Un algoritmo genético para la detección de comunidades en redes sociales mejorado mediante una técnica de clustering

David Nicolas Camelo García y Paola Carolina Suárez Suárez

Trabajo de Grado para optar el título de Ingeniero Industrial

Director

Henry Lamos Díaz

Ph. D en Física, Matemática

Codirector

David Esteban Puentes

MSc. Ingeniería Industrial

Universidad Industrial de Santander

Facultad de Ingenierías Fisicomecánicas

Escuela de Estudios Industriales y Empresariales

Bucaramanga

2021

Dedicatoria

A mi mamá, papá y mi hermana quienes confiaron en mí en más de una oportunidad para poder estudiar en esta grandiosa universidad y de la mano de Dios siempre han brindado apoyo a lo largo de mi vida gracias a sus grandes enseñanzas diarias y demás cosas inmateriales que otorgaron para culminar toda mi formación. Del mismo modo a nuestra mascota Katy quien ha sido foco de unión familiar y paciente ante la puerta de entrada de nuestro hogar.

A mis abuelos quienes quiero con todo el corazón, pues desde mi nacimiento me han aportado de experiencias personales que hoy agradezco porque han sido parte de mi ser diariamente, así como sus oraciones han sido fundamentales para mis logros.

A mi familia; tías, tíos, primos, primas de ambas partes que han sido un apoyo en cada etapa de mi vida y de mi formación, así como para mi tía Blanca quien durante el tiempo que se pudo me brindó trabajo en su local y con ello pude obtener sustento personal e independiente antes y durante gran parte de mi carrera.

A mis amigos; Camila, Melissa, Mateo, Karen, Juliana, Gabriela, Katherine y Miguel, quienes desde el principio de esta etapa aprendimos cosas únicas entre cada uno, compartimos grandiosos momentos, experiencias y esplendidas memorias, así como han sido gran apoyo incondicional en los peores y mejores momentos durante toda la carrera dejando su confianza en alto.

A las divisiones donde trabajé, mis exjefes el profesor Juan Camilo Lésmez y la profesional Alejandra, así como aquellos compañeros que cruzaron mi camino pues fueron granos de arena en este proceso. Los espacios culturales y de dispersión propuestos por la universidad pues proporcionaron momentos íntegros para mi desarrollo profesional.

Por último, pero no menos importante, a mi compañera de proyecto Paola, pues desde el principio compartimos experiencias graciosas que fortalecieron los vínculos de amistad y trabajo en equipo que durante el último año ayudaron en la superaron todos los obstáculos para cumplir con este logro final.

David Nicolas Camelo García

Dedicatoria

A mis padres, que siempre me han brindado lo mejor de sí, procurando darme las herramientas para que todo esto transite de ser un sueño a una realidad. Eternamente estaré agradecida por todos sus sacrificios y enseñanzas, ya que sin ellos no sería la mujer que soy hoy.

A mi abuela, que constantemente me tiene presente en sus oraciones y siempre ha deseado mi bienestar con sus constantes enseñanzas y sabiduría.

A María, mi segunda mamá, le agradezco su compañía, sus cuidados durante toda mi vida y cariño incondicional, además de su constante confianza en mis capacidades y este camino que culmina.

A mi hermano, por siempre impulsarme a ser mejor cada día y lograr todo lo que en algún momento soñé, además de haberme dado el mejor regalo y motivación de mi vida; mis dos lindos sobrinos Isabel y Pablo.

A Natha, por su amistad, su apoyo incondicional en los momentos más trascendentales de mi vida y darme los mejores recuerdos durante esta etapa.

A Felipe, que con todo su amor, comprensión y carisma me ha acompañado en los momentos más difíciles, además de su colaboración absoluta en este proyecto.

Finalmente, a David, que más que un compañero ha sido un amigo con el cual compartimos algunos proyectos personales y fue parte imprescindible para la culminación de este proceso ya que sin su perseverancia y tenacidad nada de esto sería posible.

Paola Carolina Suárez

Agradecimientos

Agradecemos a nuestros padres, por brindarnos la oportunidad de pertenecer a este prestigioso claustro, por todo su amor, sacrificio y confianza depositada en nosotros.

A nuestro director Henry Lamos por guiarnos a lo largo de este proyecto con su sabiduría, paciencia, experiencia y apoyo.

