura de la tesis

El presente trabajo de tesis está organizado de la siguiente manera.

El Capítulo 2 contiene los antecedentes acerca de las diferentes áreas y conocimientos que se conjugan en el desarrollo del presente trabajo. Éstas son la teoría de grafos, la teoría de redes complejas, las propiedades estructurales, las métricas estructurales, los modelos de redes complejas, los algoritmos genéticos y la optimización.

En el Capítulo 3 se abordan las propiedades estructurales que se optimizan en este trabajo, se muestra la relación de las métricas a utilizar según las propiedades estructurales que se deseén optimizar, se explica la función objetivo utilizada en este trabajo y se hace la definición formal del problema.

En el Capítulo 4 se especifica la metodología que sigue la herramienta de software para realizar la optimización en la topología de una red.

En el Capítulo 5 se reporta la experimentación computacional, las condiciones y las instancias bajo las cuales se realizaron todos los experimentos. Como se tienen varios objetivos por cumplir se realizaron los experimentos correspondientes para encontrar los mejores parámetros posibles de la herramienta de software que determinan o aseguran el conjunto de soluciones con mayor calidad. Los parámetros que se buscaron encontrar son la cantidad de mutaciones, la cantidad de iteraciones y el tamaño de la población inicial. En esta sección se demuestra la confiabilidad de los resultados y por último se muestra un estudio sobre los tiempos de ejecución que tarda la herramienta en realizar la optimización en grafos de diferentes tamaños.

En el Capítulo 6 se muestran las conclusiones obtenidas con el desarrollo del presente trabajo de investigación. Se presentan además las aportaciones realizadas y las posibles líneas de investigación futura para ampliar el conocimiento sobre la optimización de redes complejas.

Algunos modelos generadores de redes complejas y el funcionamiento del algoritmo genético se explican en los Apéndices A y B, respectivamente.

Capítulo 2

Marco teórico y antecedentes

Para ofrecer una red con una estructura topológica diferente y mayor calidad que la red original, se necesita estudiar las propiedades estructurales de la red y cuantificarlas a través de métricas, tales como el diámetro del grafo, la eficiencia, el grado de cada uno de los vértices, la vulnerabilidad y otras más. El conocimiento de estos antecedentes no bastan para lograr el objetivo del presente trabajo, ya que se deben tener conocimientos de otras áreas tales como la teoría de grafos, la teoría de redes complejas, la optimización (monobjetivo y multiobjetivo) y los algoritmos genéticos.

En este capítulo se presenta la información teórica de las diferentes áreas que se utilizaron en la metodología propuesta en este trabajo y se presentan también los antecedentes de la optimización estructural en redes complejas.

2.1 Teoría de grafos

Un *grafo* (G) es un par de conjuntos (V, E), donde V es un conjunto finito de puntos llamados *vértices* o *nodos* que usualmente se enumeran para facilitar su localización $V = \{1, 2, 3, \ldots, n\}$ y E es un conjunto finito de *aristas* que unen pares de vértices $\{i, j\}$ donde $i, j \in V$. A cada arista se le puede asignar un costo, peso o ponderación que se denomina *distancia física* ($\ell_{i,j}$). Por lo general n representa el número de los vértices del grafo (*orden*) y m representa el número de aristas del grafo (*tamaño*).

En un grafo, si las aristas tienen dirección del vértice i (punto inicial) al vértice j (punto final), entonces se dice que es un *grafo dirigido*. Si no existe dirección en las aristas de un grafo entonces se dice que es un *grafo no dirigido*. Cuando un grafo G tiene entre cada par de vértices una arista se dice que es un *grafo completo* (G_n). En un grafo completo dirigido existen $n(n-1)$ aristas mientras que en un grafo completo no dirigido existen $n(n-1)/2$ aristas. En esta tesis se trabaja con grafos no dirigidos por lo cual cada que se haga referencia a un grafo se estará omitiendo en su descripción que se trata de un grafo no dirigido.

Un ejemplo de un grafo completo dirigido y no dirigido de cuatro vértices se muestra en la Figura 2.1.

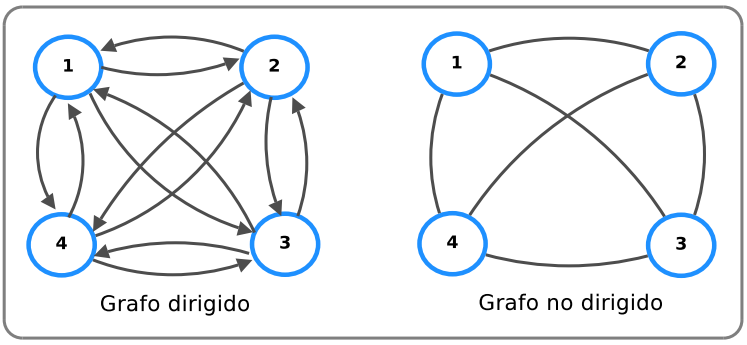


Figura 2.1: Tipos de grafos completos. Un grafo completo dirigido de cuatro vértices tiene doce aristas, mientras que en un grafo completo no dirigido tiene seis aristas.

Si existe una arista que conecte a dos vértices se dice que los vértices son *adyacentes*. El número de aristas adyacentes a un vértice i determina el *grado* de ese vértice y al conjunto de vértices adyacentes a i se le conoce como el *vecindario* de i. Si dos vértices i y j no son adyacentes, pueden estar conectados por una secuencia de aristas, al conjunto de aristas que forman la secuencia se le denomina la *ruta* de i a j, siendo la suma de los pesos de las aristas la *longitud de la ruta*. A la ruta entre los vértices i y j con la mínima longitud posible se le conoce como la *distancia geodésica* ó *ruta más corta* ($\text{dist}_{i,j}$). Existen varios algoritmos para obtener la ruta más corta, entre ellos se encuentran los algoritmos de Dijkstra y Floyd-Warshall [1].

Una *matriz de adyacencia* (A) es una matriz cuadrada de orden $n \times n$, en donde se representa la presencia o ausencia de aristas entre un par de vértices. Si existe una arista entre los vértices i y j entonces $A(i, j) = 1$, en caso contrario $A(i, j) = 0$ [36]. Para grafos no dirigidos la matriz de adyacencia es simétrica $A(i, j) = A(j, i)$.

2.2 Teoría de redes complejas

Las redes complejas se caracterizan por tener un gran número de elementos, una complejidad en su interacción y una estructura topológica (topología) no trivial porque las conexiones entre los elementos siguen una distribución del grado [4]. Algunos ejemplos de redes complejas son la World Wide Web, Internet, redes biológicas, neuronales, proteínas, reacciones metabólicas de células, telecomunicaciones, correo electrónico, eléctricas, energía, circuitos, terroristas, drenaje pluvial, carreteras y sociales entre otras [10]. Todas estas redes complejas y otras más se pueden modelar a través de un grafo, donde los componentes de las redes complejas se representan cómo vértices y las conexiones entre ellos cómo aristas. Un ejemplo de dos sistemas modelados como redes complejas se muestra en la Figura 2.2.

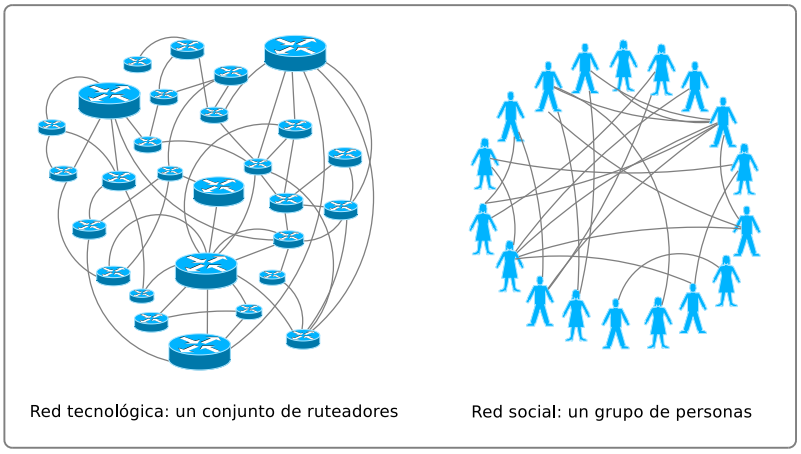


Figura 2.2: La red tecnológica es un conjunto de ruteadores representados por vértices y el cableado físico que los une se representa por medio de aristas. La red social está formada por personas y se representan por medio de vértices mientras que las interacciones entre ellos se representan por medio de aristas.

2.2.1 Estructura topológica de redes complejas

La *estructura topológica* o *topología* de una red, consiste en el *patrón* de conexión entre los vértices y aristas [54]. En la topología de una red, no afecta la manera en que fueron enumerados o nombrados los vértices ya que son solo una manera de diferenciarlos. Otra cuestión importante es que no se debe confundir la topología de la red con la figura que forman los vértices y las aristas ya que éstos pueden dibujarse de manera diferente, pero siguen modelando la misma red.

A los grafos que tienen el mismo patrón de conexión entre sus vértices, pero están representados o dibujados de manera diferente se les denominan *isomorfos*. Un ejemplo de grafos isomorfos construidos con Network Workbench [7] se observa en la Figura 2.3.

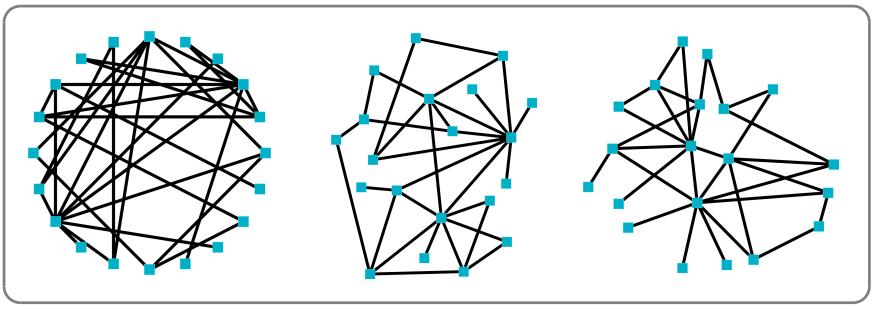


Figura 2.3: Grafos isomorfos: los tres grafos tienen 20 vértices, 31 aristas y el mismo patrón topológico, pero fueron dibujados de diferente manera para representar al mismo grafo.

Existen varias redes complejas o grafos que comparten el mismo patrón topológico pero dependiendo de la aplicación del grafo, es decir, del objetivo que se desee lograr, el conjunto de las propiedades requeridas para lograr la calidad en un grafo es diferente. Por ejemplo, imagine que tiene dos redes complejas con la misma topología, la primera es una red de telecomunicaciones y la otra una red social. En la red de telecomunicaciones se desearía que los mensajes enviados lleguen al mayor número de personas, pero en una red social si una persona estuviese contagiada

con un virus se desearía que esta enfermedad no se propagara a un mayor número de personas. Por lo cual mientras en una red se desearía maximizar la difusión de un mensaje, en la otra se desearía minimizar la difusión de una enfermedad. Por tanto dos redes pueden compartir la misma topología pero la aplicación de la red determina que propiedades estructurales son deseables.

2.2.2 Propiedades estructurales

Algunas de las características o propiedades que son relevantes en la topología de un grafo son:

Asortatividad o mezclado, $\tilde{Q}$. La asortatividad se refiere a la tendencia de que vértices con un atributo s tengan conexiones con vértices del mismo atributo s [19]. Ejemplos de los atributos que se pueden asignar a un vértice son el grado de conexión, costo, capacidad, eficiencia y la vulnerabilidad.

El *coeficiente de asortatividad* se define como

$$\tilde{Q} = \frac{\sum_s P^{(\text{tipo})}(s|s) - 1}{N_T - 1}, \tag{2.1}$$

donde N_T es el número de los diferentes tipos de vértices en la red.

La probabilidad de que un vértice del tipo s tenga un vecino del tipo t es

$$P^{(\text{tipo})}(t|s) = \frac{\hat{\text{e}}_{st}}{\sum_u \hat{\text{e}}_{su}}, \tag{2.2}$$

donde $\hat{\text{e}}_{st}$ es el número de aristas que conectan vértices del tipo s a vértices del tipo t. Cabe recordar que $\sum_t P^{(\text{tipo})}(t|s) = 1$.

Un caso especial de la asortatividad es de acuerdo al grado del vértice, conocido como el *grado de correlación* [45]. Estudiar la *asortatividad por grado* es de gran interés porque puede dar lugar a efectos interesantes en la estructura de las redes. Una forma de obtener el grado de correlación es a través del *Coeficiente de correlación de Pearson* (r) [44] de los grados en ambos extremos de las

aristas, que se define como

$$r = \frac{m^{-1}\sum_i j_i k_i - [m^{-1}\sum_i \frac{1}{2}(j_i + k_i)]^2}{m^{-1}\sum_i \frac{1}{2}(j_i^2 + k_i^2) - [m^{-1}\sum_i \frac{1}{2}(j_i + k_i)]^2}, \tag{2.3}$$

donde j_i y k_i son los grados de los vértices al final de la i-ésima arista e $i = 1, \ldots, m$, donde m es el número total de aristas.

Si $r > 0$ entonces la red es asortativa, si $r < 0$ la red es desasortativa y si $r = 0$ entonces no hay correlación entre los grados de los vértices.

Conexidad, Γ. Un grafo es conexo si existe una ruta de un vértice i a cualquier otro vértice j [59]. A través de la matriz de adyacencia del grafo se puede determinar si éste es conexo o no.

Eficiencia, $\mathcal{E}$. La eficiencia cuantifica el desempeño del intercambio de información en una red [36]. La eficiencia $\mathcal{E}_{ij}$ de la comunicación entre dos vértices i y j es definida como el inverso proporcional a la ruta más corta

$$\mathcal{E}_{ij} = \frac{1}{\text{dist}_{ij}}, \forall i, j \in V. \tag{2.4}$$

Cuando el grafo es desconexo y por tanto no existe una ruta entre los vértices $\{i, j\}$, la distancia entre estos vértices se considera como infinita y la eficiencia es igual a cero.

La *eficiencia promedio* del grafo $\mathcal{E}(\text{G})$ es definida como

$$\mathcal{E}(G) = \frac{\sum_{i,j \in V:\, i \neq j} \mathcal{E}_{ij}}{n(n-1)} = \frac{1}{n(n-1)} \sum_{i,j \in V:\, i \neq j} \frac{1}{\text{dist}_{ij}}, \tag{2.5}$$

donde $n(n-1)/2$ son las aristas del grafo. La eficiencia promedio del grafo se puede tomar como la *eficiencia global* del grafo $\mathcal{E}_{\text{glob}}$, la cual cuantifica la eficiencia de la red a la hora de mandar información entre los vértices de la red.

La *eficiencia local* $\mathcal{E}_{\text{loc}}$ es un indicador de la vulnerabilidad o tolerancia a fallas de una red [36] y se puede definir como la eficiencia promedio de los subgrafos locales

$$\mathcal{E}_{\text{loc}} = \frac{1}{n} \sum_{i \in V} \mathcal{E}(G_i), \tag{2.6}$$

donde $\mathcal{E}(G_i)$ es la eficiencia de cada vértice del subgrafo $G_i = (V_i, E(V_i))$.

Entropía de la distribución del grado, H. La *entropía* es un concepto de termodinámica, mecánica estadística y teoría de la información que está relacionado con la cantidad de desorden e información en un sistema [19]. En la teoría de la información, entropía describe qué tanta aleatoridad esta presente en eventos que se suponen son aleatorios.

Entropía de la distribución del grado provee una medida promedio de la heterogeneidad de la red, la cual puede ser definida como

$$H = -\sum_k P(k)\log P(k), \tag{2.7}$$

donde $P(k)$ es la fracción de vértices en la red con grado k.

La entropía de redes ha sido relacionada a la robustez de una red (la resistencia a ataques) y el grado de dificultad que existe cuando se busca información en una red.

Grado de intermediación, $\sigma(p)$. El grado de intermediación se refiere a que entre más veces aparezcan vértices y aristas en una ruta, se considera que esos vértices y aristas son los elementos importantes de la red. Formalmente la *intermediación* de un vértice n es el número total de rutas más cortas entre todos los posibles pares de vértices que pasan a través del vértice n. Si el número de rutas más cortas entre un par de vértices es mayor a uno, entonces la ruta que pasa a través del vértice n contribuye a la intermediación con un correspondiente peso reducido.

Si el número total de las rutas más cortas entre los vértices i y j es mayor que cero y estas rutas pasan a través de un vértice p, entonces el radio $B(i,p,j)/B(i,j)$ muestra qué tan importante es el vértice p en la conexión entre los vértices i y j. Si, por ejemplo, ninguna de las $B(i,j)$ rutas más cortas pasa a través del vértice p, entonces la importancia del vértice p para i y j es cero. En cambio, si todas las $B(i,j)$ rutas pasan a través de p, entonces la

importancia de p es grande ya que el vértice p controla las conexiones de las rutas más cortas. Así la *intermediación* de un vértice p es

$$\sigma(p) \sim \sum_{\substack{i,j \in V : i \neq j \\ B(i,j) > 0}} \frac{B(i,p,j)}{B(i,j)}. \tag{2.8}$$

Los vértices con un alto grado de intermediación controlan a la red. Por tanto, es natural sugerir que la intermediación de un vértice está estrechamente correlacionada con el grado del vértice.

El grado de intermediación cuantifica la importancia de un vértice en el tráfico de la red y por tanto puede servir como un indicador de la resistencia de la red al remover un vértice ya que indica que tan eficiente es una ruta cuando se elimine el vértice i. El grado de intermediación también ayuda a identificar la formación de cuellos de botella en la red.

Percolación, ρ. La percolación concierne al movimiento y filtrado de fluidos a través de materiales porosos [2]. Para tener un conocimiento real de la red Dorogovtsev y Mendes [24] mencionan que se deben conocer las propiedades de percolación ya que la distribución del grado, el coeficiente de agrupamiento y el grado de intermediación no son siempre suficientes para tener un conocimiento global de la estructura topológica. La teoría de percolación estudia la aparición de rutas que se filtran a través de una rejilla[1], es decir, que inician en un extremo de la rejilla y terminan en el extremo opuesto. Para una pequeña probabilidad p pocas aristas forman la ruta y entonces un pequeño grupo de vértices se conectan por las aristas que los conforman. Existe una probabilidad crítica denominada *umbral de percolación* p_c en la cual aparece una gran cantidad de aristas que conectan a los vértices [2]. A este grupo se le conoce como el *aglomerado infinito* (*infinite cluster* en inglés) porque su tamaño varia con el tamaño de la rejilla.

[1]Una rejilla o trama (*lattice* en inglés) es un conjunto de puntos que forman una red o enrejado regular entre líneas paralelas.

El principal interés de la teoría de percolación consiste en conocer:

- La probabilidad de percolación P, la cual denota la probabilidad de que un vértice pertenezca a el aglomerado infinito y es definida como

$$P = P_p(|C| = \infty) = 1 - \sum_{s<\infty} P_p(|C| = s), \tag{2.9}$$

  donde $P_p(|C| = s)$ denota la probabilidad de que el aglomerado en el origen tenga un tamaño s.

- El tamaño promedio del aglomerado $\langle t \rangle$ que es definido como

$$\langle t \rangle = E_p(|C|) = \sum_{s=1}^{\infty} sP_p(|C| = s). \tag{2.10}$$

- La distribución del tamaño del aglomerado n_s, que define la probabilidad de que un vértice tenga una posición fija dentro de un aglomerado de tamaño s y se obtiene con

$$n_s = \frac{1}{s} P_p(|C| = s). \tag{2.11}$$

La teoría de percolación es definida en una rejilla d-dimensional pero los resultados obtenidos dentro del contexto de percolación aplican directamente sobre las redes Aleatorias [2]. La idea básica de la percolación en redes Aleatorias es la existencia de una fuerte transición en la cual la conectividad global aparece o desaparece abruptamente cuando cierto parámetro de densidad alcanza un valor crítico. Dorogovtsev y Mendes [24] mencionan que los procesos complejos de difusión y epidemiología pueden ser abordados por la teoría de percolación.

Robustez, R. La robustez de una red es también identificada como la resistencia o tolerancia a fallas. Es un concepto que involucra el patrón de descomposición que se forma al eliminar un vértice o arista del grafo [50]. Una manera de cuantificar la robustez de una red es a través de la *conectividad* la cual es el menor número de aristas que produce una falla en el sistema [59].

Vulnerabilidad, Υ. La vulnerabilidad de un vértice al asociarla con el funcionamiento de una red se puede definir como la caída del desempeño cuando el

vértice y todas sus aristas son eliminadas. En la estructura topológica de una red, es importante conocer cuales son los componentes o vértices más importantes para su mejor funcionamiento. Intuitivamente los vértices críticos de la red son los *hubs* (vértices con un grado alto), sin embargo; hay situaciones en las cuales los hubs no son necesariamente los de mayor importancia [19]. Por ejemplo, en un árbol binario[2] los vértices tienen un grado similar, por lo tanto no existen hubs, pero si se desconectan los vértices más cercanos a la raíz o mejor aún si se elimina la raíz, entonces se tiene un mayor impacto o falla en la red. Una manera de encontrar los componentes o vértices más críticos en la red es buscando los vértices más vulnerables. La vulnerabilidad de un vértice i se define como

$$\Upsilon_i = \frac{\mathcal{E}_{\text{glob}} - \mathcal{E}_i}{\mathcal{E}_{\text{glob}}}, \tag{2.12}$$

donde $\mathcal{E}_{\text{glob}}$ es la eficiencia global de la red original y $\mathcal{E}_i$ es la eficiencia global después de remover el vértice i y todas sus aristas.

Una forma de medir la *vulnerabilidad* de la red es obteniendo la máxima vulnerabilidad para todos sus vértices:

$$\Upsilon(G) = \max_{i \in V} \Upsilon_i. \tag{2.13}$$

2.2.3 Métricas: medidas de calidad

Una *métrica* es una función no negativa $g(x, y)$ que describe la distancia entre los puntos vecinos de un conjunto dado [31]. Una métrica se caracteriza por ser simétrica $g(x, y) = g(y, x)$, reflexiva $g(x, x) = 0$ y por satisfacer la desigualdad del triángulo que es denotada como

$$g(x, y) + g(y, z) \geq g(x, z). \tag{2.14}$$

En este trabajo cuando se habla de una métrica no se hace referencia a su contexto original, ya que una métrica es utilizada para proveer en términos cuan-

[2]En teoría de grafos un árbol binario es un grafo conexo, acíclico y no dirigido tal que el grado de cada vértice no es mayor a tres, tiene un hijo a la derecha y un hijo a la izquierda.

titativos los valores de las propiedades estructurales. A continuación se presentan algunas de las métricas utilizadas en la teoría de redes complejas [19].

Conectividad, $\Phi(G)$. La conectividad $\Phi(G)$ y la conectividad de las aristas $\lambda(G)$ de G en la estructura topológica de una red, es el componente de menor cardinalidad (conexiones) que produce una caída o falla en el sistema que pone en peligro la comunicación de la red [59]. En otras palabras, para permanecer la red conectada puede simultáneamente tolerar ($\Phi(G)$ - 1) fallos en los componentes o (λ(G) - 1) fallas en las aristas. Así entre mayor sea la conectividad más robusta es la red.

Coeficiente de agrupamiento, C. El coeficiente de agrupamiento cuantifica la presencia de un número de triángulos en la red (conjuntos de tres vértices donde cada uno de ellos está conectado a cada vértice) [45]. Una propiedad común que se encuentra en las redes sociales es que si un vértice A está conectado a un vértice B y éste vértice se conecta a un vértice C, entonces la probabilidad de que el vértice A también esté conectado al vértice C es grande (por ejemplo, es probable que el amigo de tu amigo también sea tu amigo). Lo anterior es una tendencia de transitividad o agrupamiento la cual es cuantificada a través del coeficiente de agrupamiento, también conocido como *fracción de transiciones triples* en Albert y Barabási [2] o *densidad de la red* en Newman [45] en el cual el coeficiente de agrupamiento caracteriza la densidad de las conexiones cercanas a un vértice.

Watts y Strogatz [56] definen el coeficiente de agrupamiento como

$$C_i = \frac{2E_i}{k_i(k_i - 1)}, \tag{2.15}$$

donde E_i es el número total de aristas que conectan a los vecinos más cercanos y $k_i(k_i - 1)/2$ es el número total de todas las aristas posibles entre todos los vecinos más cercanos.

El coeficiente de agrupamiento de todo el grafo $C(G)$ es el promedio del coeficiente de agrupamiento de cada vértice $i \in V$, que se denota como

$$C(G) = \frac{\sum_{i \in V} C_i}{n}. \tag{2.16}$$

El agrupamiento es una forma específica de correlaciones en una red. Si el coeficiente de agrupamiento de una red infinita no se acerca a cero, entonces existen correlaciones entre los vértices de la red. El coeficiente de agrupamiento de un grafo completo es igual a uno.

Grado mínimo δ, máximo Δ y promedio $\langle k \rangle$. El grado de un vértice indica el número de aristas que son incidentes a un vértice i [24]. Es una característica importante porque a través de ella se pueden derivar varias métricas.

Longitud de la ruta más corta característica, L. La longitud de la ruta más corta característica es el promedio de las distancias geodésicas y se determina como

$$L(G) = \frac{\sum_{i,j \in V:\, i>j} \mathrm{dist}_{ij}}{\frac{1}{2}n(n-1)} \sim \frac{\sum_{i,j \in V:\, i \neq j} \mathrm{dist}_{ij}}{n(n-1)}. \tag{2.17}$$

Diámetro, D. El diámetro de una red es la longitud más grande de las *distancias geodésicas* por lo que se determina como

$$D(G) = \max_{i,j \in V:\, i \neq j} \mathrm{dist}_{ij}. \tag{2.18}$$

El diámetro también puede ser determinado en base al número de aristas entre el par de vértices que se encuentran más alejados en la red y es un importante parámetro para determinar el funcionamiento de la topología ya que cuantifica el retardo de la transmisión de un mensaje. Por ejemplo, si se desea mejorar o incrementar la eficiencia de la transmisión de un mensaje, entonces se debe minimizar el diámetro. Conocer el diámetro ayuda a conocer el funcionamiento de la red en el peor de los casos [59].

Distancia promedio, DP**.** La distancia promedio de un grafo G es el promedio aritmético de todas las distancias físicas diferentes a cero y se determina como

$$DP(G) = \frac{1}{n(n-1)} \sum_{i,j \in V : i \neq j} \ell_{ij}. \tag{2.19}$$

El valor promedio de todas las distancias entre pares de vértices puede ser una medida más precisa sobre el funcionamiento promedio de una red que el diámetro [59].

Distribución del grado, $P(k)$**.** La distribución del grado indica la probabilidad de que un vértice i en el grafo de tamaño n tenga k conexiones con otros vértices (k vecinos más cercanos). Se denota como $P(k, i, n)$.

Si se conoce la distribución del grado de cada vértice en la red, entonces se puede obtener la *distribución del grado total* que se define como

$$P(k, N) = \frac{1}{n} \sum_{i=1}^{n} p(k, i, n). \tag{2.20}$$

La distribución del grado de un vértice caracteriza solo propiedades locales de una red, pero con frecuencia esta poca información sobre la estructura de una red es suficiente para determinar sus propiedades básicas [24].

Índice de acoplamiento, μ_{ij}**.** Un índice de acoplamiento (*matching index* en inglés) puede ser asignado a cada arista en una red para cuantificar la semejanza entre la conectividad de dos vértices adyacentes a una arista [19]. Así un valor pequeño en el índice de acoplamiento identifica a una arista que conecta dos regiones diferentes en la red, por lo que juega un papel importante al ser como un acceso directo entre regiones distantes en la red. El índice de acoplamiento de una arista $\{i, j\}$ es obtenido computacionalmente como el número de conexiones que relacionan a los vértices i y j (por ejemplo conexiones a un mismo vértice k), divididos por el número total de conexiones de ambos vértices (excluyendo las conexiones entre los vértices i y j) y se determina como

$$\mu_{ij} = \frac{\sum_{k \neq i,j} a_{ik} a_{jk}}{\sum_{k \neq j} a_{ik} + \sum_{k \neq i} a_{jk}}, \tag{2.21}$$

donde a_{ik} es el número de conexiones de un vértice i hacia un vértice k y a_{jk} es el número de conexiones de un vértice j a un vértice k.

El índice de acoplamiento también puede ser adaptado para considerar a todos los vecinos más cercanos de un vértice [19].

Para la mayoría de las propiedades estructurales no existen formalmente métricas que las cuantifiquen, por tanto, a la hora de elegir una métrica es importante elegir aquella que sea fácil de calcular y que caracterice de la mejor manera la propiedad estructural ya que se puede perder información al no elegir la correcta.

Al trabajar con diferentes métricas es importante *normalizar* todos los valores dentro de un mismo rango para poder compararlos entre ellas. Para normalizar las métricas es necesario conocer los valores mínimos y máximos que éstas pueden tomar. En el Apéndice B.3 se muestran los valores mínimos y máximos que toman las métricas utilizadas en la herramienta de software desarrollada.

2.2.4 Modelos de redes complejas

Durante muchos años las estructuras de redes con conexiones aleatorias [29] eran el principal objeto de estudio de investigadores de diversas ramas de ciencias como las matemáticas, las ciencias de la computación, comunicaciones, biología, sociología y economía entre otras [24]. En 1990 el estudio de la evolución y la estructura de redes se convirtió en un nuevo campo de la física estadística [24]. Desde entonces varios modelos de redes complejas han sido propuestos en base a su distribución del grado y han sido objeto de gran interés, tal como el modelo Mundo pequeño de Watts y Strogatz [56], las redes Libre de escala de Barabási y Albert [8] y las redes Exponenciales de Erdős y Rényi [8].

En esta sección se describen las principales propiedades estructurales, la distribución del grado y se muestra una ilustración del aspecto de la topología de las principales redes complejas. Las ilustraciones mostradas se realizaron con la herramienta Network Workbench.

Redes Aleatorias, RAL. Las redes Aleatorias (*Random Networks* en inglés) son consideradas el modelo más básico de las redes complejas. Son aquellas en las cuales las conexiones de las aristas a los vértices se realizaron uniformemente al azar. Por lo anterior la mayoría de los vértices tienen aproximadamente el mismo grado.

Para valores grandes de n, la distribución del grado sigue una distribución de Poisson, de ahí que la probabilidad de que un vértice tenga un grado k se determina como

$$P(k) \sim \frac{\langle k \rangle^k e^{-k}}{k!}. \tag{2.22}$$

En la Figura 2.4 se muestra una imagen de la topología de una red aleatoria uniforme de 50 vértices.

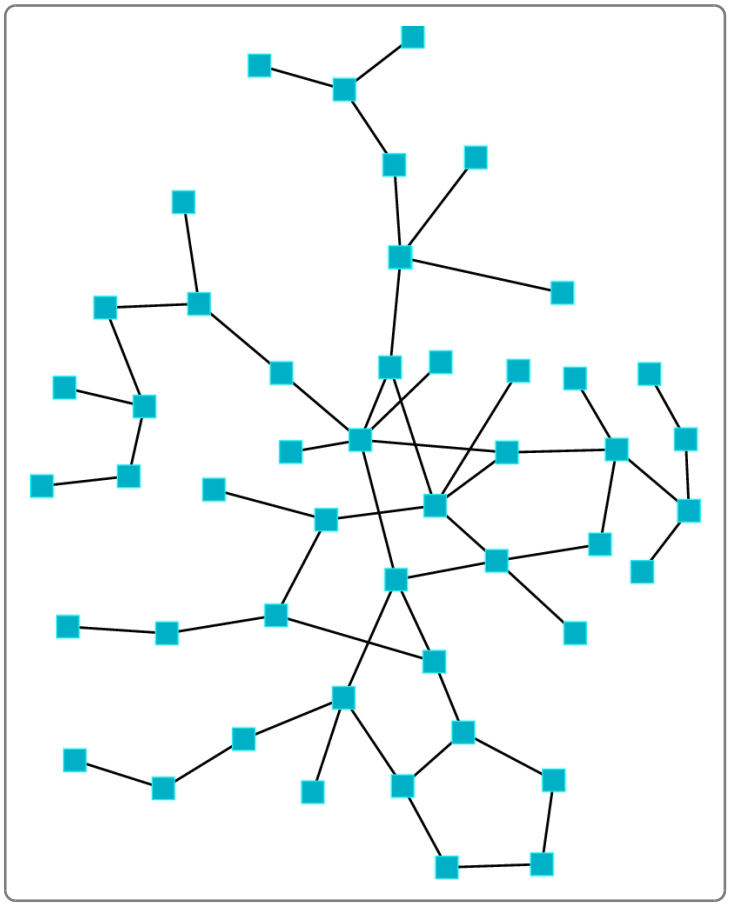


Figura 2.4: Topología de una red Aleatoria uniforme de 50 vértices donde cada uno de los vértices tienen aproximadamente el mismo grado.

Redes Exponenciales, REX. Las redes Exponenciales (*Exponential Networks* en inglés) son redes que exhiben una distribución del grado Exponencial como la red de energía del sur de California y la red de vías de ferrocarril de la India [48]. En estas redes la distribución del grado $P(k)$ muestra un pico en k y después cae exponencialmente para valores grandes de k:

$$P(k) \sim e^{-k}. \tag{2.23}$$

Este tipo de redes no son tolerantes a fallas ya que no importa que vértice se elimine porque el daño es similar [3]. En las redes Exponenciales se observa que la mayoría de los vértices se relacionan aproximadamente con la misma cantidad de vértices y solo unos cuantos se relacionan con una mayor cantidad.

En la Figura 2.5 se muestra la imagen correspondiente a la topología de una red Exponencial.

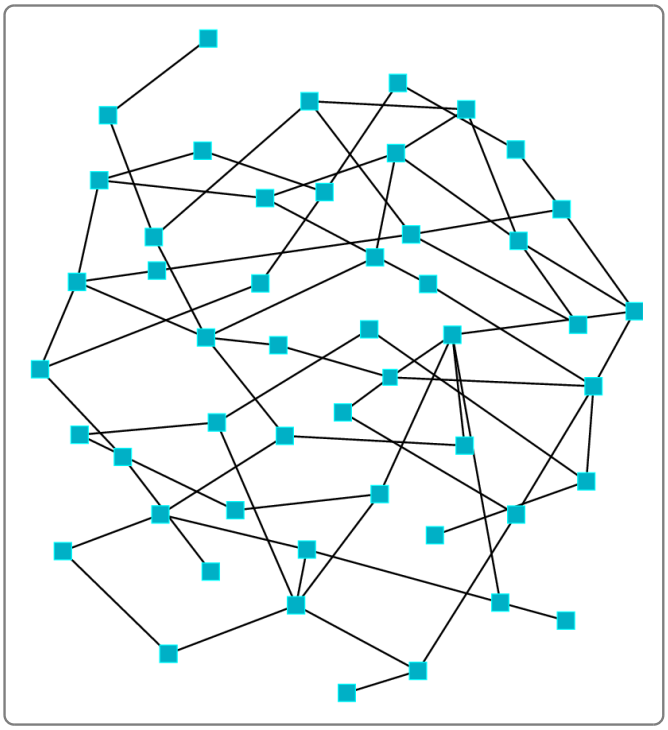


Figura 2.5: Topología de una red Exponencial de 50 vértices donde se observa que algunos vértices se conectan con la misma cantidad de vértices mientras unos cuantos se conectan con una mayor cantidad.

Redes Mundo pequeño, RMP. Las redes Mundo pequeño (*Small world Networks* en inglés) no son precisamente una clase de redes complejas, sino más bien una característica que implica que las redes complejas tengan un alto coeficiente de agrupamiento y una distancia pequeña entre sus vértices [56].

Ejemplos de redes Mundo pequeño son la red eléctrica occidental de los Estados Unidos y las redes sociales [56]. En este tipo de redes las enfermedades se difunden mucho más rápido que en cualquier otro tipo de red debido a la poca distancia entre sus vértices. Un ejemplo es la enfermedad A(H1N1) [40].

En la Figura 2.6 se muestra la topología correspondiente a una red Mundo pequeño.

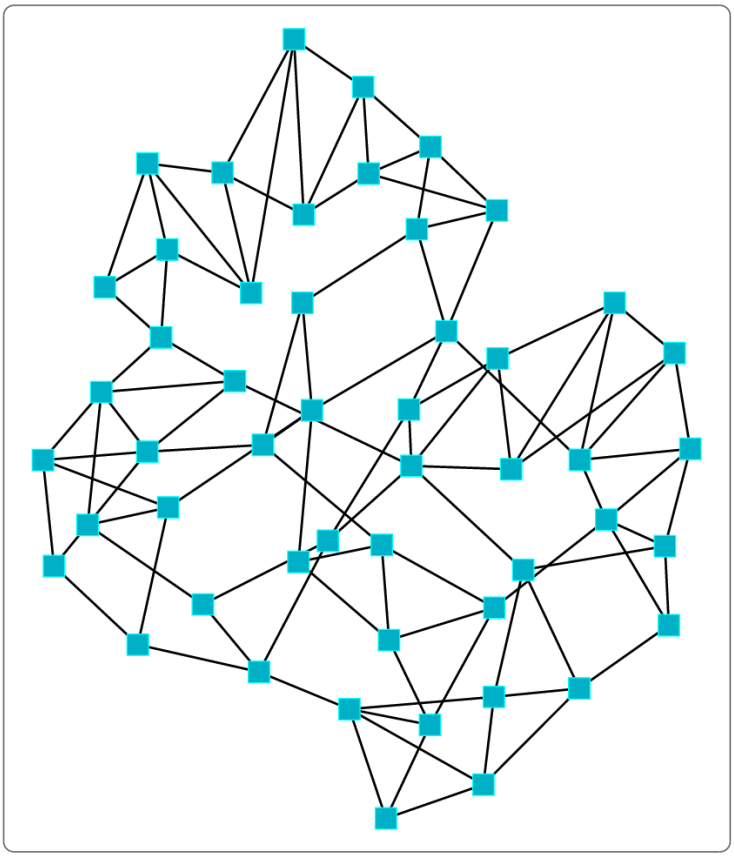


Figura 2.6: Topología de una red Mundo pequeño de 50 vértices donde se observa un alto coeficiente de agrupamiento y una distancia pequeña entre sus elementos. Estas características explican el porqué se aprecia la formación de grupos entre los elementos de la red.

Redes Libre de escala, RLE. Las redes Libre de escala (*Power Law o Scale free Networks* en inglés) son aquellas en las que independientemente del número de vértices, la distribución del grado no cambia. Ejemplos de redes complejas Libre de escala son la World Wide Web, Internet, redes de proteínas, redes de transporte y redes de metabolismo [10]. Estas redes se denominan así porque siguen una distribución del grado Libre de escala que se denota como

$$P(k) \sim k^{\gamma}, \tag{2.24}$$

donde el parámetro de control γ decrece desde ∞ hasta cero y describe que tan rápido decae la frecuencia de aparición del grado k, por tanto el grado promedio de la red se incrementa a medida que γ se decrementa.

Este tipo de redes están formadas por vértices con un grado alto mientras otros tienen pocas conexiones. Esto permite que la tolerancia a fallas aleatorias sea grande, pero si las fallas son en vértices que tienen un grado alto (vértices centrales o hubs) entonces la red se torna vulnerable [3]. En la Figura 2.7 se muestra la topología correspondiente de una red Libre de escala.

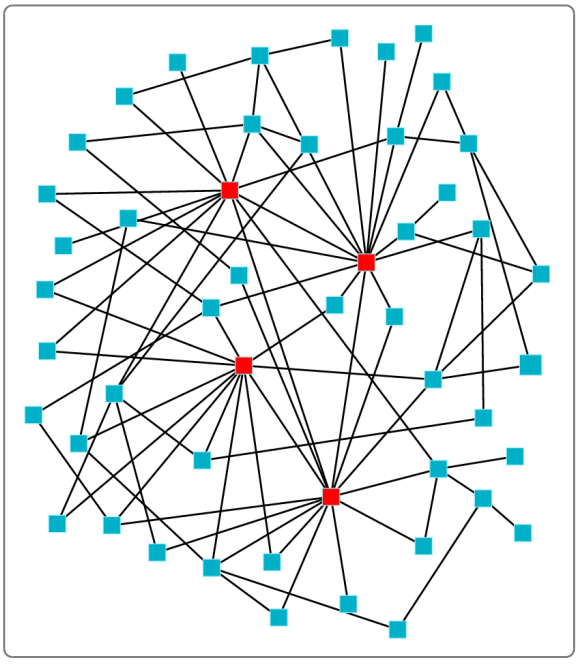


Figura 2.7: Topología de una red Libre de escala de 50 vértices donde algunos vértices tienen un alto grado (vértices en color rojo) mientras otros tienen pocas conexiones.

2.3 Algoritmos evolutivos

Los *algoritmos evolutivos* son métodos de optimización y búsqueda de soluciones basados en los postulados de la evolución biológica. En ellos se mantiene un conjunto de entidades que representan posibles soluciones, las cuales se mezclan y compiten entre sí, de tal manera que las más aptas son capaces de prevalecer a lo largo del tiempo evolucionando hacia mejores soluciones cada vez. Ghosh [28] menciona que dentro de los algoritmos evolutivos se clasifican las *estrategias evolutivas*, la *programación evolucionaria*, la *programación genética* y los *algoritmos genéticos*. Los algoritmos genéticos son los algoritmos mayormente utilizados para intentar resolver problemas de búsqueda y optimización [30] por lo cual su metodología es utilizada para resolver el problema de optimización planteado en este trabajo. A continuación se explica en qué consisten los algoritmos genéticos.

2.3.1 Algoritmos genéticos

Los algoritmos genéticos (AG) están basados en el comportamiento de la selección natural y la genética [30]. Sus principios fueron establecidos por Holland [32] y han sido descritos por Goldberg [30] y Michalewicz [41].

Los algoritmos genéticos trabajan con una *población* de *individuos* ya que éstos resultan ser posibles soluciones a un problema dado. Operan con poblaciones de individuos hasta un cierto número de *iteraciones* y en cada iteración transforman la población en una nueva *generación* utilizando los principios de Darwin basados en la reproducción y supervivencia del más fuerte. Por este mismo principio naturalmente suceden cambios en la información de cada individuo. Tales cambios genéticos consisten en *cruzamientos* y *mutaciones* en la información de cada individuo. Cada individuo de la población es conocido como un *cromosoma* y a la información contenida en cada cromosoma se le denomina *gen*, la mayoría de las veces los genes se representan con el alfabeto binario $\{0, 1\}$ pero no están limitados unicamente a

esa codificación. A cada cromosoma se le asocia un valor denominado *aptitud* que determina sus características y favorece su reproducción en la siguiente generación. La aptitud es determinada por una *función de adaptación* la cual debe ser diseñada para cada problema de manera específica. Una función de adaptación bien puede ser la *función objetivo* de un problema de optimización.

Cuando se realiza un cruzamiento entre dos cromosomas padres se combinan las características de los dos para crear dos hijos similares que bien son soluciones potenciales con mejores características gracias al intercambio de información o intercambio de genes de los padres. Una manera de realizar un cruzamiento entre dos cromosomas es seleccionando el punto de intercambio de información al azar, es decir, se corta en dos segmentos a los cromosomas en una posición escogida al azar para producir dos segmentos iniciales y dos segmentos finales. Después se intercambian los segmentos finales para producir dos nuevos cromosomas. Esta forma de cruzar dos cromosomas se conoce como *operador de cruze basado en un punto*. Un ejemplo de este operador de cruce se muestra en la Figura 2.8.

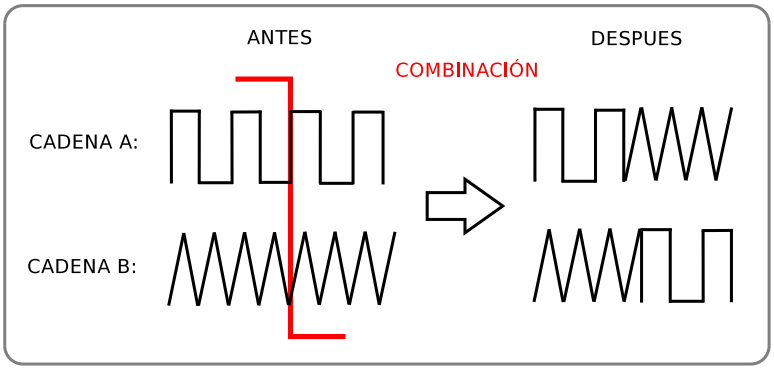


Figura 2.8: Una manera de realizar una combinación o cruzamiento de información entre dos individuos es seleccionando el punto de intercambio al azar para generar dos nuevos individuos (adaptado de [30]).

Cuando se realiza una mutación sobre un cromosoma se le alteran uno o más genes. Una manera de mutar un cromosoma es con aleatoriedad, es decir, cuando se codifica la información en una cadena binaria se tiene un conjunto de genes sobre

los cuales se seleccionan genes al azar y se mutan sus valores actuales por sus valores opuestos, de tal manera que si un gen tenía un valor igual a cero se realiza una mutación cambiando su valor a uno. En la Figura 2.9 se muestra como se realizó a un cromosoma la mutación de cinco de sus genes.

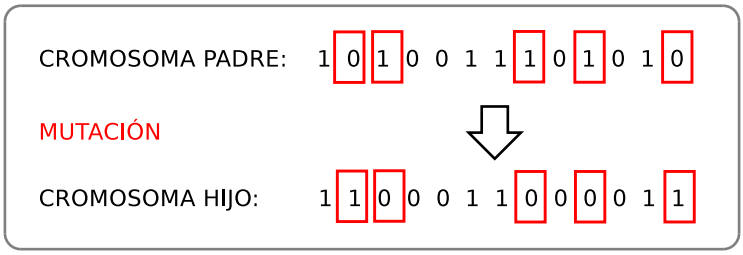


Figura 2.9: Una manera de realizar mutaciones a un cromosoma es cambiando los valores de cinco de sus genes.

Los pasos típicos que realiza un algoritmo genético pueden observarse en la Figura 2.10 y resumirse de la siguiente manera:

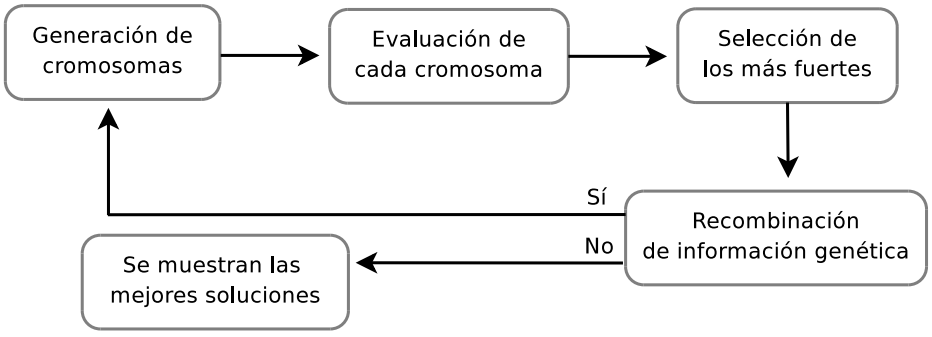


Figura 2.10: Diagrama general del funcionamiento de un algoritmo genético.

1. Se genera una población inicial (P_i) de cierto tamaño k.

2. Se eligen los mejores cromosomas de la P_i para realizar bajo un cierto criterio combinaciones entre los cromosomas y así formar una población cruzada (P_c).

3. A partir de la población cruzada P_c y con alguna probabilidad se eligen elementos y se crea una población mutada (P_m) bajo algún criterio establecido.

4. Se toman los cromosomas de la P_i, la P_c y la P_m y a través de una función de adaptación (aptitud) se evalúa a cada individuo para elegir a los mejores como las soluciones que formaran la siguiente *generación* o población inicial.

5. Después de un número fijo de iteraciones se muestran los q mejores cromosomas obtenidos de la última iteración.

Los algoritmos genéticos son diferentes a los métodos tradicionales de optimización exacta tal como ramificación y acotamiento [58] y a los métodos heurísticos de búsqueda local como búsqueda tabú, búsqueda por entornos variables y otros más [42] principalmente por cuatro motivos:

- Trabajan por lo general con información codificada en algún alfabeto, por ejemplo, en una cadena binaria finita.
- Realizan la búsqueda sobre un conjunto de soluciones y no sobre una sola.
- Utilizan solo la información de la función objetivo.
- Utilizan reglas de transición probabilistas.

Los algoritmos genéticos realizan estas funciones de cruzamiento y mutación a los individuos de una población durante un número fijo de iteraciones. Por lo general entre más iteraciones se realicen se espera obtener mejores resultados porque los cromosomas descendientes van heredando cada vez mejores genes de cada uno de los padres.

2.4 Optimización

La *optimización* se refiere al estudio de problemas en donde se busca encontrar una solución o soluciones factibles que maximicen o minimicen una función objetivo, la cual mide la calidad, costo o desempeño de una solución [35].

Una gran cantidad de problemas teóricos y de la vida real pueden ser representados a través de un *modelo matemático* de optimización. Un modelo matemático es un conjunto de relaciones matemáticas que representan una abstracción de un sistema real o complejo.

Un problema general de optimización se encuentra formado por variables de decisión, una función objetivo y restricciones. Las *variables de decisión* son las variables para las cuales se desea encontrar los *mejores* valores, por ejemplo la cantidad de producto a enviar desde el centro de producción i hasta el centro de consumo j (x_{ij}), la cantidad de insumos a adquirir en el período t (y_t) y el número de horas que se destinará a la máquina i para producir el artículo j en el período t (Z_{ij}).

La *función objetivo* especifica la relación de las variables de decisión y una medida de desempeño del sistema cuyo valor se desea minimizar o maximizar. Una función objetivo es un criterio que específica si una solución es mejor que otra. Algunos ejemplos de funciones objetivos son aquellas que intentan la maximización de utilidades, la minimización de costos o ambas, en donde éstas deben ser escritas en función de las variables del problema.

Las *restricciones* especifican los valores límites que pueden tomar las variables de decisión, ya sean limitantes físicas, económicas y técnicas entre otras. Un problema de optimización puede contener restricciones de igualdad y desigualdad. Las restricciones de igualdad frecuentemente describen la operación de un sistema, mientras que las restricciones de desigualdad definen límites de las variables.

Un modelo general de optimización se puede describir de la siguiente manera:

$$\begin{aligned} & z = \text{mín } f(x) \\ & \text{sujeto a:} \quad g(x) \leq 0,\, h(x) = 0, \\ & x \in X \subseteq \mathbb{R}^n. \end{aligned} \tag{2.25}$$

Cuando las funciones f, g y h son *lineales*, el modelo general es un problema de *programación lineal*. La programación lineal estudia la optimización de una función objetivo lineal sujeta a un conjunto determinado de restricciones lineales de igualdad

o desigualdad donde las variables de decisión pueden tomar valores continuos.

Existen algunos problemas de optimización para los cuales los valores de las variables deben ser estrictamente enteras. A estos problemas se les conoce como *problemas de programación lineal entera* [58]. Existen algunos otros problemas de optimización que necesitan ser modelados con variables que tomen valores discretos o enteros y variables que tomen valores fraccionarios. A éstos problemas se les conoce como *problemas de programación entera mixta*. El modelo matemático que se utiliza en el presente trabajo es un modelo de maximización que pertenece a la *programación entera mixta* [58]. Un problema entero mixto se puede describir de la siguiente manera:

$$\begin{aligned} z &= \text{máx } cx + hy \\ &\text{sujeto a:} \quad Bx + Hy \leq b, \\ &x \geq 0,\ y \geq 0, \\ &x \in \mathbb{R}^n, y \in \mathbb{Z}^p, \end{aligned} \tag{2.26}$$

donde B es una matriz $m \times n$; H es una matriz $m \times p$; h es un vector fila e y es un vector columna de variables enteras.

2.4.1 Optimización multiobjetivo

Cuando se tiene más de un criterio en un problema de optimización, el problema se clasifica como un problema de optimización multiobjetivo (POM) [22]. La optimización multiobjetivo no se restringe a la búsqueda de una única solución, sino de un conjunto de soluciones llamadas *soluciones no dominadas* [43]. Cada solución de este conjunto se dice que es un *óptimo de Pareto* y, al representarlas en el espacio de los valores de las funciones objetivo, conforman lo que se conoce como *frente de Pareto*. Dado un problema concreto, la obtención del frente de Pareto es la principal finalidad de la optimización multiobjetivo.

Muchos problemas de optimización del mundo real son problemas de optimización multiobjetivo, un ejemplo de esto sucede cuando una empresa pretende maximizar la calidad de un producto y minimizar su costo de producción simultáneamente.

Un ejemplo del frente de Pareto para este problema se muestra en la Figura 2.11.

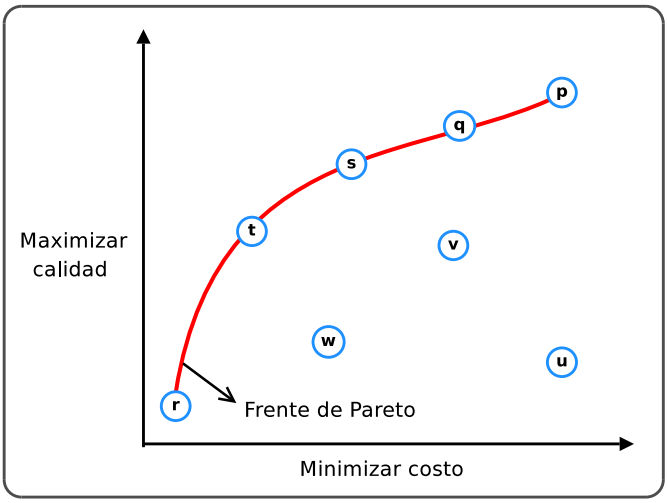


Figura 2.11: Ejemplo del frente de Pareto para un problema multiobjetivo de dos criterios. Las puntos $\{p, q, s, t, r\}$ son soluciones no dominadas y forman el frente de Pareto del problema. Los puntos $\{u, v, w\}$ son soluciones dominadas ya que existen puntos que otorgan una mayor calidad con el mismo costo. En el caso de las soluciones no dominadas el punto p representa la solución con mayor calidad pero también de mayor costo, mientras que el punto r representa la solución de menor calidad y menor costo. Considerando ambos objetivos no se puede decir que la solución p es mejor que la solución r. En estos casos es el tomador de decisiones quien decide cual es la mejor solución.

Existen diversas técnicas para obtener el frente de Pareto y éstas se pueden clasificar en tres categorías: enumerativas, deterministas y estocásticas [15]. En los últimos años, los métodos estocásticos han sido ampliamente estudiados; en particular, un gran número de autores han trabajado en la investigación de algoritmos evolutivos [43]. Sbalzarini [49] y Deb [20] mencionan que al extender los principios de los algoritmos evolutivos en la optimización multiobjetivo, se tienen dos principales problemas:

1. El cómo determinar la aptitud de cada individuo y su selección para guiar la búsqueda a través del conjunto óptimo de Pareto.

2. El cómo mantener una diversidad en los individuos de una población para evitar la prematura convergencia de las soluciones.

Deb [21] menciona que existe una carencia de eficientes algoritmos de optimización multiobjetivo por lo cual los POM deben ser planteados como problemas monobjetivos para poder resolverlos. La dificultad de resolverlos radica en las condiciones de optimalidad para múltiples objetivos.

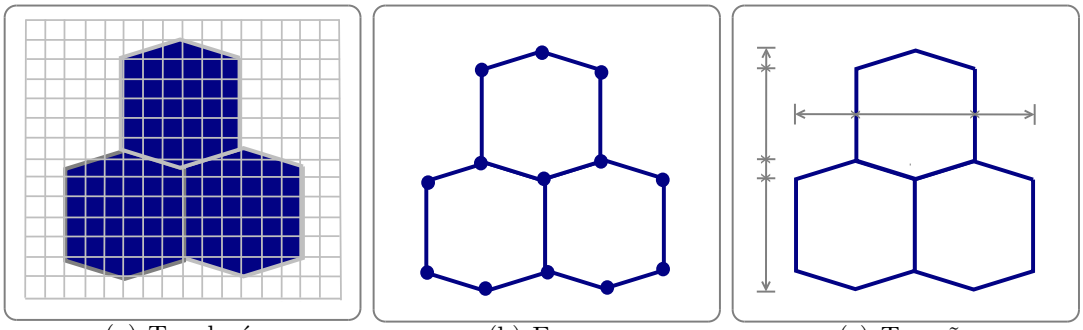
Basados en las afirmaciones de Deb [21] y en la literatura existente, en este trabajo se realizó una simplificación de la optimización multiobjetivo y se modificó el problema para abordarlo como un problema de maximización o bien un problema de optimización de un solo criterio para facilitar el proceso de solución. La función objetivo utilizada consiste en una *suma ponderada* de las propiedades a optimizar, la cual se explica más adelante en la Sección 3.3. Se está consciente que el utilizar esta estrategia limita el espacio de soluciones por lo cual se deja como trabajo futuro el abordar la optimización estructural de redes complejas como un problema multiobjetivo para así no limitar el espacio de búsqueda de soluciones.

2.4.2 Optimización estructural en redes complejas

La *optimización estructural* es clasificada en tres categorías [34, 39]:

- **Optimización de la topología.** La optimización de la topología refiere a la búsqueda de un óptimo diseño para un sistema de ingeniería.
- **Optimización de la forma.** La optimización de la forma refiere a la optimización del contorno de un sistema estructural cuya topología es fija.
- **Optimización del tamaño.** La optimización del tamaño refiere a la búsqueda de una sección transversal óptima o a la dimensión oṕtima de un sistema estructural cuya topología y forma son fijos.

Un ejemplo de las optimizaciones de topología, forma y tamaño se muestra en la Figura 2.12.



(a) Topología (b) Forma (c) Tamaño

Figura 2.12: Tipos de optimización estructural (inspirado de [34]).

En este trabajo cuando se habla de la optimización estructural se hace referencia a la optimización de la topología, es decir, al óptimo diseño de las conexiones entre los elementos. La optimización estructural ha sido exhaustivamente estudiada en sistemas compuestos por un solo componente, sin embargo; en la vida real la mayoría de las estructuras de ingeniería están compuestas por más de uno. Las carrocerías de automóviles, équipos de cómputo, máquinas y estructuras de los aviones son algunos ejemplos prácticos de las diversas aplicaciones de la ingeniería [37]. En estas estructuras los ingenieros de diseño podrían estar interesados en conocer que diseño de patrones lograrían de la manera más rápida el mejor desempeño de la estructura dentro de las limitaciones de la fabricación. Por esta razón, la optimización de las conexiones es de gran importancia ya que es capaz de proveer la solución más confiable para el diseño de múltiples sistemas.

La mayoría de los problemas de ingeniería pueden ser modelados y resueltos a través de redes complejas. La optimización estructural de redes complejas es un tema ya abordado, pero esta optimización está basada en cubrir los requerimientos de la red al inicio de su creación, es decir, cuando apenas se realiza el diseño de la estructura topológica. Xu [59] menciona que los principales métodos para la construcción de redes son la técnicas de grafos lineales, el método Caley y la técnica del producto cartesiano. Estas técnicas al construir las redes no dan importancia a

su distribución del grado, por lo cual no son útiles para crear redes complejas. Se podría aprovechar al conocimiento de las propiedades estructurales correspondientes a las redes complejas para crear redes con esas propiedades a través de generadores de modelos, pero existe un inconveniente al crear redes de esta manera, ya que no es posible maximizar o minimizar el valor de alguna otra propiedad porque estas redes ya tienen propiedades estructurales bien definidas. Para poder cambiarlas se necesitaría hacer uso de alguna herramienta de software diseñada para ello.

En la literatura existen trabajos que se enfocan principlamente al análisis y visualización de redes sociales a través de herramientas de software, un ejemplo es la herramienta *Network Workbench* (NWB) [7]. NWB funciona bajo las plataformas Linux y Windows y a través de ella se realiza un análisis de las conexiones de la red, se realizan perturbaciones a la red al eliminar vértices y aristas aleatoriamente y se ofrece la generación y visualización de modelos de redes Aleatorias, Libre de escala y Mundo pequeño entre otras. Una captura de pantalla de esta herramienta se muestra en la Figura 2.13.

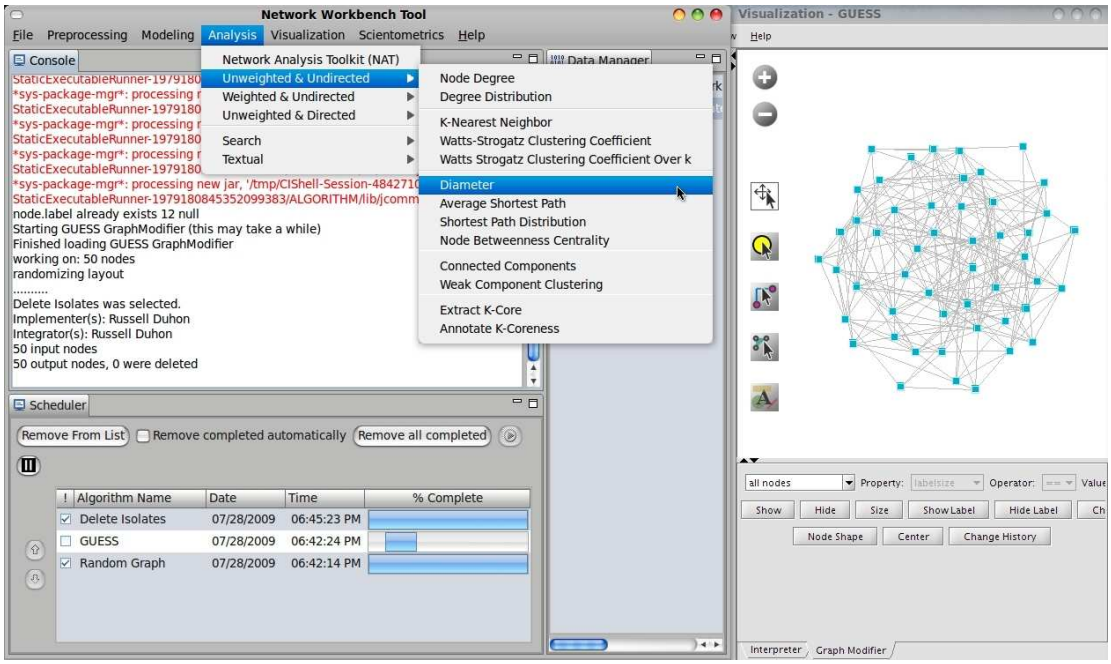


Figura 2.13: Captura de pantalla de Network Workbench.

Pajek [9] es otra herramienta que ofrece un análisis y visualización de redes complejas. Pajek fue desarrollada principalmente para un ambiente Windows pero puede ser emulada por algún otro sistema operativo. A través de Pajek se pueden realizar cortes al grafo, calcular propiedades básicas de una red, eliminar vértices y aristas aleatoriamente entre otras. Una captura de pantalla de esta herramienta se muestra en la Figura 2.14.

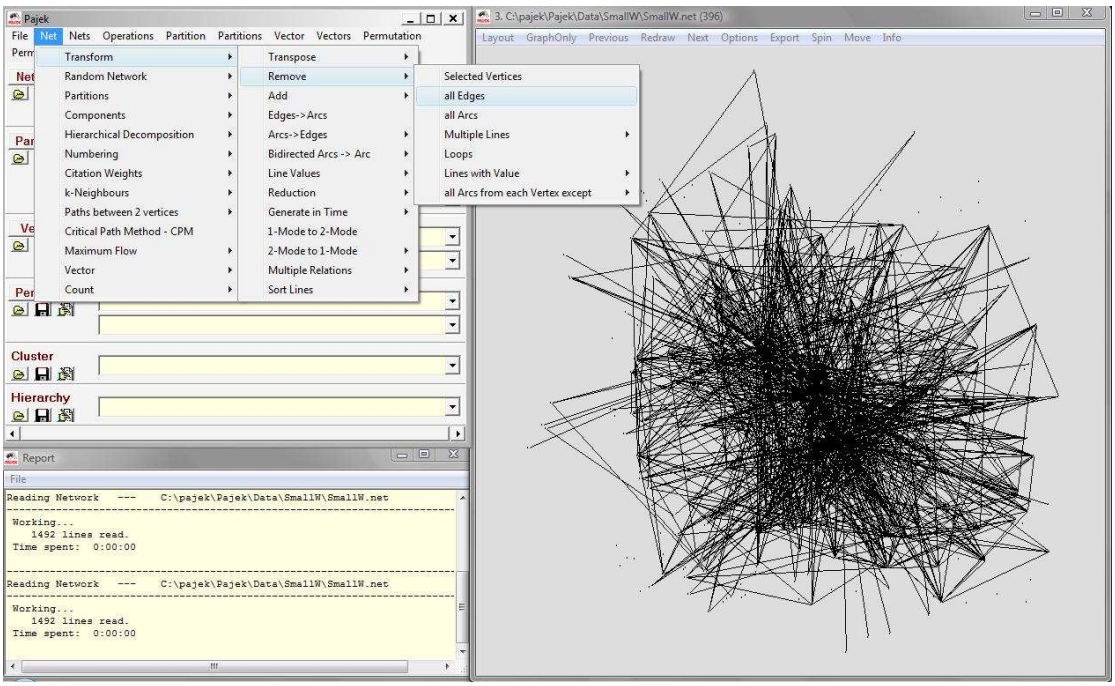


Figura 2.14: Captura de pantalla de Pajek.

Network Workbench y Pajek no son las únicas herramientas existentes, a través de la *International Network for Social Network Analysis* (INSNA) [57] se puede obtener acceso a un conjunto de herramientas de software que facilitan el entendimiento de las redes complejas al analizarlas y visualizarlas. Algunas de estas herramientas consideradas por Huisman y van Duijn [33] como las más relevantes debido a las opciones que ofrecen al usuario, son mostradas en la Tabla 2.1 con su versión, el tipo de análisis que realizan sobre las redes y la indicación sobre si es de distribución libre o comercial.

Además de estas herramientas de software, existen librerías dedicadas al análisis de redes sociales tal como *igraph* [18]. Ésta fue desarrollada en lenguaje C, es de distribución libre y contiene implementaciones para resolver problemas típicos de la teoría de grafos y flujo en redes. Otra librería de distribución libre es *SNA package* [12], la cual es una colección de rutinas para R project [27] que realiza análisis estadísticos y visualizaciones de redes sociales.

Tabla 2.1: Herramientas de software que analizan redes sociales [33]. El análisis que realizan puede ser *descriptivo* (*d*) el cual estudia las conexiones de los elementos, tal como el grado, el diámetro y la ruta más corta; *estructural* (*e*) el cual estudia las propiedades estructurales, tal como la densidad y el grado de intermediación; *posicional y de roles* (*pr*) que estudia las distancias entre los elementos y los roles que éstos toman y *estadístico* (*s*) donde se obtienen correlaciones entre los elementos, regresiones lineales entre otros.

Software	Versión	Análisis	Visualización	Disponibilidad
GRADAP	2.0	d, e	No	Comercial
Igraph	0.5.2	d, e	Sí	Libre
KrackPlot	3.3	rp	Sí	Libre
MultiNet	4.24	d, pr, s	Sí	Libre
NetDraw	1.0	d, e	Sí	Libre
NetMiner II	2.3.0	d, e, pr, s	Sí	Comercial
Network Workbench	1.0	d, e, pr	Sí	Libre
Pajek	1.25	d, e, pr	Sí	Libre
SocNetV	0.70	d, e	Sí	Libre
StOCNET	1.4	d, s	No	Libre
STRUCTURE	4.2	e, pr	No	Libre
UCINET	6.05	d, e, pr, s	Sí	Comercial
visone	1.0b1	d, e	Sí	Comercial

Si el lector está interesado en profundizar sobre estas librerías y herramientas de software para el análisis y visualización de redes sociales, se recomienda consultar el texto de Huisman y van Duijn [33] y la INSNA [57].

Al analizar las herramientas de software y las librerías disponibles en la literatura, se observó que ninguna contempla la optimización de la topología de la red. Estos antecedentes motivaron la creación de una herramienta de software que al mismo tiempo cumpliera con varias funciones, tales como:

1. El análisis de una red compleja.

2. La optimización de las propiedades estructurales de redes complejas de diferentes tamaños sin limitar la optimización de la red debido a su aplicación, es decir, que funcione tanto en redes tecnológicas, de comunicaciones, sociales y biológicas entre otras.

3. La selección de los mejores individuos para presentarlos como el conjunto de soluciones eficientes, es decir, que otorgue la libertad al usuario de elegir que solución es la mejor tomando como criterio los valores de las propiedades optimizadas y el costo involucrado al realizar los cambios sugeridos.

4. La optimización de una red existente ya que no es lo mismo optimizar la topología de una red en funcionamiento a optimizar la topología de una red que está en fase de construcción; por lo cual, la herramienta de software que se propone a partir de una red existente y en base a ciertas restricciones, modifica la topología de la red original para generar otra que cumpla con los requerimientos deseados.

La herramienta de software creada en este trabajo [14] cumple con los cuatro puntos anteriores además de ser de código abierto, libre distribución y funcional en ambientes Linux, Windows y Unix.

Capítulo 3

Planteamiento del problema de optimización

En el presente capítulo se describen las características que debe cumplir cada una de las posibles soluciones, se presentan las propiedades que la herramienta de software optimiza, las métricas elegidas para cuantificar las propiedades a optimizar y se presenta la descripción formal del problema de optimización.

En la Sección 1.1 se describió la necesidad de construir una herramienta de software que optimice las propiedades estructurales en redes complejas en base a las necesidades o requerimientos del usuario. Para que la herramienta de software optimice la topología de una red se necesita especificar la información de la misma. Esta información corresponde a:

- El número de elementos y conexiones que conforman la red para representarla como un conjunto de vértices y aristas, es decir, como un grafo. La herramienta de software a partir del número de vértices de la red determina la topología del grafo completo. Es necesario determinar el grafo completo porque así la herramienta conoce que aristas son factibles agregar, modificar o eliminar para diseñar el conjunto de nuevas topologías con mayor calidad estructural.
- El listado de las conexiones con sus respectivas distancias físicas o costos por utilizarlas.

Para que la herramienta de software diseñe un conjunto de redes alternativas existe un criterio que debe cumplirse. Este criterio es considerado como la restricción que define el conjunto de soluciones factibles y consiste en:

R1 Que el grafo sea conexo.

Para satisfacer la **R1** la herramienta de software después de realizar alguna función de mutación o cruzamiento verifica que exista una ruta que incluya todos los vértices de la red, ya que con esto se asegura que la red es conexa. En caso de que la red sea desconexa, ésta se elimina de la población para que no genere un esfuerzo computacional innecesario al calcular los valores de sus propiedades. Lo anterior está basado en que al optimizar una red, el usuario final probablemente no desea una red en donde algunos de los elementos queden aislados sin comunicación con los demás elementos. En este trabajo se espera que la red original sea conexa ya que en todo momento se validan que las redes creadas también lo sean. En caso que la red no sea conexa, la herramienta de software creará conexiones hacia el vértice que esté aislado y convertirá la red en una que sea conexa.

3.1 Optimización de las propiedades estructurales

En un grafo cuando se habla de la estructura topológica se hace referencia a las conexiones entre los componentes (vértices y aristas). El diseño de la estructura topológica es importante para construir un grafo que satisfaga los requerimientos necesarios, es decir, que cuente con la calidad que el usuario necesita. Xu [59] menciona algunos principios fundamentales que deberían tomarse en cuenta al diseñar la topología de un grafo.

1. **Grado pequeño y fijo.** El tener un grado alto en la red implica mayores costos por lo cual sería preferible una red con un grado pequeño. Tener una red con un grado pequeño genera menores costos a la hora de realizar mantenimientos en el grafo pero otorga una menor robustez.

2. **Retardo corto en la transmisión.** El retardo en la transmión de un mensaje de un vértice a otro es proporcional al tiempo que el mensaje tarda en ser almacenado y reenviado por los vértices intermedios.

3. **Robustez o tolerancia a fallas.** La red debería continuar trabajando en caso de que vértices o aristas llegaran a fallar.

4. **Fácil ruteo.** El ruteo es una importante función de comunicación entre redes. Esto implica que debe tenerse una ruta fija a través de la cual se transporten mensajes de un vértice a otro.

5. **Simetría.** Implica que todos los componentes de la red se comporten de la misma manera y se comuniquen de formas similares.

6. **Eficiencia**. La comunicación entre los elementos de una red representa el mayor costo y las limitaciones del funcionamiento, por lo cual es de suma importancia determinar que tan eficiente es el intercambio de información.

7. **Baja vulnerabilidad.** En una red es importante conocer los vértices que son los más importantes y los más vulnerables ya que estos vértices influyen directamente en la eficiencia de la red.

8. **Escalabilidad.** Se refiere a que sea factible construir una red de mayor tamaño que la actual, por lo cual algunas de las propiedades deseables deberían permanecer y algunos parámetros calcularse fácilmente.

Las propiedades estructurales a optimizar varían según la aplicación de la red, por lo cual con los principios fundamentales mencionados por Xu [59] se puede concluir que en una red de transporte o servicio se desearía minimizar: el grado, el retardo en la transmisión de mensajes, el ruteo y la vulnerabilidad; y en la misma red se desearía maximizar la robustez, la simetría, la eficiencia y la escalabilidad.

Hay que tener en cuenta que algunos de estos conceptos se hallan en conflicto, por ejemplo, un grafo completo al tener todas las posibles aristas entre sus elementos involucra el mayor costo. Un grafo en forma de árbol tiene un fácil ruteo, es escalable y eficiente pero no es robusto. Otro ejemplo es que no se puede tener una red con un grado pequeño y una gran tolerancia a fallas. Es entonces que elegir una combinación de propiedades deseables es una decisión que trae consigo compromisos y desventajas entre varias propiedades a la hora de optimizar la topología de la red.

Para optimizar la calidad de una red se decidió utilizar en esta investigación los anteriores principios fundamentales como las propiedades que debería tener un red para un mejor funcionamiento. Ahora lo relevante es el seleccionar las métricas que cuantifican tales propiedades. Las métricas seleccionadas para la herramienta de software se abordan en la Sección 3.2.

3.2 Relación de métricas a utilizar según las propiedades a optimizar

Teniendo en cuenta los principios fundamentales o propiedades deseables en una red mencionados por Xu [59] se decidieron utilizar las siguientes métricas para cuantificarlas.

- **Retardo corto en la transmisión: Diámetro.** El Diámetro es una medida que cuantifica la cantidad de aristas que atraviesa un mensaje en la red.
- **Robustez o tolerancia a fallas: Grado mínimo y Grado máximo.** La robustez se puede cuantificar a través de métricas *probabilistas* y *deterministas*. Las métricas probabilistas indican mejor la tolerancia a fallas que las deterministas, sin embargo; son computacionalmente más difíciles de calcular [6] por lo cual se recomienda utilizar métricas deterministas tal como la conectividad de las aristas. La robustez también puede ser representada como un simple modelo de percolación, pero desafortunadamente esta modelación es limitada a grafos que tienen una distribución Poisson en el grado de sus vértices [13]. Como la entrada de la herramienta de software es un grafo cuya distribución no está limitada a ser Poisson, el cálculo de la percolación no resulta útil para determinar la robustez del grafo. Entonces para medir la robustez de una red se cuantificará la conectividad de las aristas a través de la métrica grado mínimo. Al eliminar un vértice con el menor grado se elimina el menor número de aristas que convierten a la red en desconexa y provocan su falla. Por tanto entre mayor sea la conectividad (o en nuestro caso se tenga un grado promedio grande) mayor es la robustez de una red.
- **Fácil ruteo: Longitud de la ruta más corta.** Para definir una ruta fija a través de la cual se transporten mensajes por la red no hay nada mejor que calcular la ruta más corta de menor costo. Si alguno de los elementos de la ruta más corta llegase a fallar sería bueno utilizar la segunda ruta más

corta y así sucesivamente. Como no se tiene la certeza que estas rutas siempre estén disponibles (debido a mantenimiento o fallas en algunos componentes) se decidió calcular para todos los pares de vértices del grafo las rutas más cortas y promediarlas para así conocer el costo y tiempo que tardaría un ruteo.

- **Simetría: Grado promedio.** La métrica Grado promedio es una medida de simetría en los vértices de la red, por lo cual si los vértices tienen la misma cantidad de enlaces con otros elementos todos se comunican de una forma similar.

- **Eficiencia: Eficiencia global.** La métrica Eficiencia global de la red determina la eficiencia en el intercambio de información entre los componentes de la red.

- **Baja vulnerabilidad: Vulnerabilidad.** La métrica Vulnerabilidad es la propia cuantificación de la propiedad vulnerabilidad, por lo cual no hay que buscar alguna otra métrica para determinarla.

- **Escalabilidad.** La herramienta de software cuantifica el crecimiento de una red al contar el número de aristas que se agregaron y eliminaron de la topología original para desarrollar una red alternativa. Independientemente del número de aristas que constituyen a las redes alternativas el algoritmo genético asegura que las propiedades deseadas permanezcan en cada iteración.

Otra métrica que fue programada en la herramienta de software es:

- **Coeficiente de agrupamiento.** El cual determina la densidad de la red y además ayuda a optimizar las redes Mundo pequeño.

Todas las métricas mencionadas anteriormente fueron elegidas como las adecuadas porque cuantifican las propiedades mencionadas y son calculadas en un tiempo pequeño.

Muchas de las métricas que dan información útil en la red tienen la desventaja de ser costosas computacionalmente al requerir un gran tiempo de cálculo y aún peor si se trata de optimizarlas. Un ejemplo de ésto es el cálculo del grado de intermediación [19] que proporciona información sobre cuales vértices y aristas son los más importantes de la red. La desventaja de calcularla radica en tener que enumerar las veces que lo atraviesan las rutas más cortas entre cualquier par de vértices $\{i, j\}$. Esto requiere al menos un $O(nm)$ tiempo para grafos no ponderados y un tiempo $O(nm + n^2 \log n)$ para grafos ponderados [5]. Al repetir su cálculo en la herramienta de software para cada uno de los individuos que se generan durante todas las iteraciones del algoritmo genético hace que aumente de forma considerable el tiempo de cómputo y por tanto de optimización.

Una forma de realizar la optimización se obtiene al encontrar todas las diferentes estructuras topológicas que puede tener una red de n vértices. Para cada una de ellas se evaluaría la propiedad estructural de interés a través de una métrica. De todo el conjunto de redes se elegiría como la óptima aquella que tenga el mayor o el menor valor de la propiedad de interés. El resolver un problema de optimización de esta manera permite encontrar la mejor de todas las soluciones, es decir, la solución exacta. El principal inconveniente al resolver un problema de manera exacta radica en el tiempo computacional que toma ya que se deben evalular cada una de las posibles soluciones. El número de redes alternativas que se obtienen para una red de n vértices se pueden determinar a través de la *Teoría de enumeración de Polya* [17]. El número de estructuras diferentes para redes de hasta 16 vértices se muestra en la Tabla 3.1.

Tabla 3.1: Número de estructuras diferentes que existen para redes de hasta 16 vértices. [17]

n	Número de redes	Número de redes conexas
1	1	1
2	2	1
3	4	2
4	11	6
5	34	21
6	156	112
7	1044	853
8	12346	11117
9	274668	261080
10	12005168	11716571
11	1018997864	1006700565
12	165091172592	164059830476
13	50502031367952	50335907869219
14	29054155657235488	29003487462848061
15	31424685969804308768	31397381142761241960
16	6400101570452755789492	6396956011322517617627

Si el lector está interesado en conocer más sobre la resolución de este tipo de problemas y su complejidad computacional se le recomienda consultar el texto de Papadimitriou [47].

3.3 Función objetivo

Las soluciones obtenidas a través de la herramienta de software se deben discriminar de acuerdo a algún criterio para determinar que soluciones son mejores que otras. Es de suma importancia tener definida una clara y eficaz función objetivo a través de la cual se seleccionen los mejores resultados.

La función objetivo utilizada en este trabajo es una suma ponderada que evalúa

la calidad o *aptitud* de cada solución y tiene la siguiente forma

$$f(G) = p_1 \cdot \mathcal{E} + p_2 \cdot D + p_3 \cdot C + p_4 \cdot \delta + p_5 \cdot \Delta + p_6 \cdot \langle k \rangle + p_7 \cdot L + p_8 \cdot \Upsilon + p_9 \cdot \text{costo}, \quad (3.1)$$

donde $p_1, \ldots, p_8$ representan la importancia de las propiedades elegidas (*ponderaciones*) y $\mathcal{E}, \ldots, \Upsilon$ representan los valores de cada una de las propiedades cuantificadas a través de las métricas correspondientes. Por cada propiedad que se optimice el usuario debe asignarle una ponderación que identifique las propiedades que tienen mayor importancia o peso a la hora de realizar la optimización. Si la ponderación es una cantidad positiva indica que se desea maximizar y si la ponderación corresponde a una cantidad negativa se entiende que se desea minimizar dicha propiedad. En caso de no estar interesado en la optimización de una propiedad su ponderación debe ser igual a cero. Al final de la función objetivo se encuentra una variable que almacena el costo para cada solución que evalúa la $f(G)$. En este trabajo cuando se habla del costo de una red se hace referencia a la cantidad de movimientos que realizó el AG, tales movimientos consisten en agregar y eliminar conexiones en la topología del grafo. El costo es un parámetro que el usuario puede minimizar, maximizar o no darle importancia a través de su respectiva ponderación.

Como el usuario determina que propiedades desea optimizar y cuales no tienen importancia, la función objetivo calculará solo las propiedades que se le indiquen. Por ejemplo, si solo se desea maximizar la eficiencia, minimizar el diámetro y maximizar el grado promedio con el menor costo se obtendría la siguiente función objetivo

$$f(G) = (1 \cdot \mathcal{E}) \; + \; (-1 \cdot D) \; + \; (1 \cdot \langle k \rangle) \; + \; (-1 \cdot \text{costo}), \quad (3.2)$$

donde 1 y -1 son ponderaciones ilustrativas ya que el usuario elige las cantidades discretas o continuas que deseé utilizar. Esta misma función podría denotarse como un vector de ponderaciones de la siguiente manera

$$f(G) = (1, -1, 0, 0, 0, 1, 0, 0, -1), \quad (3.3)$$

donde 1 y -1 indican las propiedades a maximizar y minimizar respectivamente y el vector de ponderaciones sigue un orden de propiedades a optimizar que se determinan

por las métricas eficiencia, diámetro, coeficiente de agrupamiento, grado mínimo, grado máximo, grado promedio, longitud de la ruta más corta y vulnerabilidad. La relevancia del costo de la red se determina con el último dígito del vector de ponderaciones.

Los individuos que se eligen como los mejores son aquellos que tienen los valores más altos en su función de calidad o aptitud, porque son los que aseguran una mejora en los valores de las propiedades deseadas.

3.4 Definición del problema

Entrada. Un grafo $G_o = (V, E)$ conexo con los correspondientes costos o distancias físicas $(\ell_{i,j})\ \forall\ i, j \in E$ al cual se le pretende optimizar sus propiedades estructurales, el vector de ponderaciones que indican la minimización o maximización de las propiedades estructurales en la función objetivo $f(G)$ y el número de redes alternativas por construir (q).

Salida. Un conjunto de grafos conexos alternativos de tamaño q con mayor calidad estructural que el grafo G_o.

Capítulo 4

Herramienta de software

Para resolver el problema de encontrar un conjunto de redes con mayor calidad que la red original, la herramienta de software hace uso de los conocimientos existentes de la optimización monobjetivo y los algoritmos genéticos. Se propone la optimización porque se desea maximizar o minimizar un conjunto de propiedades en la topología del grafo. Se propone un algoritmo genético como método de solución porque su naturaleza es realizar mutaciones y cruzamientos a la información y esa naturaleza es la que se necesita aplicar a la topología de un grafo para obtener redes con una estructura diferente y con mayor calidad. Además con los algoritmos genéticos se obtiene un conjunto de soluciones lo cual otorga libertad al tomador de decisiones para elegir la más idónea de acuerdo a sus necesidades.

En este capítulo se presentan los componentes de la herramienta de software y se explica su funcionamiento en conjunto con el algoritmo genético programado.

4.1 Funcionamiento de la herramienta de software

En la Figura 4.1 se muestra de forma general el funcionamiento de la herramienta de software programada y los procesos involucrados se describen posteriormente.

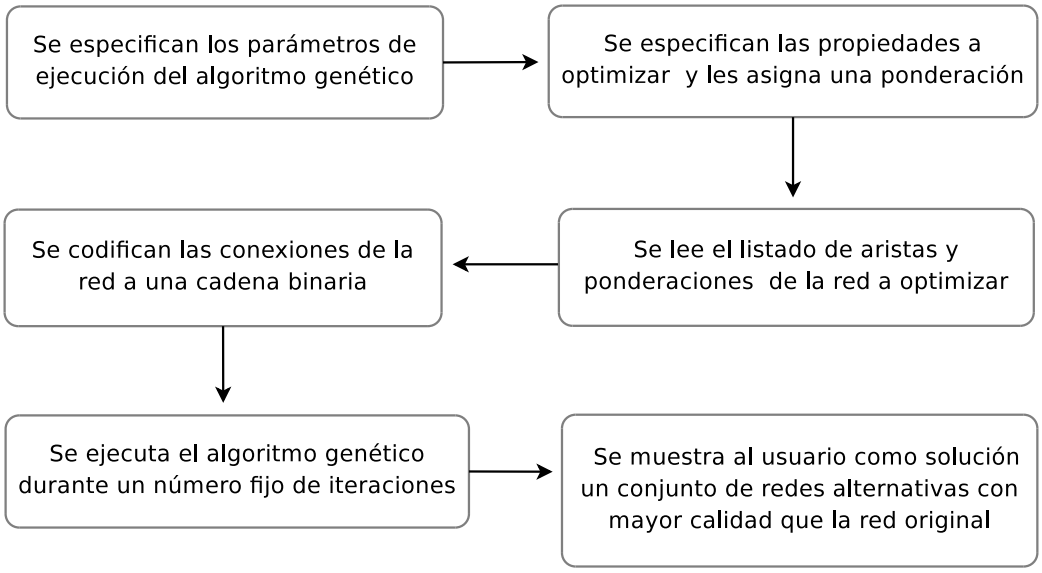


Figura 4.1: Diagrama general del funcionamiento de la herramienta de software.

- **Se especifican los parámetros para la ejecución del algoritmo genético.** Al iniciar la herramienta ésta ya cuenta con los parámetros que se consideran los recomendados por generar el conjunto de las mejores soluciones en el menor tiempo. El usuario tiene la libertad de cambiar los parámetros al iniciar la ejecución. Los parámetros que deben especificarse deben tener un valor mayor a cero y a excepción de las probabilidades de mutación y cruze que deben sumar uno, los otros parámetros no están restringidos a algún valor. Los parámetros que se manejan en la herramienta de software son:

  - Tamaño de la población inicial.
  - Cantidad de mutaciones por iteración.
  - Cantidad de iteraciones.
  - Probabilidad de mutación y cruze.

  La cantidad de mutaciones por iteración se refiere a las mutaciones que se realizan a cada uno de los cromosomas que conforman la P_m durante una iteración del algoritmo genético. Por ejemplo, si a un cromosoma x se le realizan cinco mutaciones por iteración durante cuatro iteraciones, entonces al finalizar

el algoritmo genético se habrán realizado 20 mutaciones al cromosoma x. Lo anterior siempre que el cromosoma x resulte tener una de las mejores aptitudes de entre todos los individuos generados durante las cuatro iteraciones.

Para este trabajo de tesis se utiliza una simple estrategia de selección de la población para su cruzamiento y mutación, la cual consiste en seleccionar los individuos al azar. La probabilidad de que un individuo elegido para ser cruzado con otro individuo es de 0.5 y la probabilidad de que un indiviuo sea elegido para realizarle alguna mutación es también de 0.5.

- **Se indican las propiedades a optimizar.** A las propiedades que se desean optimizar según la aplicación del grafo, el usuario les debe asignar una ponderación que bien puede ser discreta o continua. La ponderación que se asigne identifica las propiedades que tienen mayor importancia o peso a la hora de realizar la optimización, es decir, entre mayor sea la ponderación de una propiedad con respecto a las otras esa propiedad será la de mayor prioridad. Una vez asignadas las ponderaciones el usuario debe establecer la ponderación como una cantidad positiva si desea maximizarla, o bien; establecer la ponderación como una cantidad negativa en caso de minimizarla. Por ejemplo, si a la vulnerabilidad se le asigna la ponderación -1 se indica que se desea minimizar. Si el usuario no desea optimizar alguna propiedad basta con que otorgue a la ponderación correspondiente un valor igual a cero. Las propiedades que se pueden optimizar en la herramienta de software se enlistan en la Sección 3.2.

- **Se lee el archivo con información de la red.** Para que la herramienta de software funcione correctamente se debe contar con un archivo que indique en el primer renglón el número de vértices (n) y aristas del grafo (m). Posteriormente se deben listar las aristas o conexiones del grafo con sus respectivas ponderaciones. Si el inicio y el final del archivo comienza con el símbolo '#' se considera a tales líneas como comentarios. Un ejemplo del formato del archivo de entrada para una red de cuatro vértices y cuatro aristas se muestra en la Tabla 4.1.

Tabla 4.1: Formato del archivo de entrada para una red de cuatro vértices con cuatro aristas ponderadas.

```
#Comentario: Inicio de la declaración de aristas
n 4 m 4
1 2 4.3
1 3 3
1 4 5.8
3 4 2.77
#Fin de la declaración de aristas
```

Una vez leída la información del grafo original se construye su matriz de adyacencia (abordada en la Sección 2.1) en la cual basta utilizar la mitad de la información para conocer las conexiones entre los vértices.

- **Se codifica la información del grafo en una cadena binaria.** Después de construir la matriz de adyacencia se representan las conexiones del grafo a través de una *cadena binaria* que representa la información del grafo original también conocido como *cromosoma padre*. Esta codificación se realiza para poder introducir la información del grafo al algoritmo genético tal como se muestra en la Figura 4.2.

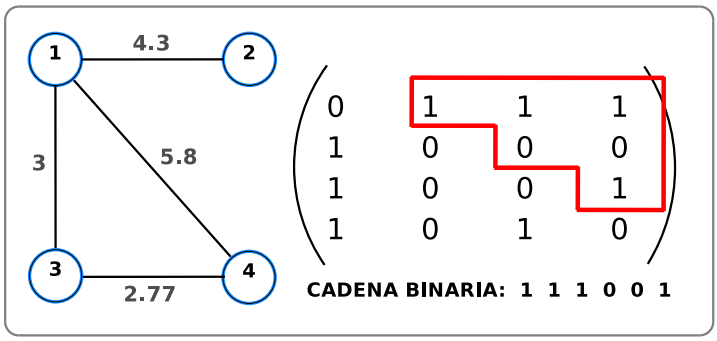


Figura 4.2: Un grafo no dirigido y ponderado de cuatro vértices y cuatro aristas con su respectiva matriz de adyacencia. Las conexiones del grafo se representan a través de una cadena binaria.

- **Se ejecuta el algoritmo genético.** Durante un número fijo de iteraciones (w) se ejecuta el algoritmo genético y se realizan los siguientes pasos:

  1. Se crea una *población inicial* (P_i). El algoritmo genético al inicializar, es decir, durante la primera iteración toma como base la información del cromosoma padre y crea un duplicado de cromosomas a los cuales se le realizan cambios (si un gen vale 0 se cambia a 1 y si vale 1 se cambia a 0), con el objetivo de crear una P_i la cual es la base para comenzar con las iteraciones del genético.

  2. Se crea una *población cruzada* (P_c). El algoritmo genético comienza a realizar mutaciones entre los individuos de la P_i, es decir, toma aleatoriamente información de dos cromosomas de la P_i para generar una tercera población a través de un cruzamiento o intercambio de información, con lo cual se obtienen q nuevos individuos que conforman la P_c.

  3. Se crea una *población mutada* (P_m). El algoritmo genético comienza a realizar mutaciones entre la P_i y la P_c, es decir, se eligen uniformemente al azar h individuos de la P_i y de la P_c para realizarles mutaciones y crear una nueva P_m. Esto se obtiene cambiando los valores de los cromosomas, es decir, si un valor elegido al azar de la cadena binaria o cromosoma era igual a 1, entonces se logra su mutación cambiando su valor a 0, y lo mismo para el caso contrario, si un valor de la cadena binaria era 0 entonces se cambia su valor a 1.

  4. Se evalua la calidad o aptitud. A cada cromosoma de las tres poblaciones creadas: P_i, P_c y la P_m se le evalúa su calidad o aptitud a través de una función objetivo. La función objetivo utilizada se describe en la sección 3.3.

  5. Se conforma la P_i de la siguiente iteración. Se seleccionan los mejores individuos de la P_i, P_c y la P_m a través de la función objetivo para formar la siguiente P_i porque se pretende que en cada iteración del genético, los mejores cromosomas pasen sus propiedades a la siguiente generación.

- **Se muestra al usuario el conjunto de redes alternativas de mayor calidad.** Cuando finaliza la ejecución de la herramienta de software se toman los mejores individuos generados durante todas las w iteraciones del algoritmo genético y se muestran al tomador de decisiones para que eliga la solución que considere más atractiva. La herramienta de software genera como salida dos archivos:

  - Un archivo que muestra las cadenas binarias de todos los individuos seleccionados como los mejores durante la ejecución del algoritmo genético. Un ejemplo de este archivo se muestra en el Apéndice B.4.
  - Otro archivo que muestra un resumen de los parámetros establecidos en la herramienta de software, los valores de las propiedades estructurales calculadas, el costo por diseñar la topología y el porcentaje de mejora obtenido. Un ejemplo del formato del archivo se muestra en el Apéndice B.5.

Capítulo 5

Experimentación computacional

Tal como se describió en la Sección 1.3 el principal objetivo de la herramienta de software es realizar cambios en la topología de una red para optimizar su calidad. Para mejorar la calidad, la herramienta de software optimiza ciertas propiedades estructurales que son definidas por el usuario.

Para satisfacer el objetivo de este trabajo, se decidió que la experimentación consistiría en optimizar redes complejas Aleatorias, Libre de escala y Mundo pequeño. En los experimentos no se pretende convertir un modelo de red compleja en otro modelo, sino más bien se desea mejorar la calidad de la red al agregarle otras propiedades estructurales. Para mejorar la calidad de cada red compleja se pretende que la herramienta de software optimice las nuevas propiedades deseadas y así se mejore su calidad.

Durante el transcurso de la optimización de las redes complejas se espera que a través de la experimentación computacional se determinen los mejores parámetros de ejecución del algoritmo genético. Los parámetros que se pretenden encontrar son:

- La cantidad de aristas mutadas por iteración, es decir, se pretende encontrar la menor cantidad de mutaciones que deben realizarse sobre cada individuo ya que se espera que al realizar el menor número de cambios se involucre el menor costo.
- La cantidad de iteraciones, es decir, determinar la cantidad de veces que debe ejecutarse el algoritmo genético para que encuentre el conjunto de mejores so-

luciones. El realizar un gran número de iteraciones involucra un mayor tiempo de ejecución mientras que un número pequeño de iteraciones limita el espacio de búsqueda de soluciones, por tanto se pretende encontrar un número de iteraciones que genere buenas soluciones en un tiempo de cómputo razonable.

- El tamaño de la población inicial, es decir, la cantidad de redes que deben diseñarse por cada iteración. Dentro de la herramienta el tamaño de la población inicial indica el tamaño del conjunto de mejores soluciones que serán mostradas al finalizar la ejecución del algoritmo genético.

5.1 Condiciones de experimentación

Las ponderaciones utilizadas en los experimentos tienen un valor igual a 1 o -1 para indicar la maximización o minimización de una propiedad estructural. La probabilidad de que un individuo sea elegido para realizarle mutaciones es de 0.5 y la probabilidad de que un individuo sea elegido como un padre para realizar un cruzamiento es también de 0.5. Durante la experimentación de este capítulo se mantienen fijas ambas probabilidades por lo cual se deja como trabajo futuro el determinar cómo el modificar las probabilidades de cruzamiento y mutación influyen en la creación de mejores soluciones.

Los experimentos se realizaron sobre redes de 50 vértices debido al tiempo de ejecución que tardaba la herramienta en dar una solución, ya que para redes mayores a 100 vértices se requiere de un mayor esfuerzo computacional. El calcular el tiempo computacional que toma realizar la optimización de las propiedades en redes complejas de hasta 1000 vértices se aborda en la Sección 5.8.

Todos los experimentos implementados en el presente trabajo se realizaron en un servidor Dell con procesador Intel Core 2 Quad a 2.4 Ghz. con 3.2 Gb en memoria RAM bajo la distribución Ubuntu 9.04 [38] de GNU/Linux [51] con el compilador GCC de GNU Compiler Collection [52].

5.2 Instancias

Para la realización de los experimentos se utilizaron como instancias redes complejas Aleatorias, Libre de escala y Mundo pequeño debido a que representan la mayoría de los sistemas complejos de nuestra sociedad [4]. Gracias a los *generadores de modelos* de Virtanen [55] se construyeron, para cada una de las redes Aleatorias, Libre de escala y Mundo pequeño, cinco redes de 50 vértices donde las redes Aleatorias cuentan con un promedio de 222 aristas, las redes Libre de escala con un promedio de 231 aristas y las redes Mundo pequeño con un promedio de 186 aristas. En total se crearon 15 redes complejas que sirven como base para la creación de las instancias a resolver durante la experimentación de este capítulo. Los modelos y los parámetros utilizados para generar las redes mencionadas se describen en el Apéndice A. Se deja como trabajo futuro el optimizar estructuralmente redes complejas de una mayor dimensionalidad. En este trabajo no fueron abordadas debido al tiempo computacional que toma resolverlas. Los tiempos de ejecución en redes de mayor tamaño se reportan en la Sección 5.8.

5.3 Experimentación: redes a optimizar

Para demostrar el funcionamiento de la herramienta de software a partir de las 15 redes complejas construidas se diseñaron seis clases de instancias. Los experimentos consisten en añadirle a cada modelo de red compleja las propiedades estructurales características de los otros dos modelos de redes complejas. Así se obtiene dos problemas de optimización por resolver por cada tipo de red compleja que se traduce en un total de seis clases de instancias.

En la Tabla 5.1 se muestra la simbología utilizada durante la experimentación y posteriormente se explica en que consisten las seis clases de instancias con las cuales se realizan todos los experimentos de este capítulo.

Tabla 5.1: Simbología utilizada durante la optimización de las seis clases de instancias.

Símbolo	Significado
$\mathcal{E}(G)$	Eficiencia del grafo
$D(G)$	Diámetro del grafo
$C(G)$	Coeficiente de agrupamiento del grafo
$\delta(G)$	Grado mínimo del grafo
$\Delta(G)$	Grado máximo del grafo
$\langle k \rangle(G)$	Grado promedio del grafo
$L(G)$	Longitud de la ruta más corta del grafo
$\Upsilon(G)$	Vulnerabilidad del grafo
$f(G)$	Función objetivo o aptitud del grafo
$\|\text{cb}\|$	Largo de la cadena binaria
costo	Número de aristas agregadas y eliminadas en la estructura del grafo
mut/iter	Mutaciones por iteración
mej/costo	Relación del porcentaje de mejora y el costo
AG	Algoritmo genético
RAL	Redes Aleatorias
RLE	Redes Libre de escala
RMP	Redes Mundo pequeño

- **Clase 1: optimización RAL/RLE.** Consiste en optimizar RAL con respecto a las propiedades de una RLE, es decir:
  - Maximizar el $\Delta(G)$.
  - Minimizar el $C(G)$ y el $\delta(G)$.

  Con lo cual se trabaja con la siguiente función objetivo:

$$f(G) = (1 \cdot \Delta(G)) + (-1 \cdot C(G)) + (-1 \cdot \delta(G)) + (-1 \cdot \text{costo}). \quad (5.1)$$

- **Clase 2: optimización RAL/RMP.** Consiste en optimizar RAL con respecto a las propiedades de una RMP, es decir:

  - Maximizar el $C(G)$.
  - Minimizar la $L(G)$.

  Con lo cual se trabaja con la siguiente función objetivo:

$$f(G) = (1 \cdot C(G)) + (-1 \cdot L(G)) + (-1 \cdot \text{costo}). \tag{5.2}$$

- **Clase 3: optimización RLE/RAL.** Consiste en optimizar RLE con respecto a las propiedades de una RAL, es decir:

  - Maximizar el $\langle k \rangle(G)$.
  - Minimizar el $\Delta(G)$.

  Con lo cual se trabaja con la siguiente función objetivo:

$$f(G) = (1 \cdot \langle k \rangle(G)) + (-1 \cdot \Delta(G)) + (-1 \cdot \text{costo}). \tag{5.3}$$

- **Clase 4: optimización RLE/RMP.** Consiste en optimizar RLE con respecto a las propiedades de una RMP, es decir:

  - Maximizar el $C(G)$.
  - Minimizar la $L(G)$ y el $\Delta(G)$.

  Con lo cual se trabaja con la siguiente función objetivo:

$$f(G) = (1 \cdot C(G)) + (-1 \cdot L(G)) + (-1 \cdot \Delta(G)) + (-1 \cdot \text{costo}). \tag{5.4}$$

- **Clase 5: optimización RMP/RAL.** Consiste en optimizar RMP con respecto a las propiedades de una RAL, es decir:

  - Maximizar el $\langle k \rangle(G)$.
  - Minimizar el $C(G)$.

Con lo cual se trabaja con la siguiente función objetivo:

$$f(G) = (1 \cdot \langle k \rangle(G)) + (-1 \cdot C(G)) + (-1 \cdot \text{costo}). \tag{5.5}$$

- **Clase 6: optimización RMP/RLE.** Consiste en optimizar RMP con respecto a las propiedades de una RLE, es decir:
  - Maximizar el $\Delta(G)$.
  - Minimizar el $\delta(G)$.

  Con lo cual se trabaja con la siguiente función objetivo:

$$f(G) = (1 \cdot \Delta(G)) + (-1 \cdot \delta(G)) + (-1 \cdot \text{costo}). \tag{5.6}$$

Por cada una de las seis clases de instancias se trabaja con un conjunto de cinco instancias, por lo cual los resultados mostrados en este capítulo representan el promedio de las cinco instancias por cada clase durante una repetición del algoritmo genético.

En la Figura 5.1 se aprecian los valores promedios de las cinco redes pertenecientes a los modelos de redes Aleatoria, Libre de escala y Mundo pequeño que se utilizaron para la creación de las instancias de los experimentos de este capítulo. El tener todas las redes con una similitud en su orden y tamaño se refleja en los valores de sus métricas ya que éstas son parecidas.

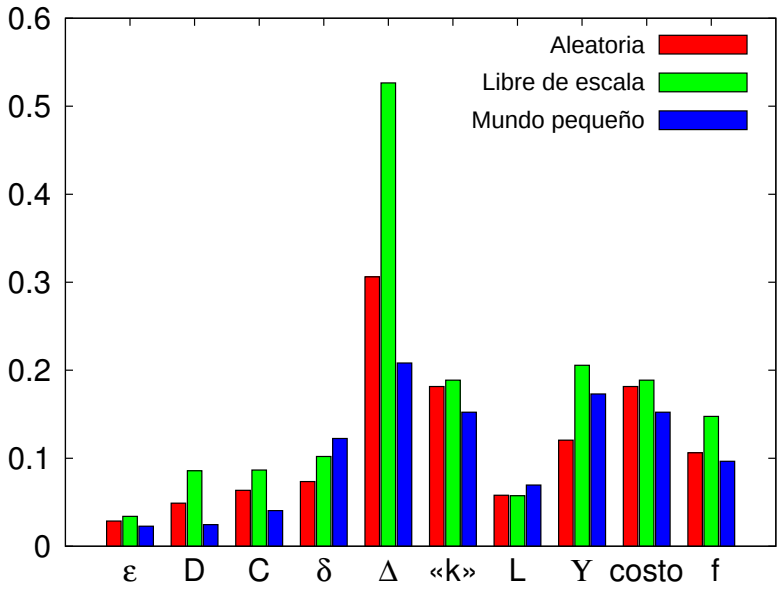


Figura 5.1: Gráfica de los valores promedios de las propiedades de cinco redes Aleatorias, cinco redes Libre de escala y cinco redes Mundo pequeño de 50 vértices.

Al observar los valores de las propiedades estructurales para las tres redes complejas mostradas en la Figura 5.1 se puede concluir que:

- Las redes Aleatorias al tener un similar $\langle k \rangle(\mathrm{G})$ son menos vulnerables que las redes Libres de escala y de Mundo pequeño.
- Las redes Libres de escala cuentan con el mayor $\Delta(\mathrm{G})$ de las tres redes complejas, ya que en su topología algunos vértices tienen un grado alto de conexiones.
- Las redes Mundo pequeño al tener un alto $C(\mathrm{G})$ y una baja $L(\mathrm{G})$ tienen un mayor $\delta(\mathrm{G})$ el cual influye en un menor valor en el $D(\mathrm{G})$ de las tres redes complejas.

5.4 Experimentación: cantidad de aristas mutadas

A través de este experimento se desea determinar la menor cantidad de aristas que deben mutarse sobre los individuos de la P_m. Se decidió no dejar al azar la cantidad de mutaciones que deben realizarse sobre los individuos de la P_m, ya que entre mayores mutaciones se realizen aumenta el costo y el número de modificaciones que deben realizarse sobre la topología.

Para encontrar la mejor solución posible se experimentó con las seis clases de instancias al realizarles diferentes cantidades de mutaciones. Para elegir la cantidad de aristas a mutar se tomaron porcentajes del orden de las redes y la longitud de las cadenas binarias (|cb|) de las instancias propuestas. Una opción era tomar el tamaño de las instancias como parámetro para obtener un porcentaje de aristas mutadas, pero se optó por el tamaño de la cadena binaria porque la herramienta de software trabaja la mayor parte del tiempo con ella por representar todas las posibles conexiones de la red. El tamaño de la cadena binaria para un grafo de 50 vértices es de 1225 aristas y se obtiene al calcular el tamaño del grafo completo no dirigido $n(n-1)/2$.

El número de mutaciones realizadas en base al orden de las redes son del 2 %, 10 %, 20 %, 30 %, 50 % y 100 % que corresponden a realizar 1, 5, 10, 15, 25 y 50 mutaciones. Las mutaciones realizadas en base al tamaño de la cadena binaria son del 13 %, 20 %, 33 %, 60 % y 80 % que equivalen a 153, 245, 408, 735 y 980 mutaciones. Se eligieron estos once porcentajes por considerarse como un rango de mutaciones que pueden otorgar un amplio espacio de búsqueda para encontrar las mejores soluciones. Cabe aclarar que estas cantidades de mutaciones se realizan por iteración (*mut/iter*), es decir, por cada iteración del AG se vuelven a realizar la misma cantidad de mutaciones a cada uno de los individuos que conforman la P_m, de tal manera que si se realizan 20 mutaciones por iteración durante cinco iteraciones, ésto

significa que el AG al finalizar realizó un total de 100 mutaciones por cada individuo de la P_m.

Las seis clases de instancias propuestas en la Sección 5.3 se resolvieron con el AG bajo una P_i igual a 30, durante seis iteraciones y bajo once diferentes cantidades de mutaciones. Los resultados obtenidos para cada una de las seis clases de instancias propuestas se muestran a través de tablas donde el primer renglón indica el porcentaje de n y |cb| sobre el cual se tomaron porcentajes para obtener la cantidad de aristas a mutar, tales cantidades de aristas se muestran en el segundo renglón de la tabla. En la columna a la izquierda se muestran las métricas que se calcularon y con una flecha hacia arriba ($\Uparrow$) se indica si el valor de la métrica se maximizó y con una flecha hacia abajo ($\Downarrow$) se indica si el valor se minimizó. En el renglón donde se muestra el costo de la topología creada, el costo está basado en la cantidad de movimientos (cantidad de aristas agregadas y eliminadas) que se realizó el AG. En el penúltimo renglón de la tabla se observa la aptitud ($f(G)$) obtenida bajo una cierta cantidad de aristas mutadas, siendo la aptitud la indicadora de la calidad promedio de las redes creadas por el AG. En el último renglón se indica la relación del porcentaje de mejora y el costo al realizar un solo movimiento (*mej/costo*) en base a la aptitud y el costo de la red creada. Es importante elegir los valores más altos de la mej/costo sobre la aptitud porque refleja la mayor calidad estructural que se obtiene con el menor costo.

A continuación se muestran los resultados y las conclusiones obtenidas para cada una de las seis clases de instancias propuestas.

5.4.1 Optimización de redes Aleatorias

En la Tabla 5.2 se muestran los resultados obtenidos al realizar la optimización RAL/RLE, es decir, optimizar las cinco redes Aleatorias con respecto a propiedades de una red Libre de escala.

La mejor solución proporcionada por la aptitud durante seis iteraciones del AG se obtiene al realizar 1 mut/iter la cual también produce una mayor calidad por movimiento (mej/costo), es decir, al realizar 1 mut/iter se obtiene una mejora promedio del 388 % con un costo promedio de 8 y una mejora por movimiento del 48.5 %, por tanto, al realizar 1 mut/iter produce la mayor calidad estructural del conjunto de redes alternativas.

Cuando el AG realizó 408, 735 y 980 mut/iter, se crearon redes alternativas que resultaron tener una aptitud menor que la aptitud de la red original, ésto indica que el algoritmo genético durante seis iteraciones no pudo encontrar ninguna red con mejor topología. En estos casos la solución recomendada es naturalmente conservar la topología de la red original. La razón por la cual éstas redes obtuvieron una peor calidad se debió a que los valores de las métricas $C(G)$ y $\delta(G)$ no pudieron ser mejorados por el AG. En este experimento el realizar 980 mut/iter generó las redes de mayor costo (322 movimientos) y con la peor calidad estructural por movimiento (-3.76 %).

Tabla 5.2: Resultados obtenidos al realizar la optimización de la instancia RAL/RLE bajo once diferentes mutaciones por iteración. Los resultados se presentan como porcentajes promedios de mejora y los costos como el número de modificaciones que se realizaron sobre la topología de la red original.

Obj.	mut/iter	2 %n	10 %n	20 %n	30 %n	50 %n	100 %n	13 %|cb|	20 %|cb|	33 %|cb|	60 %|cb|	80 %|cb|
		1	5	10	15	25	50	153	245	408	735	980
⇓	$C(G)$	105 %	121 %	135 %	152 %	172 %	167 %	177 %	206 %	-274 %	-344 %	-400 %
⇓	$\delta(G)$	99 %	83 %	87 %	89 %	102 %	106 %	128 %	151 %	-184 %	-214 %	-229 %
⇑	$\Delta(G)$	115 %	124 %	134 %	143 %	149 %	139 %	132 %	145 %	154 %	166 %	172 %
⇓	costo(G)	8	36	58	86	111	103	108	146	218	273	290
⇑	$f(G)$	388 %	275 %	306 %	317 %	281 %	197 %	24 %	13 %	-233 %	-407 %	-1090 %
⇑	mej/costo	48.5 %	7.63 %	5.27 %	3.68 %	2.53 %	1.91 %	0.22 %	0.08 %	-1.06 %	-1.49 %	-3.76 %

En la Tabla 5.3 se muestran los resultados obtenidos al realizar la optimización RAL/RMP, es decir, optimizar las cinco redes Aleatorias con respecto a propiedades de una red Mundo pequeño.

Al observar la información proporcionada por la aptitud durante seis iteraciones del AG la mayor calidad en una red se obtiene al realizar 408, 735 y 980 mut/iter, pero al realizar 1 mut/iter se produjo la red con mayor calidad estructural por movimiento realizado, es decir, al realizar 408, 735 y 980 mut/iter se obtiene una mejora promedio del 89 %, 88 % y 90 % con un costo promedio igual a 614, 614 y 613 respectivamente, mientras que al realizar 1 mut/iter se produjo una mejora promedio del 1 % con un costo promedio igual a 8. Al analizar la relación de la mejora en la calidad estructural por movimiento realizado se encuentra que es mejor realizar 1 mut/iter ya que produce una mejora del 0.25 %.

Cuando el AG realizó 5 y 10 mut/iter se crearon redes alternativas con una aptitud menor que la aptitud de la red original debido a que la métrica $L(G)$ no pudo ser mejorada por el AG. Por tanto, bajo estos parámetros y durante 6 iteraciones el AG no pudo encontrar ninguna red con mejor topología que la red original. En este experimento las redes de mayor costo (613 y 614 movimientos) se obtuvieron al realizar 408, 735 y 980 mut/iter con una mej/costo del 0.14 % y 0.15 %.

Tabla 5.3: Resultados obtenidos al realizar la optimización de la instancia RAL/RMP bajo once diferentes mutaciones por iteración. Los resultados se presentan como porcentajes promedios de mejora y los costos como el número de modificaciones que se realizaron sobre la topología de la red original.

Obj.	mut/iter	2 %n	10 %n	20 %n	30 %n	50 %n	100 %n	13 %|cb|	20 %|cb|	33 %|cb|	60 %|cb|	80 %|cb|
		1	5	10	15	25	50	153	245	408	735	980
⇑	$C(G)$	0.40 %	1.2 %	7 %	85 %	176 %	352 %	641 %	707 %	722 %	721 %	727 %
⇓	$L(G)$	3 %	-1.6 %	-5 %	18 %	34 %	46 %	56 %	58 %	58 %	58 %	58 %
⇓	costo(G)	4	5	12	115	211	363	565	599	614	614	613
⇑	$f(G)$	1 %	-0.6 %	-2.3 %	0.76 %	8 %	27 %	73 %	86 %	89 %	88 %	90 %
⇑	mej/costo	0.25 %	-0.12 %	-0.19 %	0.007 %	0.04 %	0.07 %	0.13 %	0.14 %	0.14 %	0.14 %	0.15 %

5.4.2 Optimización de redes Libre de escala

En la Tabla 5.4 se muestran los resultados obtenidos al realizar la optimización RLE/RAL, es decir, optimizar las cinco redes Libre de escala con respecto a propiedades de una red Aleatoria.

La información proporcionada por la aptitud durante seis iteraciones del AG indica que al resolver esta instancia se obtuvo la mayor calidad estructural al realizar 50 mut/iter, pero al realizar 1 mut/iter produjo la red con mayor calidad estructural por movimiento realizado, es decir, al realizar 50 mut/iter se obtiene una mejora promedio del 25 % con un costo promedio igual a 252, mientras que al realizar 1 mut/iter se obtiene una mejora promedio del 5 % con un costo promedio igual a 9. Al analizar la relación de la mejora en la calidad estructural por movimiento realizado se encuentra que es mejor realizar 1 mut/iter ya que produce una mejora del 0.56 %.

Cuando el AG realizó 735 y 980 mut/iter se crearon redes alternativas con una aptitud menor que la aptitud de la red original debido a que la métrica $\Delta(G)$ no pudo ser mejorada por el AG. Por tanto, bajo estos parámetros y durante 6 iteraciones el AG no pudo encontrar ninguna red con mejor topología que la red original. En este experimento las redes de mayor costo (414 y 444 movimientos) se obtuvieron precisamente al realizar 735 y 980 mut/iter con una mej/costo del -0.0001 % y -0.004 % respectivamente.

Tabla 5.4: Resultados obtenidos al realizar la optimización de la instancia RLE/RAL bajo once diferentes mutaciones por iteración. Los resultados se presentan como porcentajes promedios de mejora y los costos como el número de modificaciones que se realizaron sobre la topología de la red original.

Obj.	mut/iter	2 %n	10 %n	20 %n	30 %n	50 %n	100 %n	13 %|cb|	20 %|cb|	33 %|cb|	60 %|cb|	80 %|cb|
		1	5	10	15	25	50	153	245	408	735	980
$\Downarrow$	$\Delta(G)$	96 %	92 %	90 %	89 %	87 %	87 %	94 %	96 %	100 %	-103 %	-106 %
$\Uparrow$	$\langle k \rangle(G)$	102 %	108 %	116 %	124 %	138 %	162 %	183 %	191 %	197 %	209 %	217 %
$\Downarrow$	costo(G)	9	37	68	104	157	252	323	351	372	414	444
$\Uparrow$	$f(G)$	5 %	14 %	16 %	17 %	21 %	25 %	19 %	17 %	14 %	-0.39 %	-1.94 %
$\Uparrow$	mej/costo	0.56 %	0.38 %	0.24 %	0.16 %	0.13 %	0.1 %	0.06 %	0.05 %	0.04 %	-0.0001 %	-0.004 %

En la Tabla 5.5 se muestran los resultados obtenidos al realizar la optimización RLE/RMP, es decir, optimizar las cinco redes Libre de escala con respecto a propiedades de una red Mundo pequeño.

Al observar la información proporcionada por la aptitud durante seis iteraciones del AG la mayor calidad en una red se obtiene al realizar 50 mut/iter, pero al realizar 1 mut/iter produjo la red con mayor calidad estructural por movimiento realizado, es decir, al realizar 50 mut/iter se obtiene una mejora promedio del 19 % con un costo promedio igual a 268, mientras que al realizar 1 mut/iter se obtiene una mejora promedio del 3 % con un costo promedio igual a 7. Al analizar la relación de la mejora en la calidad estructural por movimiento realizado se encuentra que es mejor realizar 1 mut/iter ya que produce una mejora del 0.43 %. En este experimento las redes alternativas con menor calidad y mayor costo se obtuvieron al realizar 153, 245, 408, 735 y 980 mut/iter con una mejora por costo de entre el 0.03 % y 0.04 %.

Tabla 5.5: Resultados obtenidos al realizar la optimización de la instancia RLE/RMP bajo once diferentes mutaciones por iteración. Los resultados se presentan como porcentajes promedios de mejora y los costos como el número de modificaciones que se realizaron sobre la topología de la red original.

Obj.	mut/iter	2 %n	10 %n	20 %n	30 %n	50 %n	100 %n	13 %|cb|	20 %|cb|	33 %|cb|	60 %|cb|	80 %|cb|
		1	5	10	15	25	50	153	245	408	735	980
$\Uparrow$	$C(G)$	100 %	108 %	121 %	140 %	171 %	228 %	418 %	447 %	525 %	530 %	542 %
$\Downarrow$	$\Delta(G)$	95 %	93 %	89 %	89 %	89 %	88 %	104 %	106 %	114 %	114 %	115 %
$\Downarrow$	$L(G)$	105 %	128 %	112 %	89 %	73 %	63 %	50 %	48 %	45 %	44 %	44 %
$\Downarrow$	costo(G)	7	31	75	111	169	268	344	379	444	473	481
$\Uparrow$	$f(G)$	3 %	6 %	9 %	11 %	14 %	19 %	9 %	8 %	7 %	6 %	7 %
$\Uparrow$	mej/costo	0.43 %	0.19 %	0.12 %	0.10 %	0.08 %	0.07 %	0.03 %	0.02 %	0.02 %	0.01 %	0.01 %

5.4.3 Optimización de redes Mundo pequeño

En la Tabla 5.6 se muestran los resultados obtenidos al realizar la optimización RMP/RAL, es decir, optimizar las cinco redes Mundo pequeño con respecto a propiedades de una red Aleatoria.

La mejor solución proporcionada por la aptitud durante seis iteraciones del AG se obtiene al realizar 1 mut/iter la cual también produce una mayor calidad por movimiento (*mej/costo*), es decir, al realizar 1 mut/iter se obtiene una mejora promedio del 5 % con un costo promedio igual a 5 y una mejora por movimiento del 1 %, por tanto, al realizar 1 mut/iter produce la mayor calidad estructural del conjunto de redes alternativas.

Cuando el AG realizó 5, 10, 15, 25, 50, 153, 245, 408, 735 y 980 mut/iter las redes alternativas creadas resultaron tener una aptitud menor que la aptitud de la red original, por lo cual no hubo un porcentaje de mejora. La razón por la cual éstas redes obtuvieron una peor calidad se debió a que el valor de la métrica $C(G)$ no pudo ser mejorado por el AG. En este experimento las redes alternativas de mayor costo (324 movimientos) se obtuvieron al ralizar 980 mut/iter.

Tabla 5.6: Resultados obtenidos al realizar la optimización de la instancia RMP/RAL bajo once diferentes mutaciones por iteración. Los resultados se presentan como porcentajes promedios de mejora y los costos como el número de modificaciones que se realizaron sobre la topología de la red original.

Obj.	mut/iter	2 %n	10 %n	20 %n	30 %n	50 %n	100 %n	13 %|cb|	20 %|cb|	33 %|cb|	60 %|cb|	80 %|cb|
		1	5	10	15	25	50	153	245	408	735	980
$\Downarrow$	$C(G)$	2 %	-0.7 %	-5 %	-7 %	-12 %	-27 %	-88 %	-135 %	-250 %	-379 %	-484 %
$\Uparrow$	$\langle k \rangle(G)$	1 %	0.1 %	2 %	3 %	5 %	11 %	32 %	45 %	74 %	98 %	118 %
$\Downarrow$	costo(G)	5	7	8	10	17	34	92	127	207	273	324
$\Uparrow$	$f(G)$	5 %	-0.7 %	-5 %	-7 %	-12 %	-27 %	-88 %	-135 %	-250 %	-379 %	-484 %
$\Uparrow$	mej/costo	1 %	-0.1 %	-0.63 %	-0.7 %	-0.71 %	-0.79 %	-0.96 %	-1.06 %	-1.21 %	-1.39 %	-1.49 %

En la Tabla 5.7 se muestran los resultados obtenidos al realizar la optimización RMP/RLE, es decir, optimizar las cinco redes Mundo pequeño con respecto a propiedades de una red Libre de escala.

Al observar la información proporcionada por la aptitud durante seis iteraciones del AG la mayor calidad se obtiene al realizar 25 mut/iter, pero el realizar 1 mut/iter produce la mayor calidad estructural por movimiento realizado, es decir, realizar 25 mut/iter produce en una mejora del 326 % con un costo igual a 149, mientras que al realizar 1 mut/iter produjo una mejora del 108 % con un costo promedio igual a 7. Al analizar la relación de la mejora en la calidad estructural por movimiento realizado se encuentra que es mejor realizar 1 mut/iter ya que produce una mejora del 15.42 %. En este experimento las redes alternativas con menor calidad y mayor costo (385 y 390 movimientos) se obtuvieron al realizar 735 y 980 mut/iter con una mejora por costo de 0.22 % y 0.21 % respectivamente.

Tabla 5.7: Resultados obtenidos al realizar la optimización de la instancia RMP/RLE bajo once diferentes mutaciones por iteración. Los resultados se presentan como porcentajes promedios de mejora y los costos como el número de modificaciones que se realizaron sobre la topología de la red original.

Obj.	mut/iter	2 %n	10 %n	20 %n	30 %n	50 %n	100 %n	13 %|cb|	20 %|cb|	33 %|cb|	60 %|cb|	80 %|cb|
		1	5	10	15	25	50	153	245	408	735	980
$\Downarrow$	$\delta(G)$	153 %	119 %	131 %	132 %	160 %	204 %	259 %	256 %	292 %	390 %	406 %
$\Uparrow$	$\Delta(G)$	71 %	102 %	130 %	153 %	182 %	221 %	233 %	221 %	234 %	268 %	273 %
$\Downarrow$	costo(G)	7	38	73	106	149	226	249	258	294	385	390
$\Uparrow$	$f(G)$	108 %	218 %	259 %	299 %	326 %	322 %	270 %	222 %	186 %	83 %	83 %
$\Uparrow$	mej/costo	15.42 %	5.74 %	3.54 %	2.82 %	2.19 %	1.42 %	1.08 %	0.86 %	0.63 %	0.22 %	0.21 %

5.4.4 Conclusiones

En base a los resultados de los experimentos anteriores se construyó la Tabla 5.8 donde cada renglón muestra las mut/iter que producen las redes con la mayor y menor calidad estructural (MACA y MECA) y también se muestran las mut/iter que producen las redes con el menor y mayor costo (MECO y MACO) por cada clase de instancia. A la derecha de cada una de las mut/iter se muestra el porcentaje promedio de mejora por movimiento realizado.

Al observar los resultados mostrados en la Tabla 5.8 se puede concluir que el algoritmo genético durante seis iteraciones:

- Construye las soluciones con mayor calidad y con el menor costo con 1 mutación por iteración, que resulta ser la menor cantidad de mutaciones que se pueden realizar por iteración.
- Construye las soluciones con menor calidad con siete diferentes cantidades de mutaciones por iteración: 5, 10, 153, 245, 408, 735 y 980 mutaciones.
- Entre más mutaciones por iteración se realicen por lo general se producen redes alternativas de mayor costo.
- El algoritmo genético bajo ciertos parámetros, tal como un gran número de mutaciones por iteración, puede no encontrar un conjunto de redes alternativas con mayor calidad estructural que la red original. Ejemplo de esto sucedió durante la optimización de la instancia RAL/RLE con 408, 735 y 980 mut/iter, en la instancia RLE/RAL con 735 y 980 mut/iter y en la instancia RMP/RAL con 5, 10, 15, 25, 50, 153, 245, 408, 735 y 980 mut/iter. En estos casos se recomienda mantener la topología de la red original.

Tabla 5.8: Comparación de las cantidades de mutaciones que producen las redes con los mejores (MACA) y peores (MECA) porcentajes de mejora, también se muestran las redes de menor (MECO) y mayor (MACO) costo para cada una de las instancias optimizadas.

Instancia	MACA	mej/costo	MECA	mej/costo	MECO	Costo	MACO	Costo
RAL/RLE	1	48.5 %	980	-3.76 %	1	8	980	290
RAL/RMP	1	0.25 %	5, 10	-0.12 %, -0.19 %	1	4	408, 735, 980	613, 614
RLE/RAL	1	0.56 %	735, 980	-0.0001 %, -0.004 %	1	9	980	444
RLE/RMP	1	0.43 %	153, 245, 408, 735, 980	0.01 %, 0.03 %	1	7	980	481
RMP/RAL	1	1 %	980	-1.49 %	1	5	980	324
RMP/RLE	1	15.42 %	735, 980	0.22 %, 0.21 %	1	7	980	390

5.5 Experimentación: cantidad de iteraciones

El objetivo de este experimento es determinar el número de iteraciones a partir del cual el valor de la aptitud muestre un comportamiento asintótico ya que esto indica la convergencia de las soluciones.

Para realizar este experimento se consideró la cantidad de mutaciones por iteración que generaron durante seis iteraciones las redes de mejor y peor calidad estructural en base a la relación del porcentaje de mejora y el costo al realizar un solo movimiento (mej/costo). Por tanto, se analizó el comportamiento que producen éstas cantidades de mutaciones por iteración sobre las soluciones generadas durante 1000 iteraciones con una P_i igual a 30. Los resultados obtenidos al realizar este experimento se muestran en las Figuras 5.2, 5.3 y 5.4 donde se muestra el valor promedio de las aptitudes con sus respectivas desviaciones en los casos que corresponde.

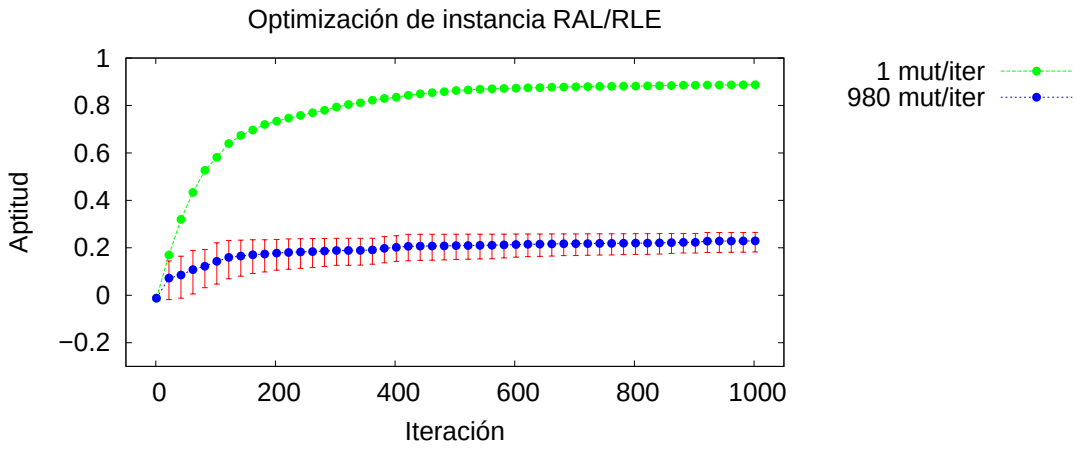


Figura 5.2: Desarrollo de la aptitud en la instancia RAL/RLE al optimizarla durante 1000 iteraciones del AG. Los valores de las aptitudes son los valores promedios (con sus respectivas desviaciones mostradas en color rojo) de las mejores 30 aptitudes obtenidas en la última iteración del AG durante una repetición de la herramienta de software.

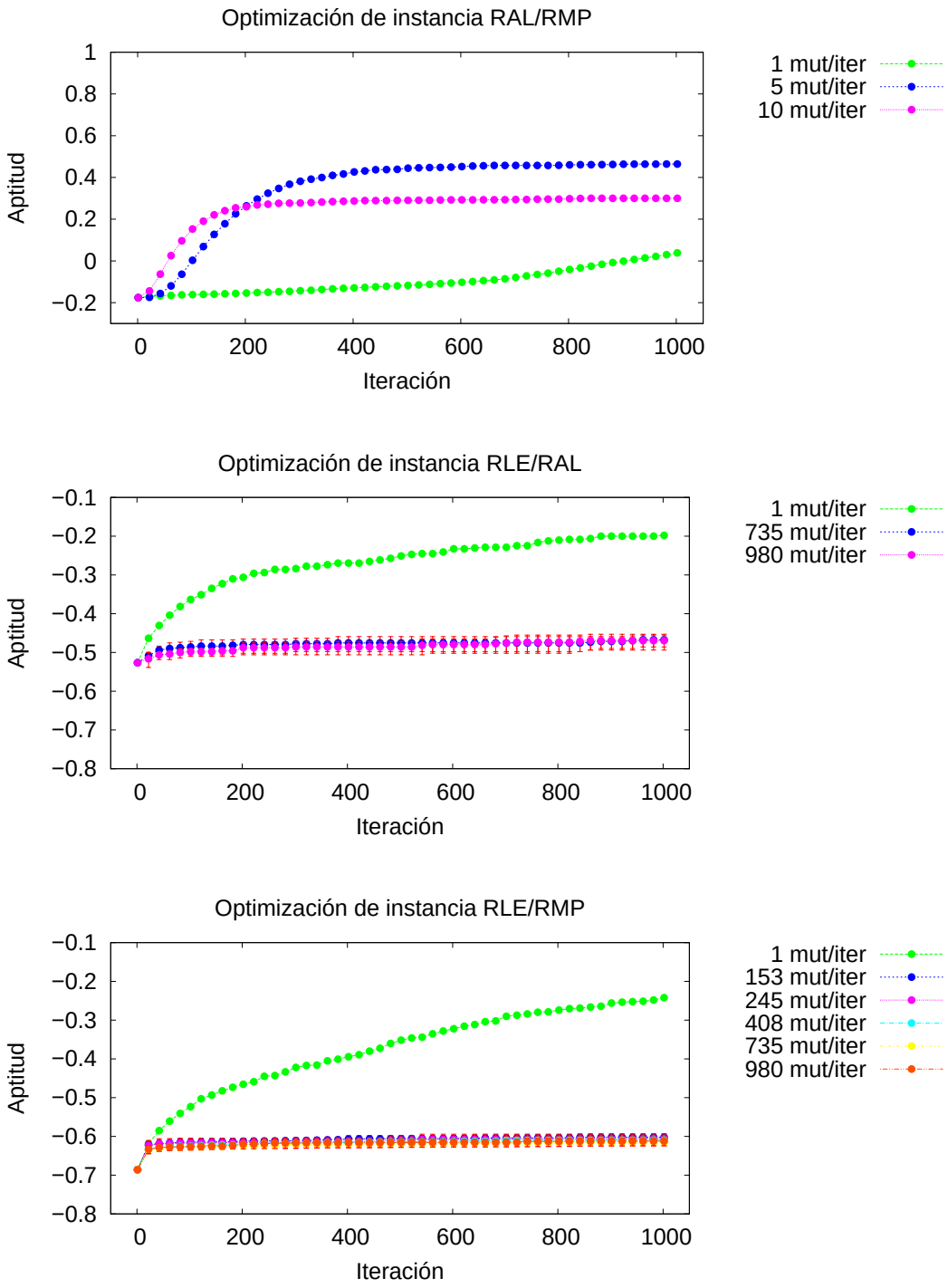


Figura 5.3: Desarrollo de la aptitud en las instancias RAL/RMP, RLE/RAL y RLE/RMP al optimizarlas durante 1000 iteraciones del AG. Los valores de las aptitudes son los valores promedios (con sus respectivas desviaciones mostradas en color rojo) de las mejores 30 aptitudes obtenidas en la última iteración del AG durante una repetición de la herramienta de software.

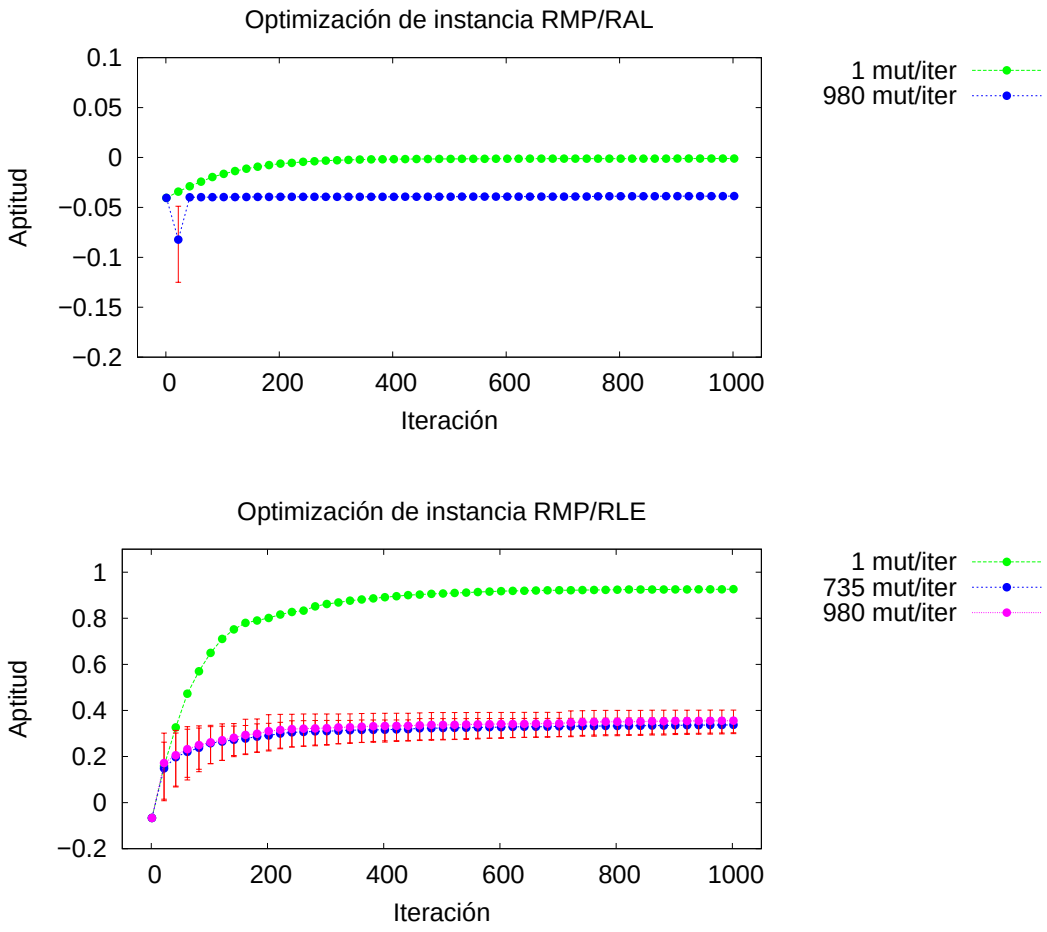


Figura 5.4: Desarrollo de la aptitud en las instancias RMP/RAL y RMP/RLE al optimizarlas durante 1000 iteraciones del AG. Los valores de las aptitudes son los valores promedios (con sus respectivas desviaciones mostradas en color rojo) de las mejores 30 aptitudes obtenidas en la última iteración del AG durante una repetición de la herramienta de software.

Para tener un mejor entendimiento sobre como mejoran los valores de las aptitudes se construyeron las Tablas 5.9 y 5.10. En estas tablas se muestran la misma información que las Figuras 5.2, 5.3 y 5.4, pero se presentan las aptitudes como porcentajes promedios de mejora al ser comparados con la aptitud de la red original, se muestra el costo involucrado al diseñar tal topología y se muestra (en base a la aptitud y el costo de la red creada) la relación del porcentaje de mejora y el costo al realizar un solo movimiento sobre la topología de la red original.

Tabla 5.9: Porcentajes promedios de mejora de las redes alternativas al compararlas con la aptitud de la red original al resolver las clases de instancias RAL/RLE, RAL/RMP, RLE/RAL y RLE/RMP durante 1000 iteraciones del AG. Los porcentajes promedios de mejora corresponden a las 30 mejores aptitudes obtenidas en la última iteración del AG durante una repetición de la herramienta de software

Instancia	mut/iter	Par.	6i	50i	100i	300i	500i	800i	1000i
RAL/RLE	1	$f(G)$	388 %	3081 %	4798 %	6507 %	7045 %	7217 %	7259 %
		costo	8	66	108	214	247	250	251
		mej/costo	48.5 %	46.68 %	44.43 %	30.41 %	28.52 %	28.87 %	28.92 %
	980	$f(G)$	-1090 %	896 %	1128 %	1593 %	1803 %	1873 %	1950 %
		costo	290	107	98	73	66	69	65
		mej/costo	-37.59 %	83.74 %	11.51 %	21.82 %	27.32 %	27.14 %	30 %
RAL/RMP	1	$f(G)$	1 %	5 %	8 %	22 %	39 %	77 %	122 %
		costo	4	34	60	183	272	348	391
		mej/costo	0.25 %	0.15 %	0.13 %	0.12 %	0.14 %	0.22 %	0.31 %
	5	$f(G)$	-0.6 %	21 %	103 %	315 %	349 %	362 %	364 %
		costo	5	254	499	792	825	838	838
		mej/costo	-0.12 %	0.08 %	0.21 %	0.40 %	0.42 %	0.43 %	0.43 %
	10	$f(G)$	-2.3 %	94 %	189 %	261 %	266 %	269 %	270 %
		costo	12	523	682	770	775	767	767
		mej/costo	-0.19 %	0.18 %	0.28 %	0.34 %	0.34 %	0.35 %	0.35 %
RLE/RAL	1	$f(G)$	5 %	19 %	29 %	45 %	53 %	60 %	62 %
		costo	9	33	90	180	195	211	231
		mej/costo	0.56 %	0.58 %	0.32 %	0.25 %	0.27 %	0.28 %	0.27 %
	735	$f(G)$	-0.39 %	6 %	7 %	8 %	9 %	10 %	11 %
		costo	414	190	201	190	182	140	123
		mej/costo	-0.001 %	0.03 %	0.03 %	0.04 %	0.05 %	0.07 %	0.09 %
	980	$f(G)$	-1.94 %	5 %	7 %	8 %	8 %	10 %	11 %
		costo	444	188	171	146	145	115	104
		mej/costo	-0.004 %	0.03 %	0.04 %	0.05 %	0.06 %	0.09 %	0.11 %
RLE/RMP	1	$f(G)$	3 %	15 %	23 %	41 %	50 %	60 %	65 %
		costo	7	35	53	155	244	281	287
		mej/costo	0.43 %	0.43 %	0.43 %	0.26 %	0.2 %	0.21 %	0.23 %
	153	$f(G)$	9 %	10 %	11 %	11 %	12 %	12 %	12 %
		costo	344	399	405	408	385	329	336
		mej/costo	0.03 %	0.03 %	0.03 %	0.03 %	0.03 %	0.04 %	0.04 %
	245	$f(G)$	8 %	9 %	10 %	10 %	11 %	12 %	12 %
		costo	379	398	420	403	383	399	388
		mej/costo	0.02 %	0.02 %	0.02 %	0.02 %	0.03 %	0.03 %	0.03 %
	408	$f(G)$	7 %	9 %	9 %	10 %	10 %	11 %	11 %
		costo	444	405	445	462	438	395	395
		mej/costo	0.02 %	0.02 %	0.02 %	0.02 %	0.02 %	0.03 %	0.03 %

Tabla 5.10: Porcentajes promedios de mejora de las redes alternativas al compararlas con la aptitud de la red original al resolver las clases de instancias RLE/RMP, RMP/RAL y RMP/RLE durante 1000 iteraciones del AG. Los porcentajes promedios de mejora corresponden a las 30 mejores aptitudes obtenidas en la última iteración del AG durante una repetición de la herramienta de software.

Instancia	mut/iter	Par.	6i	50i	100i	300i	500i	800i	1000i
RLE/RMP	735	$f(G)$	6 %	8 %	8 %	10 %	10 %	10 %	10 %
		costo	473	412	451	482	464	441	422
		mej/costo	0.01 %	0.02 %	0.02 %	0.02 %	0.02 %	0.02 %	0.02 %
	980	$f(G)$	7 %	8 %	8 %	9 %	9 %	11 %	11 %
		costo	481	427	438	454	466	433	412
		mej/costo	0.01 %	0.02 %	0.02 %	0.02 %	0.02 %	0.03 %	0.03 %
RMP/RAL	1	$f(G)$	5 %	36 %	61 %	93 %	97 %	97 %	97 %
		costo	5	59	103	175	192	196	201
		mej/costo	1 %	0.61 %	0.59 %	0.53 %	0.51 %	0.5 %	0.48 %
	980	$f(G)$	-484 %	2 %	2 %	3 %	3 %	4 %	5 %
		costo	293	2	2	5	7	11	12
		mej/costo	1.65 %	1 %	1 %	0.6 %	0.43 %	0.36 %	0.42 %
RMP/RLE	1	$f(G)$	108 %	739 %	1141 %	1402 %	1467 %	1487 %	1491 %
		costo	7	68	123	203	220	222	219
		mej/costo	15.43 %	10.87 %	9.28 %	6.91 %	6.67 %	6.7 %	6.81 %
	735	$f(G)$	83 %	431 %	488 %	574 %	597 %	600 %	606 %
		costo	385	178	123	89	84	88	89
		mej/costo	0.22 %	2.42 %	3.97 %	6.45 %	7.11 %	6.82 %	6.81 %
	980	$f(G)$	83 %	399 %	471 %	563 %	593 %	627 %	634 %
		costo	390	196	127	92	88	87	84
		mej/costo	0.21 %	2.04 %	3.71 %	6.12 %	6.74 %	7.21 %	7.55 %

En el caso de las instancias RLE/RAL y RLE/RMP se observa que realizar 1 mut/iter durante 1000 iteraciones produce las mejores soluciones, pero no se logra una convergencia en los valores de las aptitudes. Por tal motivo se repitieron los experimentos de éstas dos instancias aumentando la cantidad de iteraciones a 2000. Los resultados obtenidos se muestran en la Figura 5.5.

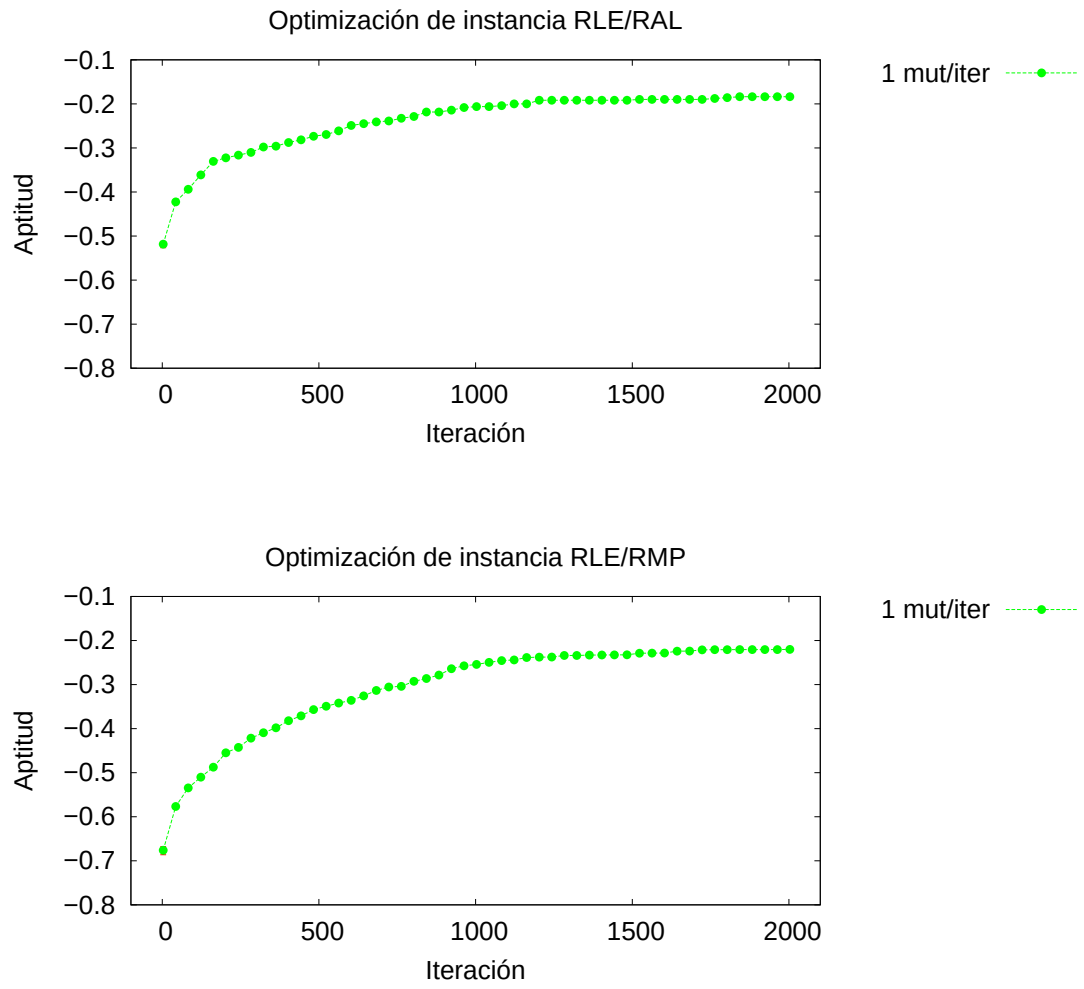


Figura 5.5: Desarrollo de la aptitud en las instancias RLE/RAL y RLE/RMP al optimizarlas durante 2000 iteraciones del AG. Los valores de las aptitudes son los valores promedios de las mejores 30 aptitudes obtenidas en la última iteración del AG durante una repetición de la herramienta de software.

Estos mismos resultados se presentan en la Tabla 5.11, pero se presentan las aptitudes como porcentajes promedios de mejora al ser comparados con la aptitud de la red original, se muestra el costo involucrado al diseñar tal topología y se muestra la relación del porcentaje de mejora y el costo al realizar un solo movimiento sobre la topología de la red original.

Tabla 5.11: Porcentajes promedios de mejora de las redes alternativas al compararlas con la aptitud de la red original al resolver las clases de instancias RLE/RAL y RLE/RMP durante 2000 iteraciones del AG. Los porcentajes promedios de mejora mostrados corresponden a las 30 mejores aptitudes obtenidas en la última iteración del AG durante una repetición de la herramienta de software.

Instancia	mut/iter	Par.	1000i	1300i	1500i	1800i	2000i
RLE/RAL	1	$f(G)$	62 %	64 %	64 %	65 %	65 %
		costo	231	244	244	247	248
		mej/costo	0.27 %	0.26 %	0.26 %	0.26 %	0.26 %
RLE/RMP	1	$f(G)$	65 %	66 %	66 %	68 %	68 %
		costo	287	289	289	289	290
		mej/costo	0.23 %	0.23 %	0.23 %	0.24 %	0.23 %

5.5.1 Conclusiones

Al analizar los resultados obtenidos de las seis clases de instancias a través de las Figuras 5.2, 5.3, 5.4 y 5.5 y en las Tablas 5.9, 5.10 y 5.11 se obtuvieron las siguientes conclusiones:

- En la optimización RAL/RLE se observa que las mejores aptitudes se obtienen al realizar 1 mut/iter y después de 500 iteraciones las aptitudes de las redes alternativas muestran un incremento moderado.

- En el caso la optimización RAL/RMP a partir de 50 iteraciones las redes obtenidas al realizar 5 y 10 mut/iter tienen una aptitud superior a la aptitud de las redes obtenidas al realizar 1 mut/iter. La importancia de esto radica en que anteriormente al realizar seis iteraciones con 1 mut/iter se producía el conjunto de redes con mayor calidad estructural y ahora al realizar 5 mut/iter con por lo menos 50 iteraciones produce las mejores soluciones. Durante esta optimización el valor de la aptitud de las redes alternativas obtenidas con 5

mut/iter comenzó a tener un comportamiento asintótico alrededor de las 500 iteraciones.

- En la optimización de RLE/RAL las mejores aptitudes se obtienen al realizar 1 mut/iter y es hasta las 1300 iteraciones que comienza un comportamiento asintótico en los valores de las aptitudes.
- En la optimización RLE/RMP las mejores aptitudes se obtienen al realizar 1 mut/iter y al llegar a las 1300 iteraciones se observa un comportamiento asintótico en los valores de las aptitudes.
- En la optimización RMP/RAL las mejores aptitudes se obtienen al realizar 1 mut/iter y es alrededor de 500 iteraciones que comienza un comportamiento asintótico en los valores de las aptitudes.
- En la optimización RMP/RLE las mejores aptitudes se obtienen al realizar 1 mut/iter y es alrededor de las 500 iteraciones que comienza un comportamiento asintótico en los valores de las aptitudes.
- Por lo general entre mayores iteraciones se realicen, la calidad estructural y su costo se incrementan.

Estas mismas conclusiones se pueden resumir en la Tabla 5.12.

Tabla 5.12: Cantidad de mutaciones por iteración y cantidad de iteraciones que se recomiendan utilizar en la herramienta de software para generar el conjunto de redes alternativas con la mayor calidad estructural posible.

Instancia	mut/iter	P_i	Iteraciones	Mejora	costo	mej/costo
RAL/RLE	1	30	500	7045 %	247	28.52 %
RAL/RMP	5	30	500	349 %	825	0.42 %
RLE/RAL	1	30	1300	64 %	244	0.26 %
RLE/RMP	1	30	1300	66 %	289	0.23 %
RMP/RAL	1	30	500	97 %	192	0.51 %
RMP/RLE	1	30	500	1467 %	220	6.67 %

5.6 Experimentación: tamaño de la población inicial

Para determinar el tamaño promedio de la P_i se realizaron experimentos en las seis clases de instancias mencionadas en la Sección 5.3. Las optimizaciones se realizaron con una P_i igual a 30, 50 y 100 individuos. El número de aristas mutadas por iteración y la cantidad de iteraciones para este experimento se tomaron en base a las conclusiones obtenidas en la Sección 5.5.1. Los resultados obtenidos de este experimento se muestran en las Figuras 5.6, 5.7 y 5.8.

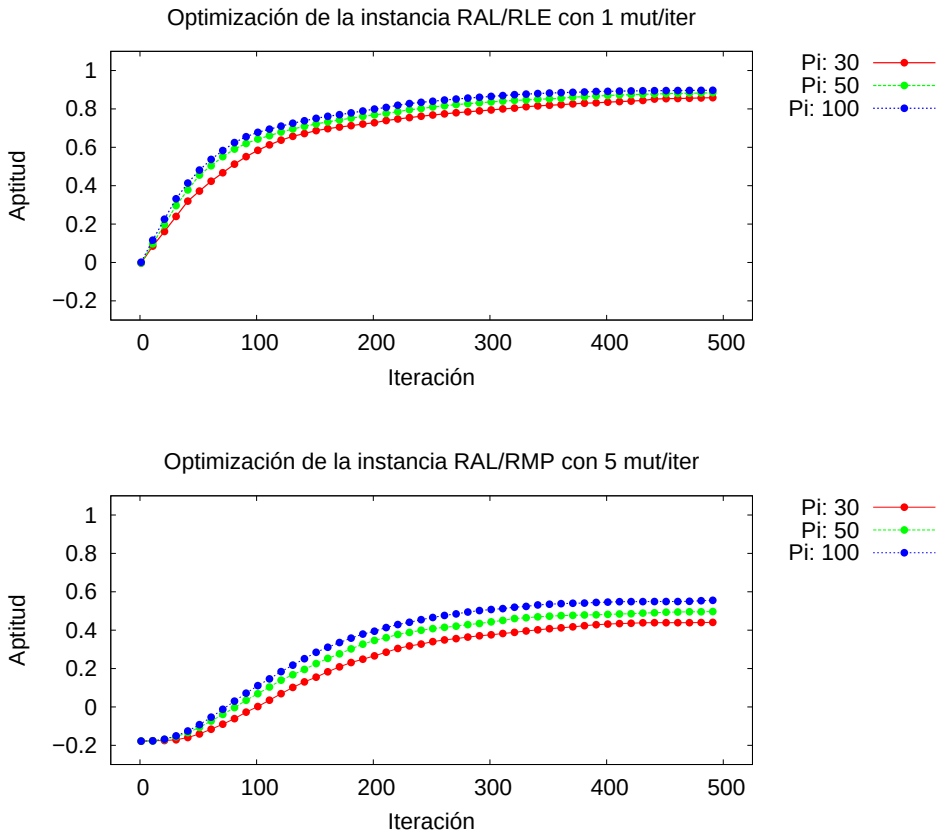


Figura 5.6: Desarrollo de la aptitud en las instancias RAL/RLE y RAL/RMP con una P_i igual a 30, 50 y 100 individuos durante 500 iteraciones del AG. Los valores de las aptitudes son los valores promedios de las mejores aptitudes obtenidas en la última iteración del AG durante una repetición de la herramienta de software.

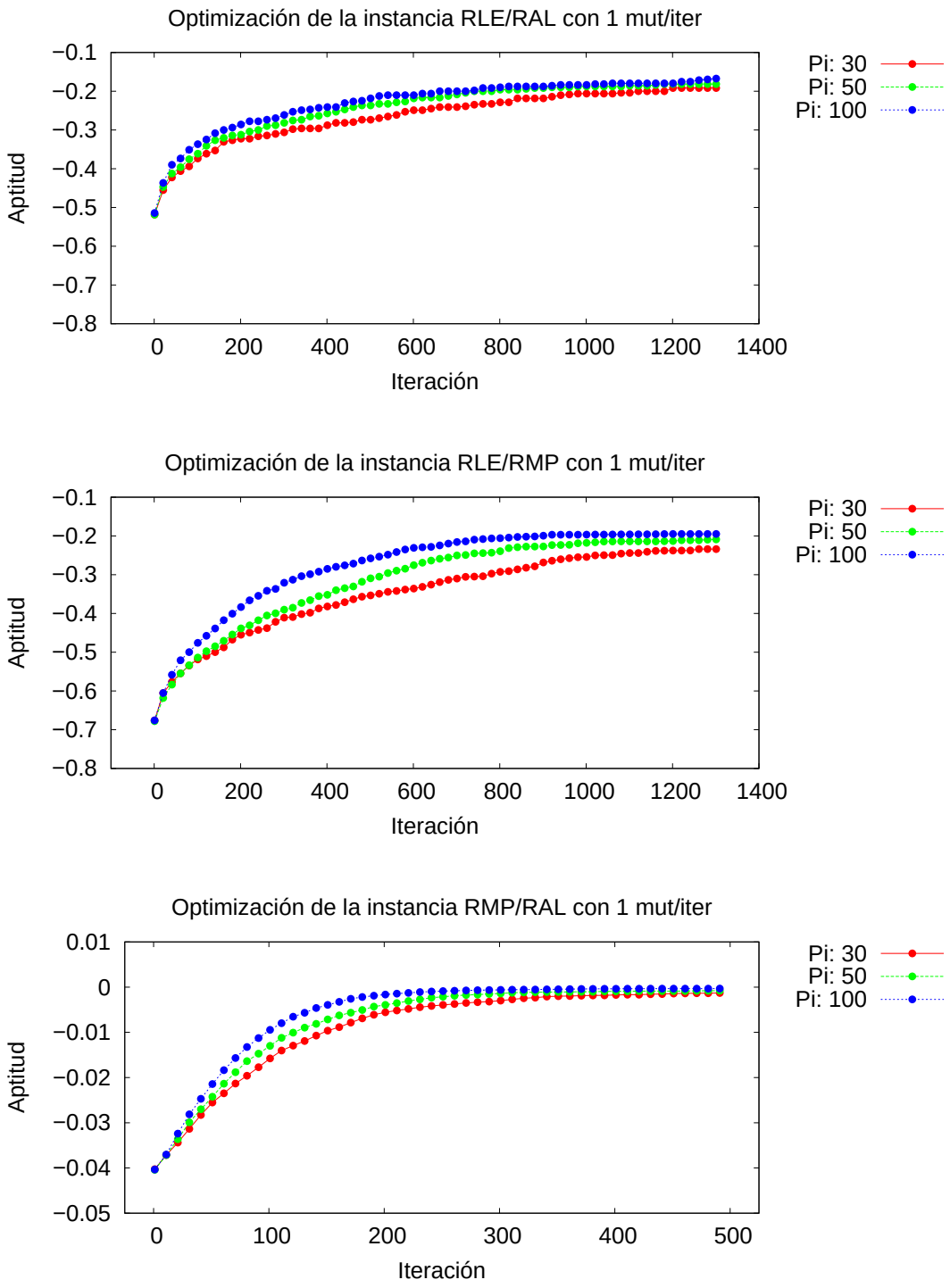


Figura 5.7: Desarrollo de la aptitud en las instancias RLE/RAL, RLE/RMP durante 1300 iteraciones y RMP/RAL durante 500 iteraciones, con una P_i igual a 30, 50 y 100 individuos del AG. Los valores de las aptitudes son los valores promedios de las mejores aptitudes obtenidas en la última iteración del AG durante una repetición de la herramienta de software.

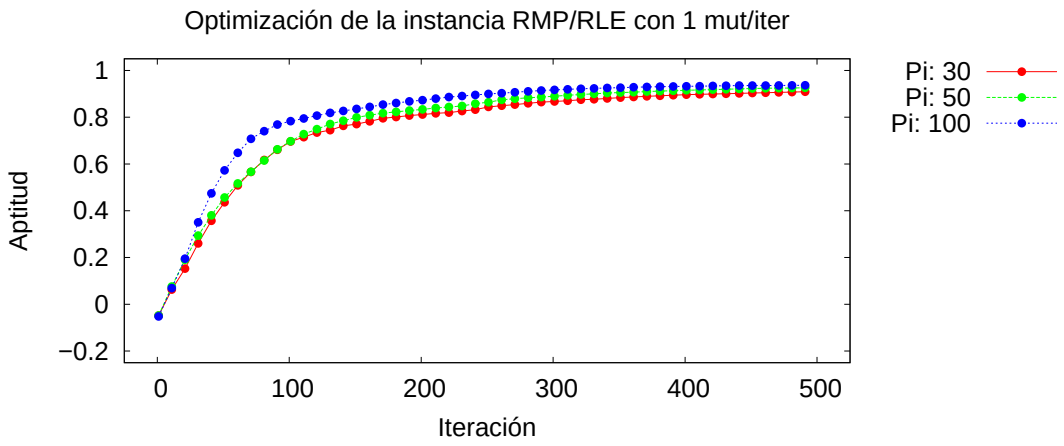


Figura 5.8: Desarrollo de la aptitud en la instancia RMP/RLE con una P_i a 30, 50 y 100 individuos durante 500 iteraciones del AG. Los valores de las aptitudes son los valores promedios de las mejores aptitudes obtenidas en la última iteración del AG durante una repetición de la herramienta de software.

Al analizar los resultados mostrados en las Figuras 5.6, 5.7 y 5.8 se determina que entre mayor sea el tamaño de la población inicial el algoritmo genético tiene un mayor campo de búsqueda de soluciones y por tanto encuentra las mejores soluciones con una P_i igual a 100 individuos seguidos por una P_i igual a 50 y por último por una P_i igual a 30.

En las Tablas 5.13 y 5.14 se comparan los mismos resultados obtenidos en las Figuras 5.6, 5.7 y 5.8, pero se presentan las aptitudes como porcentajes promedios de mejora al ser comparados con la aptitud de la red original, se muestra el costo involucrado al diseñar tal topología y se muestra la relación del porcentaje de mejora y el costo al realizar un solo movimiento sobre la topología de la red original.

Tabla 5.13: Comparación del desarrollo de la aptitud con poblaciones iniciales de tamaño 30, 50 y 100 en instancias RAL/RLE, RAL/RMP, RMP/RAL y RMP/RLE durante 500 iteraciones del algoritmo genético. Los porcentajes promedios de mejora mostrados corresponden a las mejores aptitudes obtenidas en la última iteración del AG durante una repetición de la herramienta de software.

Instancia	mut/iter	P_i	Parám.	6i	50i	100i	300i	500i
RAL/RLE	1	30	$f(G)$	388 %	3081 %	4798 %	6507 %	7045 %
			costo	8	66	108	214	247
			mej/costo	48.5 %	46.68 %	44.43 %	30.41 %	28.52 %
		50	$f(G)$	398 %	3700 %	5268 %	6846 %	7245 %
			costo	9	69	118	221	240
			mej/costo	44.22 %	53.62 %	44.64 %	30.98 %	30.19 %
		100	$f(G)$	561 %	3917 %	5555 %	7079 %	7342 %
			costo	9	66	115	226	236
			mej/costo	62.33 %	59.34 %	48.30 %	31.32 %	31.11 %
RAL/RMP	5	30	$f(G)$	-0.6 %	21 %	103 %	315 %	349 %
			costo	5	254	499	792	825
			mej/costo	-0.12 %	0.08 %	0.21 %	0.4 %	0.42 %
		50	$f(G)$	-0.6 %	38 %	137 %	352 %	384 %
			costo	6	332	565	819	844
			mej/costo	-0.1 %	0.11 %	0.24 %	0.43 %	0.45 %
		100	$f(G)$	-0.6 %	45 %	161 %	388 %	416 %
			costo	6	335	573	844	869
			mej/costo	-0.1 %	0.13 %	0.28 %	0.46 %	0.48 %
RMP/RAL	1	30	$f(G)$	5 %	36 %	61 %	93 %	97 %
			costo	5	59	103	175	192
			mej/costo	1 %	0.61 %	0.59 %	0.53 %	0.51 %
		50	$f(G)$	4 %	39 %	68 %	97 %	98 %
			costo	7	60	103	171	184
			mej/costo	0.57 %	0.65 %	0.66 %	0.57 %	0.53 %
		100	$f(G)$	4 %	46 %	76 %	98 %	99 %
			costo	8	63	108	165	169
			mej/costo	0.5 %	0.73 %	0.7 %	0.59 %	0.59 %
RMP/RLE	1	30	$f(G)$	108 %	739 %	1141 %	1402 %	1467 %
			costo	7	68	123	203	220
			mej/costo	15.43 %	10.87 %	9.28 %	6.91 %	6.67 %
		50	$f(G)$	125 %	767 %	1143 %	1435 %	1490 %
			costo	10	67	113	205	215
			mej/costo	12.5 %	11.45 %	10.12 %	7 %	6.93 %
		100	$f(G)$	117 %	944 %	1273 %	1476 %	1507 %
			costo	9	73	118	205	211
			mej/costo	13 %	12.93 %	10.79 %	7.2 %	7.14 %

Tabla 5.14: Comparación del desarrollo de la aptitud con poblaciones iniciales de tamaño 30, 50 y 100 en instancias RLE/RAL y RLE/RMP durante 1300 iteraciones del algoritmo genético. Los porcentajes promedios de mejora mostrados corresponden a las mejores aptitudes obtenidas en la última iteración del AG durante una repetición de la herramienta de software.

Instancia	mut/iter	P_i	Parám.	300i	500i	800i	1000i	1300i
RLE/RAL	1	30	$f(G)$	45 %	53 %	60 %	62 %	64 %
			costo	180	195	211	231	244
			mej/costo	0.25 %	0.27 %	0.28 %	0.27 %	0.26 %
		50	$f(G)$	47 %	57 %	63 %	64 %	66 %
			costo	170	186	207	241	238
			mej/costo	0.28 %	0.31 %	0.30 %	0.27 %	0.28 %
		100	$f(G)$	49 %	60 %	64 %	65 %	68 %
			costo	173	196	220	239	233
			mej/costo	0.28 %	0.31 %	0.29 %	0.27 %	0.29 %
RLE/RMP	1	30	$f(G)$	41 %	50 %	60 %	65 %	66 %
			costo	155	244	281	287	289
			mej/costo	0.26 %	0.2 %	0.21 %	0.23 %	0.23 %
		50	$f(G)$	44 %	57 %	65 %	68 %	70 %
			costo	172	258	277	289	287
			mej/costo	0.26 %	0.22 %	0.23 %	0.24 %	0.24 %
		100	$f(G)$	53 %	62 %	70 %	71 %	72 %
			costo	183	246	276	283	278
			mej/costo	0.29 %	0.25 %	0.25 %	0.25 %	0.26 %

5.6.1 Conclusiones

Al observar las Figuras 5.6, 5.7 y 5.8 y las Tablas 5.13 y 5.14 se encontró que para las seis clases de instancias los mejores porcentajes promedios de mejora en la aptitud se obtienen con una población inicial igual a 100 individuos durante 500 iteraciones para las instancias de redes Aleatorias y Mundo pequeño, mientras que para las instancias Libre de escala se recomiendan 1300 iteraciones.

5.7 Experimentación: confiabilidad de los resultados

El objetivo de este experimento es demostrar que la herramienta de software otorga resultados robustos, por lo cual no es necesario realizar repeticiones independientes de la herramienta de software para obtener mejores soluciones. Para comprobar la robustez de las soluciones se tomó una red por cada clase de instancia y se repitió independientemente la ejecución de la herramienta de software 30 veces para resolverla. El algoritmo genético se ejecutó con una población inicial igual a 50, 15 mutaciones por iteración, 100 iteraciones para redes Aleatorias y Libre de escala y 300 iteraciones para redes Mundo pequeño.

Los resultados obtenidos se muestran en las Figuras 5.9, 5.10 y 5.11 donde se muestra el valor promedio de las mejores 50 aptitudes obtenidas en la última iteración del AG, además de mostrar sus respectivas desviaciones. Cabe destacar que la magnitud de las soluciones obtenidas son diferentes para cada gráfica mostrada, por lo cual es de interés poner atención a éstas.

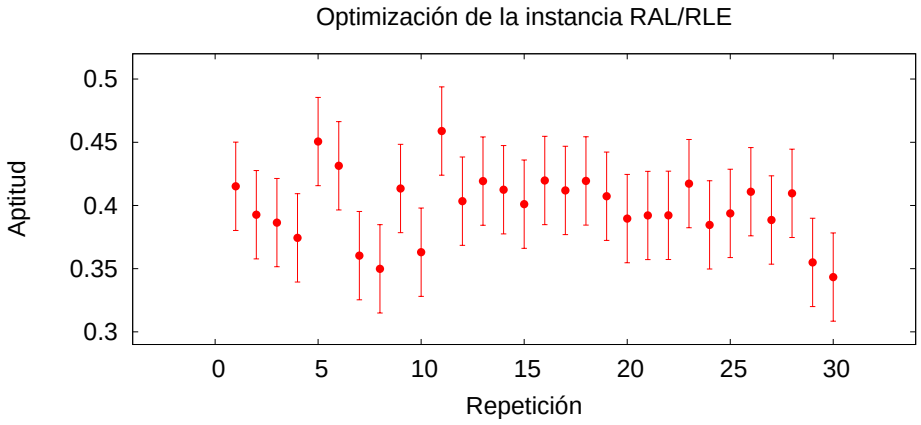


Figura 5.9: Variabilidad de las aptitudes obtenidas al resolver la instancia RAL/RLE durante 30 repeticiones independientes de la herramienta de software. Los valores de las aptitudes son los valores promedios de las mejores 50 aptitudes obtenidas en la última iteración del AG.

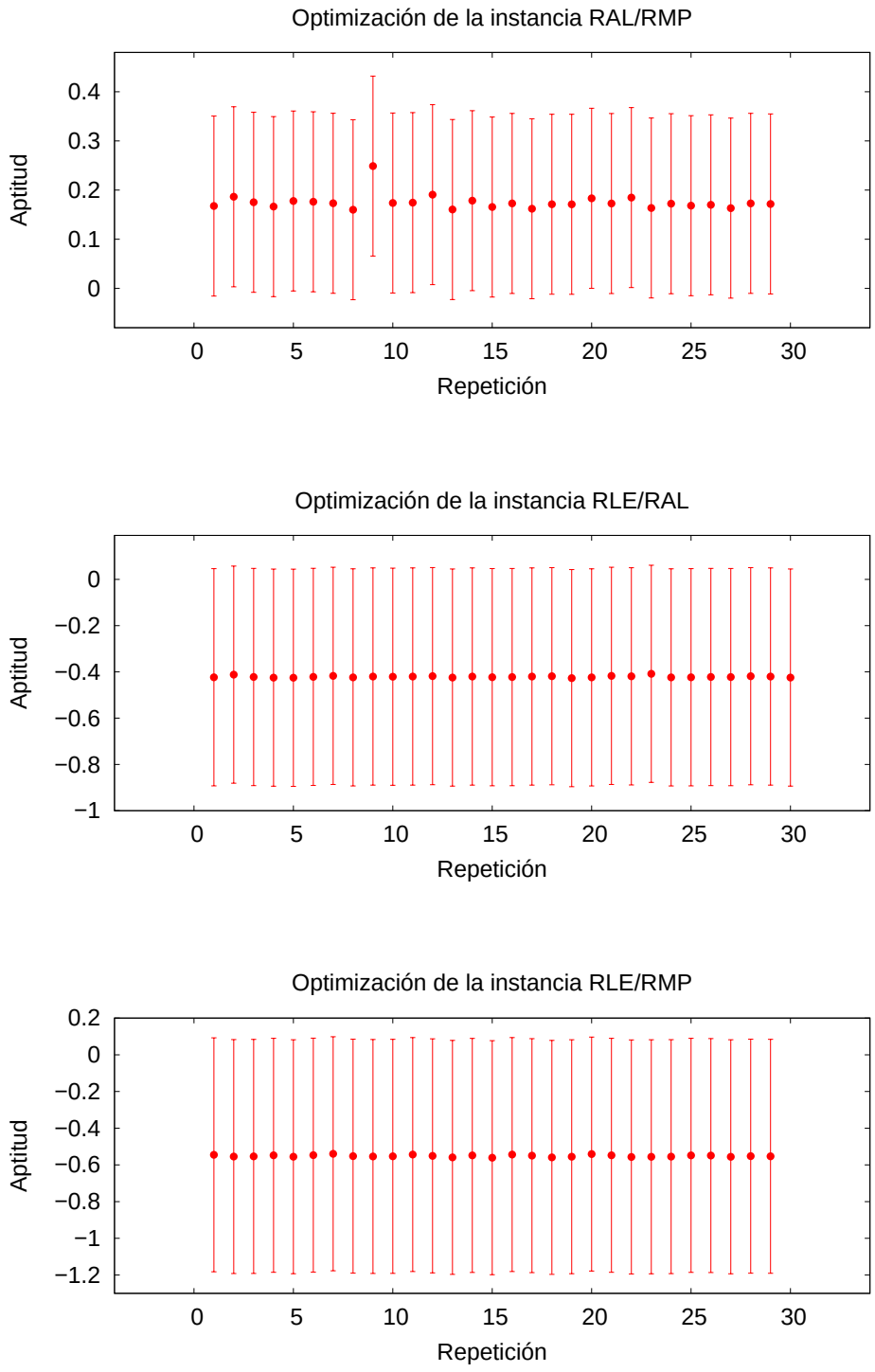


Figura 5.10: Variabilidad de las aptitudes obtenidas al resolver las instancias RAL/RMP, RLE/RAL y RLE/RMP durante 30 repeticiones independientes de la herramienta de software. Los valores de las aptitudes son los valores promedios de las mejores 50 aptitudes obtenidas en la última iteración del AG.

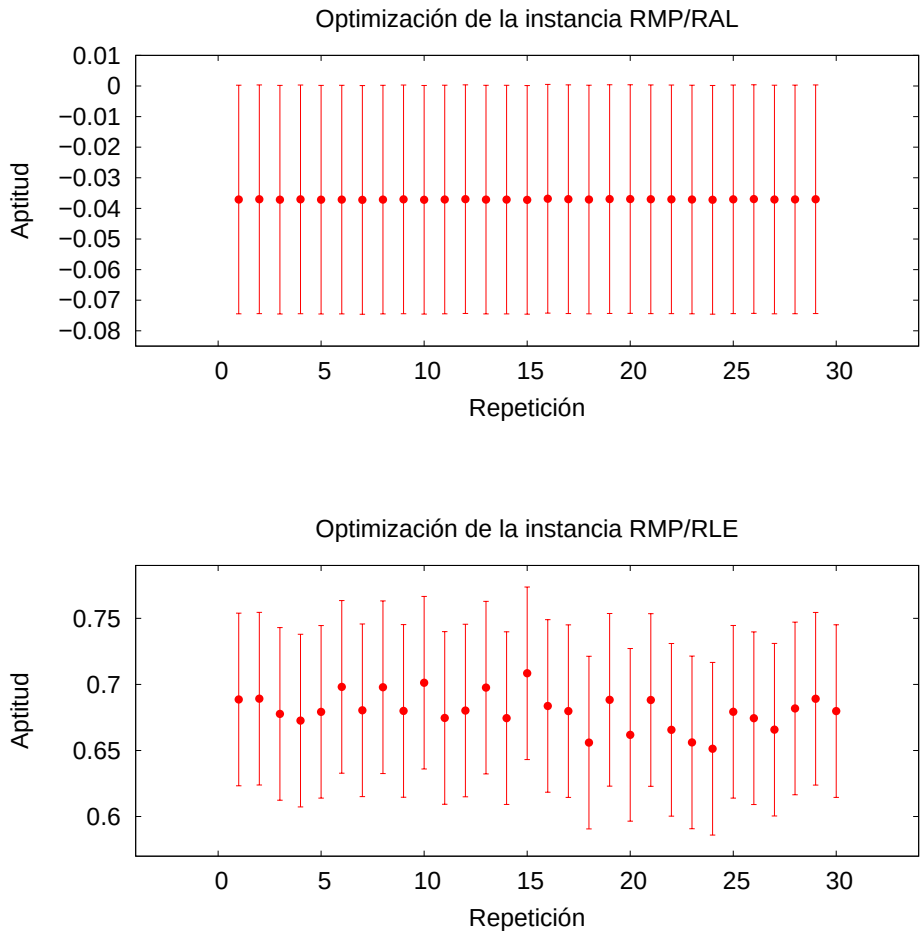


Figura 5.11: Variabilidad de las aptitudes obtenidas al resolver las instancias RMP/RAL y RMP/RLE durante 30 repeticiones independientes de la herramienta de software. Los valores de las aptitudes son los valores promedios de las mejores 50 aptitudes obtenidas en la última iteración del AG.

5.7.1 Conclusiones

En las Figuras 5.9, 5.10 y 5.11 se observa que los resultados arrojados por la herramienta de software en las instancias RAL/RMP, RLE/RAL, RLE/RMP, RMP/RAL y RMP/RLE se encuentran en el mismo rango, por lo cual basta una repetición de la herramienta de software para obtener el conjunto de redes alternativas con la mayor calidad estructural posible.

En el caso de la instancia RAL/RLE los resultados arrojados por la herramienta de software muestran un pequeño margen de variabilidad el cual se debe a que las soluciones de mayor aptitud son las encontradas generalmente durante un mayor número de iteraciones del algoritmo genético. Los resultados anteriores indican un buen funcionamiento de la herramienta de software por lo cual en este trabajo los resultados mostrados se han obtenido durante una repetición. Se recomienda al usuario realizar también una repetición de la herramienta de software ya que los resultados arrojados han demostrado ser robustos.

5.8 Experimentación: tiempos de ejecución

Para determinar el tiempo de cómputo que tarda la herramienta de software en resolver un problema de optimización estructural en redes complejas, se calculó para grafos de diferente orden, el tamaño de la cadena binaria con la cual trabaja el algoritmo genético y se midió el tiempo de ejecución que toma optimizar por separado cada una de las propiedades.

Los tiempos obtenidos se desarrollaron durante una iteración del algoritmo genético, una población inicial igual a 30 y mutando el 33 % del número de elementos de la cadena binaria. Estos tiempos obtenidos se muestran en la Tabla 5.8 donde el primer renglón de la tabla indica el orden del grafo para los cuales se calcularon las métricas disponibles en la herramienta de software. Estas métricas se indican en la columna a la izquierda de la misma tabla y en el último renglón se muestra el tiempo total en segundos que tarda la herramienta de software al calcular las ocho métricas de una red de n vértices.

Tabla 5.15: Tiempos de ejecución en segundos que toma el algoritmo genético para calcular cada métrica de una red de n vértices durante una iteración del algoritmo genético.

Mét/n	$50n$	$100n$	$200n$	$300n$	$400n$	$600n$	$800n$	$1000n$
|cb|	1225	4950	19900	44850	79800	179700	319600	499500
$\mathcal{E}(G)$	0.23	5	35	39	93	328	727	1429
$D(G)$	0.23	6	35	39	93	302	719	1435
$C(G)$	0.25	7	41	45	187	403	780	1874
$\delta(G)$	0.23	5	35	38	156	308	736	1441
$\Delta(G)$	0.23	5	35	38	166	310	731	1430
$\langle k \rangle(G)$	0.23	5	35	38	196	311	733	1430
$L(G)$	0.23	5	29	39	183	311	739	1437
$\Upsilon(G)$	9	200	2289	11605	36396	178353	180000	180000
Total	10.63	238	2534	11881	37470	180626	185165	190476

Durante la ejecución del algoritmo se consideró un límite de tiempo de 180000 segundos. Este tiempo límite se alcanzó al optimizar la vulnerabilidad en grafos de 800 y 1000 vértices.

A través de la Tabla 5.8 se puede calcular para cualquier número de iteración el tiempo que tardará la herramienta de software en realizar la optimización estructural en grafos de hasta 1000 vértices. Este tiempo calculado otorga una aproximación del tiempo de ejecución que tardará la herramienta de software en otorgar soluciones. Se espera que entre un mayor número de iteraciones que se realizen se ahorre tiempo computacional ya que la herramienta de software cuenta con varias estrategias computacionales las cuales se abordan en el Apéndice B.2.

Tomando en cuenta los tiempos de ejecución anteriores se puede calcular el tiempo de cómputo esperado al realizar una optimización con cualquier combinación de propiedades. Por ejemplo, para obtener el tiempo esperado al optimizar en un grafo de 50 vértices las ocho propiedades durante seis iteraciones se sumaría cada uno de los tiempos obtenidos al optimizar todas las propiedades (10.63 segundos) y se le multiplicaría por la cantidad de iteraciones que se pretenden realizar, (10.63 segundos $\cdot$ 6 iteraciones), que resulta en 63.78 segundos o 1.063 minutos que se deben esperar para obtener el conjunto de mejores soluciones.

Los tiempos obtenidos en la Tabla 5.8 se pueden apreciar también en las Figuras 5.12, 5.13 y 5.14 donde claramente se observa el incremento de los tiempos conforme aumenta el número de vértices del grafo.

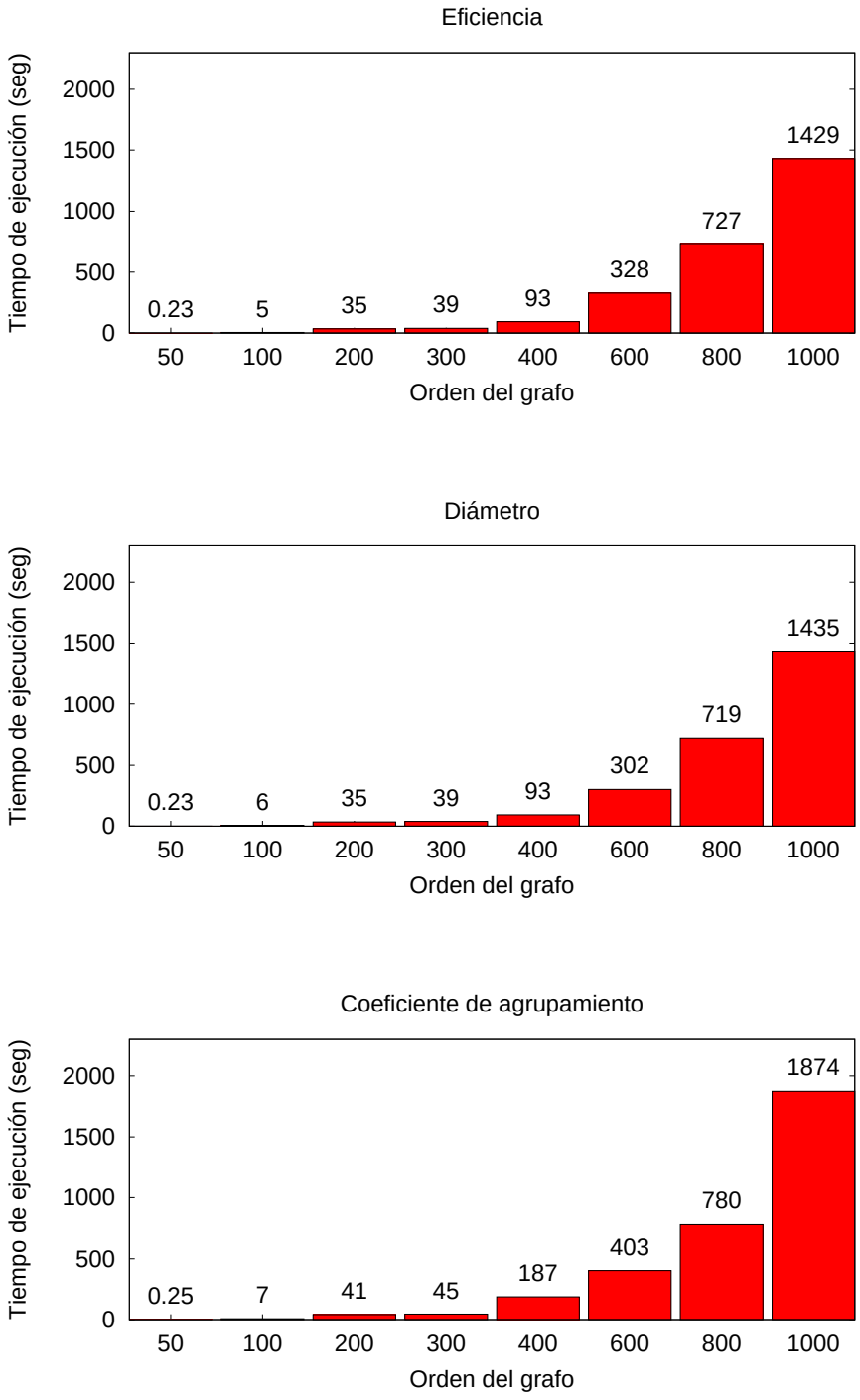


Figura 5.12: Comparación de los tiempos de ejecución al calcular las métricas eficiencia, diámetro y coeficiente de agrupamiento en grafos de 50, 100, 200, 300, 400, 600, 800 y 1000 vértices.

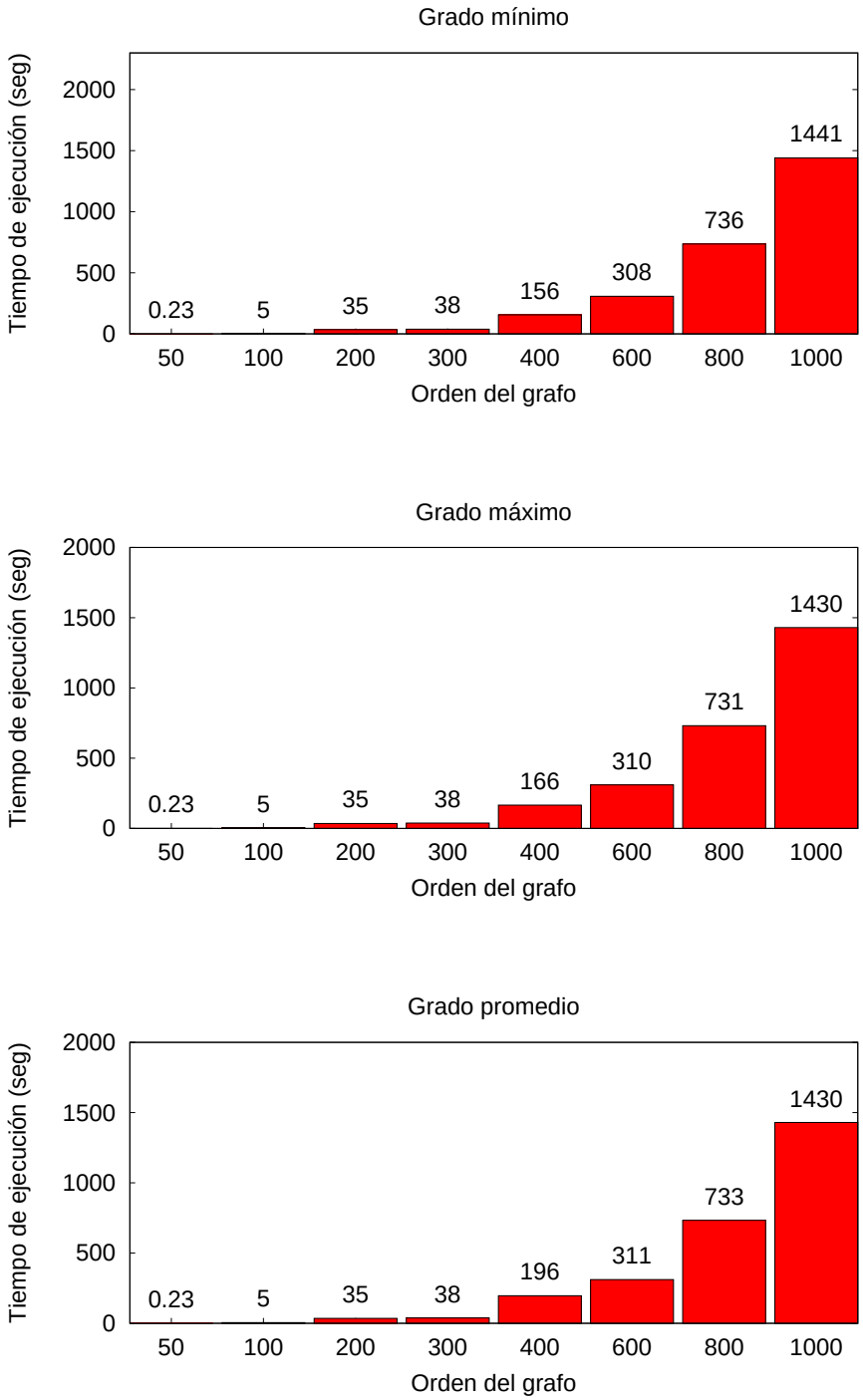


Figura 5.13: Comparación de los tiempos de ejecución al calcular las métricas grado mínimo, grado máximo y grado promedio en grafos de 50, 100, 200, 300, 400, 600, 800 y 1000 vértices.

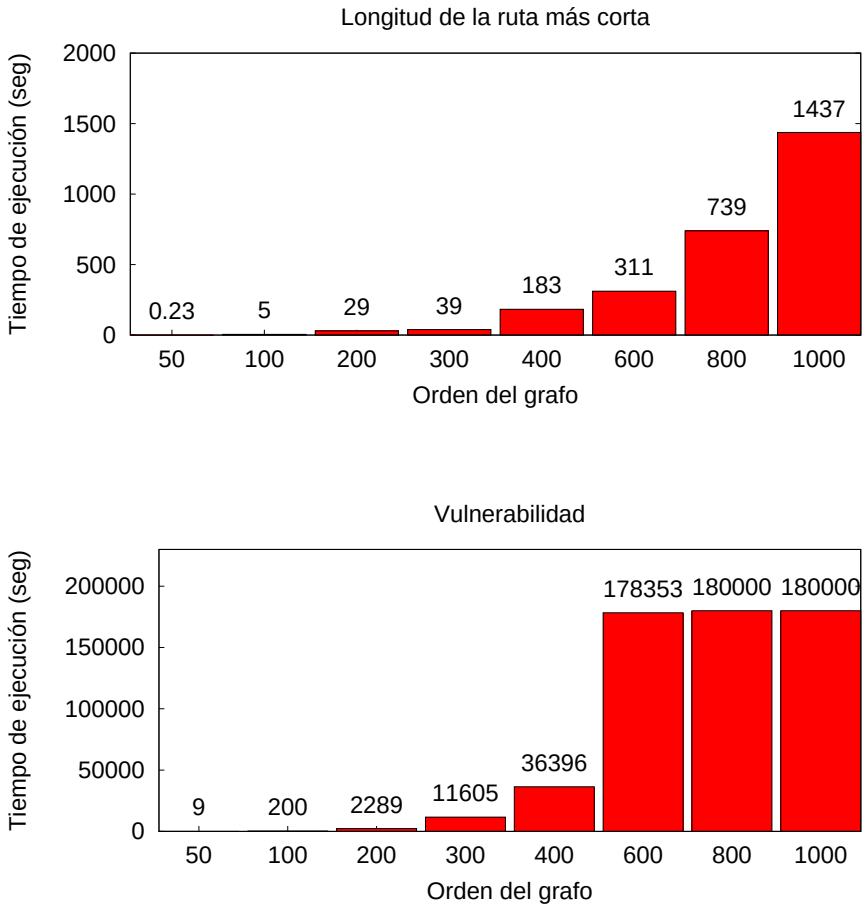


Figura 5.14: Comparación de los tiempos de ejecución al calcular las métricas longitud de la ruta más corta y vulnerabilidad en grafos de 50, 100, 200, 300, 400, 600, 800 y 1000 vértices.

5.8.1 Conclusiones

Al analizar los tiempos de ejecución mostrados en las Figuras 5.12, 5.13 y 5.14 se aprecia con claridad que el determinar la vulnerabilidad de un grafo es la propiedad más costosa computacionalmente. Con ayuda de *gprof* [26] se analizaron los cálculos que realiza la herramienta de software y se observó que sin importar el orden del grafo calcular la vulnerabilidad abarca alrededor del 96 % de los cálculos. Se recomienda al usuario calcular el tiempo computacional que requerirá la herramienta de software en caso de optimizar la vulnerabilidad ya que dependidendo del

tamaño del grafo la herramienta podría tardar un tiempo razonable para otorgar una solución. El otro 4 % del costo computacional lo conforma tanto el cálculo de otras propiedades, funciones de cruzamiento y mutación, la búsqueda y selección de los mejores individuos de cada iteración entre otras funciones.

En la Figura 5.15 se observa como aumenta el tiempo de ejecución al calcular todas las métricas de una red de 50, 100, 200, 300, 400, 600, 800 y 1000 vértices. Cabe recordar que el cálculo de la vulnerabilidad es detenido a los 180000 segundos, por lo cual se utilizó la cantidad de 180000 segundos como el tiempo mínimo que toma el cálculo de la vulnerabilidad en grafos de 800 y 1000 vértices.

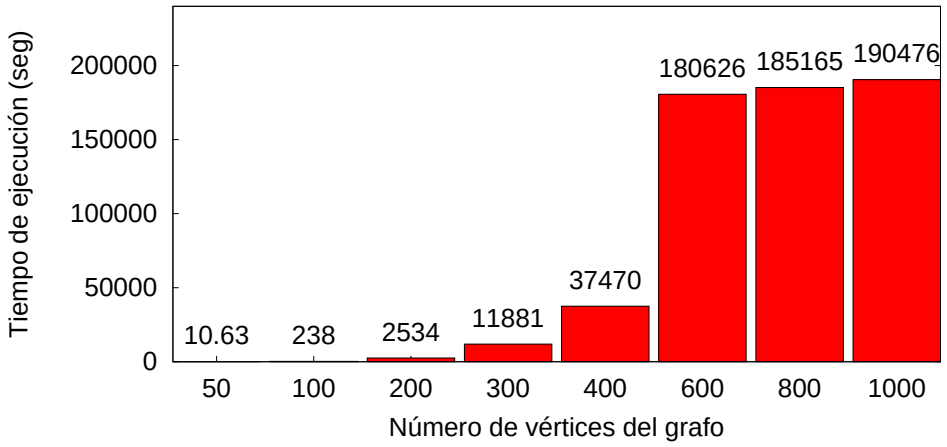


Figura 5.15: Crecimiento de los tiempos de ejecución (en segundos) al calcular las ocho métricas disponibles en la herramienta de software en redes de n vértices durante una iteración del algoritmo genético.

Capítulo 6

Conclusiones, aportaciones y trabajo futuro

En este capítulo se presentan las conclusiones del presente trabajo de investigación, las aportaciones científicas realizadas y las posibles líneas de investigación futura para ampliar esta área de conocimiento.

6.1 Conclusiones

En este trabajo de investigación se abordó el problema de optimizar la calidad de tres diferentes modelos de redes complejas: redes Aleatorias, Libre de escala y Mundo pequeño. Se construyó una herramienta de software que a través de un algoritmo genético optimiza una función que representa las propiedades estructurales definidas por el usuario.

El mejorar la calidad estructural en redes complejas no fue el único objetivo, ya que también se realizaron experimentos para encontrar los mejores parámetros del algoritmo genético que aseguran en redes Aleatorias, Libre de escala y Mundo pequeño encontrar el mejor conjunto de soluciones durante su ejecución. Los parámetros investigados fueron la cantidad de aristas mutadas por iteración, el número de iteraciones y el tamaño de la población inicial.

Se observó que para cinco de las seis clases de instancias, las mejores soluciones se obtuvieron bajo 1 mutación por iteración, solo en la instancia RAL/RMP las mejores soluciones se obtuvieron bajo 5 mutaciones por iteración. Con este experimento se demostró que no es necesario realizar un gran número de mutaciones a una red ya que las soluciones generadas no tienen la mayor calidad estructural y sus costos son elevados. Además al realizar una gran cantidad de mutaciones se pueden obtener soluciones cuya calidad estructural es menor que la calidad de la red original.

Se identificó que la aptitud de las redes optimizadas crece notablemente durante las primeras 500 iteraciones para redes Aleatorias y Mundo pequeño, mientras que para redes Libre de escala la aptitud aumenta notablemente hasta las primeras 1300 iteraciones. Después de éstas iteraciones la aptitud de las redes complejas comienza a mostrar una comportamiento asintótico que indica la convergencia de las soluciones.

Al resolver las instancias de prueba con diferentes tamaños en la población inicial, es decir, al utilizar una población inicial de 30, 50 y 100 individuos durante un mismo número de iteraciones, las soluciones con la mayor calidad estructural se obtuvieron con una población inicial de 100 individuos.

Durante los experimentos realizados en las instancias de prueba bajo una población inicial de 100 individuos y realizando 1 mutación por iteración, el menor porcentaje de mejora fue obtenido al optimizar la instancia RLE/RAL durante 1300 iteraciones donde se obtuvo una mejora del 72 %. El mayor porcentaje mejora fue obtenida al optimizar la instancia RAL/RLE durante 500 iteraciones con una mejora del 7342 %. Como es apreciable los porcentajes de mejora obtenidos varian notablemente y ésto es debido a la combinación de las propiedades estructurales que se tuvieron que optimizar.

Cuando la herramienta de software calcula la vulnerabilidad de la red esta propiedad consume el 96 % de los cálculos que realiza la herramienta de software. La vulnerabilidad es una importante propiedad estructural a través de la cual se cuantifica la eficiencia de la red a la hora de mandar información entre sus vértices,

pero es una propiedad costosa de calcular en términos de tiempo de ejecución.

Con base a los experimentos desarrollados se comprueba que la herramienta de software cumple con su objetivo, obtiene soluciones en un tiempo razonable y otorga soluciones robustas.

6.2 Aportaciones

La aportación principal de este trabajo de investigación es el desarrollo e implementación computacional de un método para encontrar soluciones de buena calidad para un problema de optimización de redes complejas, el cual no había sido tratado hasta donde se tiene conocimiento. Este método está basado en los algoritmos genéticos por lo cual se otorga un conjunto de soluciones pero el tomador de decisiones es libre de elegir la que más le convenga en base al porcentaje de mejora obtenido, el costo involucrado y la mejora en la calidad estructural por movimiento realizado.

Algunos resultados de este trabajo de tesis fueron publicados en:

- P. E. Cantú Cerda y S. E. Schaeffer. Análisis y optimización estructural de redes complejas. En Memorias del VI Congreso Internacional en Innovación y Desarrollo Tecnológico (CIINDET'08), artículo 332, págs. 1–7, Cuernavaca, México, Octubre 2008.

Fueron expuestos como cartel en:

- Primer Simposio sobre Investigación Científica e Innovación Tecnológica 2009 del Centro de Innovación, Investigación y Desarrollo en Ingeniería y Tecnología (CIIDIT), Apodaca, México, Mayo 2009.
- XVIII Escuela Nacional de Optimización y Análisis Numérico (ENOAN), Universidad Autónoma de Coahuila (UAC), Saltillo, México, Abril 2008.

6.3 Trabajo futuro

Dados los resultados obtenidos previamente con el desempeño del algoritmo genético propuesto, se tiene que es una herramienta de software prometedora para la optimización de redes complejas la cual bien vale la pena seguir mejorando. A continuación se dejan los siguientes puntos como áreas de oportunidad para trabajo futuro en esta línea de investigación.

- Cambiar la probabilidad de mutación y cruzamiento para conocer si el algoritmo genético crea poblaciones con mayor calidad que la actual.
- Modificar el algoritmo genético para poder añadir y eliminar vértices a la red, ya que en este trabajo solo se han realizado cambios a la topología en base al número de aristas de la cadena binaria.
- Existen muchas otras maneras de definir la función objetivo para evaluar la calidad de un grafo según su aplicación, así que sería bueno generar otras funciones objetivos para determinar si la forma de seleccionar a los mejores individuos influye posteriormente en la creación de individuos con mayor calidad estructural.
- Agregar más métricas a la herramienta de software para así poder mejorar más propiedades estructurales en redes complejas. Un buen ejemplo sería poder evalular el grado de intermediación de una red, la cual es una propiedad que nos da información sobre los elementos más importantes e identifica los principales cuellos de botella. Se debe tener claro que al agregar nuevas propiedades éstas deben calcularse bajo algoritmos o métodos eficientes que no conviertan el cálculo de tal propiedad en un problema difícil de resolver y comprometan negativamente la eficiencia de la herramienta de software.
- Buscar o crear alguna otra métrica alternativa con la cual se pueda determinar la vulnerabilidad de una red, pero sin un excesivo costo computacional.

- Estudiar que tanto afecta la densidad del grafo a la hora de optimizar las propiedades estructurales.

- Aplicar la herramienta de software en casos de estudio de un mayor tamaño ya que se ha visto que la verdadera efectividad de un procedimiento heurístico sólo puede comprobarse al estudiar problemas de alta dimensionalidad.

- Agregar a la herramienta una opción de visualización de las redes alternativas generadas, ya que actualmente solo se da como resultado una cadena binaria que representa las conexiones de la red.

- Realizar un análisis estadístico a los datos para demostrar su validez.

- Eficientar el algoritmo mediante el uso de estructuras de datos internas más apropiadas dada la densidad del grafo.

- Abordar el problema como un modelo multiobjetivo. En esta tesis se trabajó con una simplificación que consideró una función objetivo igual a la suma ponderada de los respectivos índices de las propiedades estructurales. Como es bien conocido, esta aproximación limita el espacio de soluciones y si se hace uso de la optimización multiobjetivo (que se dedica precisamente al estudio y desarrollo de métodos que puedan generar Frentes de Pareto) se obtendrá información valiosa sobre el compromiso de los diferentes criterios considerados.

- Paralelizar el algoritmo genético. En el contexto de los métodos heurísticos, el paralelismo no sólo significa resolver los problemas de forma más rápida, sino que además se obtienen modelos de búsqueda más eficientes: un algoritmo heurístico paralelo puede ser más efectivo que uno secuencial aún ejecutándose en un solo procesador.

- Probar el funcionamiento de la herramienta de software con datos reales al optimizar la estructura de la red de alguna empresa o del sector público.

Apéndice A

Modelos generadores de redes complejas

En este apéndice se mencionan los principales modelos de generación de redes complejas y los parámetros bajo los cuales se generaron las redes que sirvieron como instancias en este trabajo de tesis.

A.1 Modelos de generación de redes complejas

Los modelos de generación de redes son una herramienta importante que reproducen redes que comparten las características topológicas de las redes del mundo real. El estudio de estos modelos nace con la finalidad de analizar y comprender las funciones y fenómenos que se llevan a cabo en las redes complejas [54]. De entre los modelos existentes se distinguen dos tipos de modelos:

- **Modelos de generación sin crecimiento.** Estos modelos de caracterizan por dos aspectos importantes:
  1. El número de vértices es fijo.
  2. La agregación de aristas entre cualquier par de vértices se realiza con una probabilidad p.

Ejemplos de modelos de generación sin crecimiento son:

- **$\mathbf{G}_{n,p}$.** Propuesto en 1959 por Gilbert [29], el cual consiste que a un grafo con n vértices se le incluya cada una de las posibles aristas del grafo $E=\binom{M}{2}$ con una probabilidad igual a p.
- **$\mathbf{G}_{n,m}$.** Propuesto por Erdős y Rényi [25], el cual consiste que partiendo de un grafo con n vértices se le deben incluir con una probabilidad p, solo m aristas de todo el conjunto de posibles aristas.
- **Watts-Strogatz.** Propuesto en 1998 por Watts y Strogatz [56], el cual es un simple procedimiento para redes aleatorias que produce redes que tienen propiedades identificadas en las redes naturales del mundo real. El modelo se inicia con un grafo en forma de anillo $C_{n,k}$ de n vértices en donde cada vértice está conectado a $2k$ vecinos más cercanos (estando k conectado en dirección a las manecillas del reloj). Así Watts y Strogatz introdujeron aleatoriedad al grafo inicial seleccionando un vértice v y una arista (v, w) que conectaron a el siguiente vértice w del anillo. Con una probabilidad p, la arista (v, w) es redireccionada al sustituir w con un vértice aleatorio. Esto se repite para cada vertice del anillo. Así en la segunda ronda de redireccionamientos ahora las aristas conectan a sus segundos vecinos más cercanos en el anillo, hasta completar un total de k redireccionamientos. Este método en ocasiones originaba grafos desconexos por lo cual Newman [46] modificó el modelo Watts-Strogatz para evitar que se siguieran generando.

- **Modelos de generación basados en crecimiento.** Estos tipos de modelos se caracterizan por los siguientes aspectos:

  1. La construcción del grafo comienza con un número fijo de vértices y en cada paso se añaden x vértices.
  2. La agregación de aristas entre cualquier par de vértices se realiza con una probabilidad p.

Ejemplos de modelos de generación basados en crecimiento son:

- **Libre de escala.** El modelo Barabási-Albert [8] fue introducido en 1999 y está basado en el crecimiento y el enlace preferencial. Fue el primer modelo que reprodujo redes con una distribución del grado *Libre de escala*. Este modelo considera dos elementos:

  1. *Crecimiento*. La construcción del grafo comienza con un número pequeño de vértices m_0 y a cada paso t se añade un nuevo vértice i con $m \leq m_0$ aristas que enlazan a i con m diferentes vértices ya existentes en el grafo.
  2. *Enlace preferencial*. Cuando se incluye un nuevo vértice i al grafo, se deben seleccionar los vértices que seran vecinos de i. Para elegir los vecinos del vértice i se asume que la probabilidad Π de que i sea conectado a un vértice ya existente j depende del grado k_j del vértice j, tal que:
  $$\Pi(k_j) = \frac{k_j}{\sum_{v \in V} k_v}. \tag{A.1}$$
  Después de t lapsos de tiempo, el procedimiento resulta en una red con $n = t + m_0$ vértices con mt aristas.

Para mas información sobre los modelos de generación de redes complejas el lector puede consultar el texto de Virtanen [55].

A.2 Parámetros de las instancias

Los parámetros bajo los cuales se crearon las instancias de las redes Aleatorias, Libre de escala y Mundo pequeño utilizando los generadores de modelos de redes complejas proporcionados por Virtanen [55] se describen a continuación.

- **Redes Aleatorias, RAL.** Para la creación de las redes Aleatorias se utilizó el modelo de Erdős y Rényi. Se construyeron 5 redes *Aleatorias* de 50 vértices y alrededor de 222 aristas. La probabilidad utilizada para agregar una arista aleatoriamente al grafo fue de 0.19.

- **Redes Libre de escala, RLE.** Para la creación de las redes Libre de escala se utilizó el modelo de Barabási y Albert. Se construyeron 5 redes *Libre de escala* de 50 vértices y alrededor de 231 aristas. Los parámetros utilizados son un grafo inicial de 5 vértices y cada vértice i que se añadía al grafo se conectaba con los demás vértices a través de 5 aristas. La probabilidad utilizada para conectar al nuevo vértice i con otro vértice j fue de 0.3.

- **Redes Mundo pequeño, RMP.** Para la creación de las redes Mundo pequeño se utilizó el modelo de Watts y Strogatz. Se construyeron 5 redes *Mundo pequeño* de 50 vértices con alrededor de 186 aristas. Partiendo de un grafo en forma de anillo cada vértice se conectó a sus $3k$ vecinos más cercanos. Después se seleccionaba un vértice al azar y su arista era direccionada con una probabilidad igual a 0.3.

Se eligieron tales valores de los parámetros para los tres modelos de redes complejas porque se encontró que con estos parámetros se obtenían redes del mismo orden y tamaño. La importancia de la similitud en las instancias radica en que si los experimentos se realizaban bajos los mismos parámetros iniciales entonces se obtendrían resultados comparables.

Apéndice B

Funcionamiento del algoritmo genético

B.1 Generalidades

El algoritmo genético funciona bajo ciertos criterios que se mencionan a continuación.

- **Igualdad en cruzamientos y mutaciones.** El algoritmo genético proporciona a cada individuo la misma probabilidad para que su información sea cruzada o mutada. La probabilidad utilizada en este trabajo fue de 0.5 pero puede ser modificada por el usuario al inicio de la ejecución de la herramienta de software.

- **Agregar una arista tiene un costo igual a uno.** Durante los procesos referentes a las mutaciones y cruzamientos de individuos, si el algoritmo genético agrega una nueva arista dentro de las conexiones del grafo, el costo de la arista se considera igual a uno. Este costo es de uno porque se necesita un costo diferente de cero para poder calcular la *eficiencia* de una red, es decir, para calcularla se necesitan calcular las rutas más cortas y si algunas de estas valen cero entonces no se puede determinar la *eficiencia* de la red, porque ésta se define como el inverso de las rutas más cortas.

B.2 Estrategia computacional

Con el fin de reducir procesos computacionales y ahorrar tiempo de cómputo, el algoritmo genético evita realizar ciertas funciones ó cálculos en los siguientes casos:

- Cuando se determina que una red es no conexa. El eliminar en un principio individuos no conexos ahorra valioso esfuerzo computacional al no calcular las propiedades, especialmente en redes complejas que se caracterizan por la gran cantidad de elementos.
- En la selección de los individuos con mejores aptitudes. Al término de cada iteración del algoritmo genético se realiza una comparación entre los individuos generados (IG) y los individuos que pertenecen al conjunto de mejores soluciones (IMS). Así se busca al individuo con mayor calidad de los IG y se busca al individuo con menor calidad de los IMS. Si el individuo seleccionado de la IG no supera la calidad del individuo de los IMS entonces el algoritmo genético deja de realizar la búsqueda en los IG sin importar queden cientos de individuos aún por comparar.

B.3 Normalización de métricas

Para la obtención de mejores resultados se decidió normalizar cada uno de los valores de las propiedades obtenidas con la herramienta de software. El objetivo de normalizarlas es para obtener todos los valores en el rango $\{0, 1\}$ y así poder compararlas fácilmente entre ellas. Para la normalización se encontró los valores máximos y mínimos de cada métrica. Éstos se muestran en la Tabla B.1 (donde $V_{mín}$ y $V_{máx}$ denotan los valores mínimos y máximos, respectivamente) y se describen a continuación.

Tabla B.1: Valores máximos y mínimos de las métricas normalizadas.

Métrica	$V_{mín}$	$V_{máx}$
Eficiencia	0	1
Diámetro	1	$n-1$
Coeficiente de agrupamiento	0	1
Grado mínimo	1	$n-1$
Grado máximo	1	$n-1$
Grado promedio	1	$n-1$
Longitud de la ruta más corta característica	1	$n-1$
Vulnerabilidad	0	1

- **Eficiencia.** El $V_{mín}$ que puede tomar es cero, el cual se obtiene cuando no existe ningún camino de i a j en el grafo y por tanto la distancia se considera como infinita. Cuando se calcula la eficiencia con una distancia infinita la eficiencia se aproxima a cero. El $V_{máx}$ que puede tomar la eficiencia es 1, la cual se obtiene cuando el grafo es completo y por tanto existe una arista entre cualquier par de vértices $\{i, j\}$.

- **Diámetro.** El $V_{mín}$ que puede tomar es uno, el cual se obtiene cuando el grafo es completo y solo es necesario atravesar una arista para llegar de un vértice i a un vértice j. El $V_{máx}$ que puede tomar es $(n-1)$ el cual sucedería cuando se tuviese que atravesar todos los vértices del grafo hasta llegar al vértice deseado.

- **Coeficiente de agrupamiento.** El $V_{mín}$ que puede tomar es cero. El $V_{máx}$ que puede tomar el coeficiente de agrupamiento en un grafo es igual a uno, el cual por definición se encuentra en un grafo completo.

- **Grado mínimo.** El $V_{mín}$ que puede tomar es uno, ya que es el mínimo grado que debe tener cada vértice para pertenecer a la ruta que permite que el grafo sea conexo. El $V_{máx}$ que puede tomar es $(n-1)$ que se obtiene cuando el grafo es completo y cada uno de los vértices tiene una conexión con cada uno de los

demás vértices de la red[1].

- **Grado máximo.** El $V_{\text{mín}}$ que puede tomar es uno, ya que cualquier vértice debe tener al menos un vecino para formar la ruta que identifica al grafo como conexo. El $V_{\text{máx}}$ que puede tomar es $(n-1)$ que ocurre cuando cada vértice tiene una conexión con cada uno de los demás vértices de la red.

- **Grado promedio.** El $V_{\text{mín}}$ que puede tomar es uno, ya que es el mínimo valor que debe tener cada vértice para conformar la ruta con la cual se identifica al grafo como conexo. El $V_{\text{máx}}$ es $(n-1)$ que ocurre cuando cada vértice se conecta con cada uno de los demás vértices del grafo por ser un grafo completo.

- **Longitud de la ruta más corta característica.** El $V_{\text{mín}}$ que puede tomar es uno, el cual se obtiene cuando el grafo es completo y solo es necesario atravesar una arista para llegar de un vértice i a un vértice j. El $V_{\text{máx}}$ que puede tomar es $(n-1)$ el cual sucedería cuando se tuviese que atravesar todos los vértices del grafo hasta llegar al vértice deseado.

- **Vulnerabilidad.** La vulnerabilidad de un grafo está asociado a su eficiencia, por tanto el $V_{\text{mín}}$ que puede tomar es cero, que ocurre cuando la eficiencia de todos los vértices de la red es igual a uno. El $V_{\text{máx}}$ que puede tomar es 1 que ocurre cuando la eficiencia de todos los vértices de la red es igual a cero.

En conclusión la herramienta de software normaliza la mayoría de las propiedades respecto al grafo completo. Solo la eficiencia y la vulnerabilidad se normalizan con respecto a las distancias geodésicas. Independientemente de que se normalicen las métricas en base a el grafo completo o las distancias, los valores de las métricas quedan comprendidos en el rango $\{0, 1\}$.

[1]No se cuenta como una conexión las aristas de un vértice hacia sí mismo ya que no se permite tal comportamiento dentro de la herramienta de software.

B.4 Archivo de salida: cadenas binarias

El formato del archivo donde se muestran las cadenas binarias de todos los individuos o cromosomas seleccionados como los mejores durante la ejecución del algoritmo genético se muestra en la Tabla B.2.

Tabla B.2: Formato del archivo de salida donde se muestran las cadenas binarias del conjunto de mejores individuos.

```
#Red, Cadena Binaria
0 111000010111010 RO
1 111000010111010
2 010001100011000
3 000100010010101
4 001100010010101
5 101000101100110
6 011110101000000
7 010100010000101
8 111101000000010
9 101110000010010
10 010000110100110
```

Cada una de las cadenas binarias mostradas anteriormente representa las conexiones de una red alternativa creada con el algoritmo genético. Solo la cadena binaria mostrada en la posición cero corresponde a la red original (RO). En cuanto al nombre del archivo generado la herramienta por default le asigna el nombre *CB-filename.dat* donde *CB* es la abreviación de *cadena binaria* y *filename* hace referencia al nombre del archivo de entrada. El nombre de este archivo resultante puede ser cambiado por el usuario al inicio de la ejecución de la herramienta de software.

B.5 Archivo de salida: valores de las propiedades optimizadas

El segundo archivo generado por la herramienta de software muestra en forma de comentarios un resumen de los parámetros establecidos para la ejecución de la herramienta y se muestran en forma de columnas los valores de las propiedades calculadas/optimizadas para cada una de las redes alternativas. El formato de este archivo se aprecia en la Tabla B.3.

Tabla B.3: Formato del archivo de salida donde se muestran los valores de cada una de las propiedades optimizadas por la herramienta de software.

```
#Grafo de: [n: 50, m: 233, Cadena binaria: 1225]
#Mutaciones: 1, Pi: 10, Iteraciones: 30
#Ponderaciones a propiedades (0:No, -1:Min, 1:Max)
#Red: Efi:0.1, Diam:0.1, CA:-1, Gmin:-1, Gmax:1, Gprom:0.1,
#Lrmc:0.1, Vul:0.0, Costo:-1, Aptitud, Mejora(%)
0 0.032 0.1 0.069 0.082 0.306 0.190 0.0621 0.0 0.190 0.004 RO
1 0.043 0.1 0.070 0.082 0.347 0.190 0.0699 0.0 0.190 0.046 15%
2 0.046 0.1 0.072 0.082 0.367 0.193 0.0673 0.0 0.193 0.062 55%
3 0.041 0.1 0.069 0.082 0.347 0.189 0.0664 0.0 0.189 0.047 18%
4 0.049 0.1 0.072 0.082 0.367 0.194 0.0665 0.0 0.194 0.061 53%
5 0.043 0.1 0.069 0.082 0.347 0.190 0.0652 0.0 0.190 0.046 15%
6 0.044 0.1 0.070 0.062 0.347 0.191 0.0691 0.0 0.191 0.065 63%
7 0.043 0.1 0.069 0.082 0.347 0.190 0.0652 0.0 0.190 0.046 15%
8 0.041 0.1 0.069 0.082 0.347 0.190 0.0664 0.0 0.190 0.046 15%
9 0.050 0.1 0.073 0.082 0.367 0.194 0.0656 0.0 0.194 0.060 50%
10 0.037 0.1 0.069 0.061 0.327 0.189 0.0638 0.0 0.189 0.046 15%
#Tiempo(milisegundos):3020
```

Las columnas de los valores están ordenados de la siguiente manera: número de red, eficiencia, diámetro, coeficiente de agrupamiento, grado mínimo, grado máximo, grado promedio, longitud de la ruta más corta, vulnerabilidad, costo, aptitud y porcentaje de mejora. Al final de la línea correspondiente a la red 0, se muestran las siglas RO, lo cual significa que esos valores son los correspondientes a la red original. En la última línea del archivo se muestra el tiempo en milisegundos que tardó la herramienta de software calcular el conjunto de las mejores soluciones. La herramienta por default le asigna a este archivo el nombre *VP-filename.dat* donde *VP* es la abreviación de *valores de las propiedades* y *filename* hace referencia al nombre del archivo de entrada. El nombre de este archivo resultante puede ser cambiado por el usuario al inicio de la ejecución de la herramienta de software.

Bibliografía

[1] Ahuja, R. K., T. L. Magnanti y J. B. Orlin, *Network Flows: Theory, Algorithms, and Applications*, Prentice Hall, Englewood Cliffs, EUA, 1993.

[2] Albert, R. y A.-L. Barabási, «Statistical mechanics of complex networks», *Review of Modern Physics*, **74**(1), págs. 47–97, 2002.

[3] Albert, R., H. Jeong y A.-L. Barabási, «Error and attack tolerance of complex networks», *Nature*, **406**(6794), págs. 378–382, 2000.

[4] Amaral, L. A. N. y J. M. Ottino, «Complex systems and networks: Challenges and opportunities for chemical and biological engineers», *Chemical Engineering Science*, **59**(8–9), págs. 1653–1666, 2004.

[5] Bader, D., S. Kintali, K. Madduri y M. Mihail, «Approximating betweenness centrality», en A. Bonato y F. R. K. Chung (editores), *Proceedings of the 5th Workshop on Algorithms and Models for the Web-Graph (WAW2007)*, tomo 4863, San Diego, EUA, págs. 124–137, 2007.

[6] Ball, M. O., «Complexity of network reliability computations», *Networks*, **10**(2), págs. 153–165, 1980.

[7] Barabási, A.-L., «Network Workbench: A Large-Scale Network Analysis, Modeling and Visualization Toolkit for Biomedical, Social Science and Physics Research», `https://nwb.slis.indiana.edu/community/?n=Main.NWBTool`, Junio 2009.

[8] BARABÁSI, A.-L. y R. ALBERT, «Emergence of scaling in random networks», *Science*, **286**(5439), págs. 509–512, 1999.

[9] BATAGELJ, V. y A. MRVAR, «Pajek: Analysis and Visualization of Large Networks», `http://vlado.fmf.uni-lj.si/pub/networks/pajek/`, Junio 2009.

[10] BORNHOLDT, S. y H. G. SCHUSTER (editores), *Handbook of Graphs and Networks: From the Genome to the Internet*, Wiley, New York, EUA, 2003.

[11] BRODER, A. Z., S. R. KUMAR, F. MAGHOUL, P. RAGHAVAN, S. RAJAGOPALAN, R. STATA, A. TOMKINS y J. WIENER, «Graph structure in the Web», *Computer Networks*, **33**(1–6), págs. 309–320, 2000.

[12] BUTTS, C. T., «Network: A package for managing relational data in R», *Journal of Statistical Software*, **24**(2), págs. 1–36, 12 2007.

[13] CALLAWAY, D. S., M. E. J. NEWMAN, S. H. STROGATZ y D. J. WATTS, «Network robustness and fragility: Percolation on random graphs», *Physical Review Letters*, **85**(25), págs. 5468–5471, 2000.

[14] CANTÚ, P. E., «Herramienta para el análisis y optimización estructural de redes complejas», `http://it.ciidit.uanl.mx/~perla/herramienta.html`, Julio 2009.

[15] COELLO, C. A., G. B. LAMONT y D. A. VAN VELDHUIZEN, *Evolutionary Algorithms for Solving Multi-Objective Problems*, Springer-Verlag, Nueva York, EUA, 2006.

[16] COHEN, R., K. EREZ, D. BEN AVRAHAM y S. HAVLIN, «Resilience of the internet to random breakdowns», *Physical Review Letters*, **85**(21), págs. 4626–4628, 2000.

[17] COLBOURN, C. J. y J. H. DINITZ, *Handbook of Combinatorial Designs, Second Edition*, Chapman & Hall, New York, EUA, 2006.

[18] CSÁRDI, G. y T. NEPUSZ, «The igraph library for complex network research», http://igraph.sourceforge.net/, Junio 2009.

[19] DA COSTA, L., F. A. RODRIGUES, G. TRAVIESO y P. R. VILLAS BOAS, «Characterization of complex networks: A survey of measurements», *Advances in Physics*, **56**(1), págs. 167–242, 2007.

[20] DEB, K., «Multi-objective genetic algorithms: Problem difficulties and construction of test problems», *Evolutionary Computation*, **7**(3), págs. 205–230, 1999.

[21] DEB, K., «Multi-objective evolutionary algorithms: Introducing bias among Pareto-optimal solutions», en A. Ghosh y S. Tsutsui (editores), *Advances in Evolutionary Computing. Theory and Applications*, Springer, Berlín, Alemania, págs. 263–292, 2003.

[22] DEB, K., *Multi-Objective Optimization Using Evolutionary Algorithms*, Wiley, New York, EUA, 2004.

[23] DIESTEL, R., *Graph Theory*, *Graduate Texts in Mathematics*, tomo 173, Springer-Verlag, New York, EUA, 2000.

[24] DOROGOVTSEV, S. N. y J. F. F. MENDES, *Evolution of Networks: From Biological Nets to the Internet and World Wide Web*, Oxford University Press, Oxford, Reino Unido, 2003.

[25] ERDŐS, P. y A. RÉNYI, «On the evolution of random graphs», en *Selected Papers of Alfréd Rényi*, tomo 2, Akadémiai Kiadó, Budapest, Hungría, págs. 482–525, 1976.

[26] FENLASON, J. y R. STALLMAN, «The GNU gprofiler: GNU gprof», http://www.cs.utah.edu/dept/old/texinfo/as/gprof_toc.html, Junio 2009.

[27] GENTLEMAN, R. y R. IHAKA, «The R project for statistical computing», http://www.r-project.org/, Junio 2009.

[28] GHOSH, A. y S. DEHURI, «Evolutionary algorithms for multi-criterion optimization: A survey», *International Journal of Computing and Information Sciences*, **2**(1), págs. 38–57, 2005.

[29] GILBERT, E. N., «Random graphs», *Annals of Mathematical Statistics*, **30**(4), págs. 1141–1144, 1959.

[30] GOLDBERG, D. E., *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley, Boston, EUA, 1989.

[31] GRAY, A., *Modern Differential Geometry of Curves and Surfaces with Mathematica*, CRC Press, Boca Raton, EUA, 1996.

[32] HOLLAND, J., *Adaptation in Natural and Artificial Systems*, The MIT Press, Cambridge, EUA, 1992.

[33] HUISMAN, M. y M. A. VAN DUIJN, «Software for social network analysis», en P. J. Carrington, J. Scott y S. Wasserman (editores), *Models and Methods in Social Network Analysis*, Cambridge University Press, Cambridge, Reino Unido, págs. 270–316, 2005.

[34] KICINGER, R., T. ARCISZEWSKI y K. DE JONG, «Evolutionary computation and structural design : A survey of the state of the art», **83**(23-24), págs. 1943–1978, 2005.

[35] LASDON, L. S., *Optimization Theory for Large Systems*, Dover Publications, New York, EUA, 2002.

[36] LATORA, V. y M. MARCHIORI, «Efficient behavior of small-world networks», *Physical Review Letters*, **87**(19), págs. 1987 011–1987 014, 2001.

[37] LI, Q., G. P. STEVEN y Y. M. XIE, «Evolutionary structural optimization for connection topology design of multi-component systems», *Engineering Computations*, **18**(3/4), págs. 460–479, 2001.

[38] LTD CANONICAL, «Ubuntu: Linux for human beings», `http://www.ubuntu.com/`, Junio 2009.

[39] LYU, N. y K. SAITOU, «Topology optimization of multi-component structures via decomposition-based assembly synthesis», en *Proceedings of the ASME 2003 Design Engineering Technical Conferences and Computers and Information in Engineering Conference (DETC'2003)*, Chicago, EUA, págs. 1–13, 2003.

[40] MAINES, T., A. JAYARAMAN, J. BELSER, D. WADFORD, C. PAPPAS, H. ZENG, K. GUSTIN, M. PEARCE, K. VISWANATHAN, Z. SHRIVER, R. RAMAN, N. COX, R. SASISEKHARAN, J. KATZ y T. TUMPEY, «Transmission and pathogenesis of swine-origin 2009 A(H1N1) influenza viruses in ferrets and mice», *Science*, **325**(5939), págs. 484–487, 2009.

[41] MICHALEWICZ, Z., *Genetic Algorithms + Data Structures = Evolution Programs*, Springer-Verlag, Berlín, Alemania, 1996.

[42] MICHALEWICZ, Z. y D. B. FOGEL, *How to Solve It: Modern Heuristics*, segunda edición, Springer-Verlag, Berlín, Alemania, 2004.

[43] NEBRO, A. J., E. ALBA y F. LUNA, «Multiobjective optimization using grid computing», *Soft Computing*, **11**(6), págs. 531–540, 2007.

[44] NEWMAN, M. E. J., «Assortative mixing in networks», *Physical Review Letters*, **89**(20), págs. 208 701–208 704, 2002.

[45] NEWMAN, M. E. J., «The structure and function of complex networks», *SIAM Review*, **45**(2), págs. 167–256, 2003.

[46] NEWMAN, M. E. J., C. MOORE y D. J. WATTS, «Mean-field solution of the small-world network model», *Physical Review Letters*, **84**(14), págs. 3201–3204, 2000.

[47] PAPADIMITRIOU, C. H., *Computational Complexity*, Addison-Wesley, New York, EUA, 1994.

[48] PARONGAMA, S., D. SUBINAY, C. ARNAB, SREERAM P. A., MUKHERJEE G. y MANNA S. S., «Small-world properties of the Indian railway network», *Physical Review E*, **67**(3), págs. 036 106–036 110, 2003.

[49] SBALZARINI, I. F., S. MÜLLER y P. KOUMOUTSAKOS, «Multiobjective optimization using evolutionary algorithms», en *Center for Turbulence Research: Proceedings of the 2000 Summer Program*, NASA Ames/Stanford University, EUA, págs. 63–74, 2000.

[50] SCHAEFFER, S. E., *Algorithms for Nonuniform Networks*, Tesis Doctoral, Helsinki University of Technology, Espoo, Finlandia, Abril 2006.

[51] STALLMAN, R. y L. TORVALDS, «GNU/Linux», `http://www.linux.com/`, Junio 2009.

[52] STALLMAN, R. M., *Using the GNU Compiler Collection*, GNU Press, Boston, EUA, 2003.

[53] STROGATZ, S. H., «Exploring complex networks», *Nature*, **410**(6825), págs. 268–276, 2001.

[54] TURRUBIATES, T., *Clasificación de Redes Complejas Usando Funciones de Caracterización que Permitan Discriminar entre Redes Aleatorias, Power-Law y Exponenciales*, Tesis de Maestría, Instituto Tecnológico de Ciudad Madero, Ciudad Madero, México, 2007.

[55] VIRTANEN, S. E., «Properties of nonuniform random graph models», Research Report A77, Helsinki University of Technology, Laboratory for Theoretical Computer Science, Espoo, Finlandia, Mayo 2003.

[56] WATTS, D. J. y S. H. STROGATZ, «Collective dynamics of small world networks», *Nature*, **393**(6684), págs. 440–442, 1998.

[57] WELLMAN, B., «International Network for Social Network Analysis», `http://www.insna.org/index.html`, Junio 2009.

[58] WOLSEY, L. A., *Integer Programming*, Wiley-Interscience, New York, EUA, 1998.

[59] XU, J., *Topological Structure and Analysis of Interconnection Networks*, Kluwer Academic Publishers, Dordrecht, Holanda, 2001.

Ficha autobiográfica

Perla Elizabeth Cantú Cerda

Candidato para el grado de Maestro en Ciencias en Ingeniería de Sistemas
Universidad Autónoma de Nuevo León

Facultad de Ingeniería Mecánica y Eléctrica

Tesis:

Análisis y optimización estructural de redes complejas

Nací el 11 de diciembre de 1983 en Monterrey, Nuevo León, México. Soy hija del Sr. José Antonio Cantú Aguilar y la Sra. Victoria Cerda Ramírez. Soy egresada de la Facultad de Ingeniería Mecánica y Eléctrica de la UANL como Ingeniero Administrador de Sistemas (2000–2005). Estuve laborando en empresas locales por dos años hasta que inicié mis estudios de Maestría en Ciencias en Ingeniería de Sistemas en Enero del 2007 con el apoyo del Programa de Posgrado en Ingeniería de Sistemas y una beca de manutención otorgada por el CONACYT. Durante el 1er Simposio sobre innovación Tecnológica y Científica del CIIDIT mi cartel fue premiado como uno de los mejores dentro de las actividades del 1er Aniversario del CIIDIT.