A David Puentes, nuestro codirector, que con su constante transferencia de conocimiento en los diferentes aspectos del proyecto y acompañamiento frecuente se convirtió en una parte fundamental para este proyecto.

A la Universidad Industrial de Santander y Escuela de Estudios Industriales y Empresariales quienes proporcionaron a los mejores mentores para la adquisición de conocimiento y formación de profesionales íntegros.

A lo eventos culturales y espacios esparcimiento realizados dentro y fuera del campus, pues complementaron en gran medida con áreas transversales que cualquier profesional debería contemplar durante su formación.

Contenido

Lista de Tablas

Lista de Figuras

Lista de Apéndices

(Ver apéndices adjuntos en la carpeta)

Resumen

Título: Un algoritmo genético para la detección de comunidades en redes sociales mejorado mediante una técnica de clustering.[*]

Autores: David Nicolas Camelo García, Paola Carolina Suárez[**]

Palabras clave: Algoritmo genético, detección de comunidades, coeficiente de clustering.

Descripción:

Alrededor del mundo las academias son el centro de concentración y difusión de conocimiento más importante para la sociedad, mediante sus programas de formación profesional, se entrega constantemente a la sociedad motores de conocimiento de alta calidad, los cuales por medio de alianzas bien establecidas a través de la divulgación científica se logra dar solución en distintas formas a las diversas necesidades e inquietudes que abruman en la cotidianidad. Este tipo de sociedades, donde se conjuntan diferentes metodologías, disciplinas, y temas, se conocen como redes de colaboración y una de las temáticas más utilizadas para lograr entender el comportamiento y encontrar agrupaciones fuertes en estas, es el problema de detección de comunidades (CD).

El problema de detección de comunidades tiene gran reconocimiento en los últimos años, pues durante su aplicación otorga claridad sobre los patrones más representativos que pueden existir en las agrupaciones dentro de la red, el cual siendo ejecutado por medio de algoritmo genético y técnicas de clustering, se logran determinar estructuras comunitarias fuertemente cohesionadas por medio de una metodología evolutiva para hallar el mejor valor de calidad de conexión de la red.

En el presente trabajo de investigación se busca dar solución al problema de detección de comunidades (CD) por medio de un algoritmo genético basado en coeficiente de clustering (CC-GA) a una red de colaboración de la Universidad Industrial de Santander conformada por docentes que han dirigido y codirigido trabajos de grado al interior del campus en programas diferentes a los que ellos se vinculan originalmente. Una vez determinadas las comunidades detectadas por el algoritmo, se procederá a analizarlas y así precisar los patrones existentes en esta forma de colaboración. Con ello se podrá establecer la condición de colaboración interdisciplinaria de la red, así como identificar los docentes más participativos en estas modalidades, entre otras características representativas de la red.

[*] Trabajo de grado

[**] Facultad de Ingenierías Físico-Mecánicas. Escuela de Estudios Industriales y Empresariales. Director: Henry Lamos Díaz. Codirector: David Esteban Puentes Garzón

Abstract

Title: A genetic algorithm to community detection in social networks improved by clustering technique.[*]

Autors: David Nicolas Camelo García, Paola Carolina Suárez [**]

Keywords: Genetic algorithm, community detection, clustering coefficient.

Description:

Around the world, the academies are the most important concentration and diffusion center of knowledge for society. Through their professional training programs, they constantly provide to the society high quality knowledge engines, which through well-established alliances through scientific popularization, are able to provide solutions in different forms to the diverse needs and concerns that overwhelm in everyday life. This type of societies, where different methodologies, disciplines, and topics are combined, are known as collaborative networks and one of the most used topics to understand behavior and find strong groupings in these, is the community detection problem (CD).

The community detection problem has been recognized in the last years, because during its application it gives clarity about the most representative patterns that can exist in the bunches within the network, which being executed by means of genetic algorithm and clustering techniques, it is possible to determine strongly cohesive community structures by means of an evolutionary methodology to find the best quality value of connection of the network.

In this research work, it seeks to solve the community detection problem (CD) through a clustering coefficient based genetic algorithm (CC-GA) to a collaborative network of the Universidad Industrial de Santander formed by teachers who have directed and co-directed work on campus in different programs to which they are originally linked. Once determined the communities detected by the algorithm, it will proceed to analyze them and thus specify the existing patterns in this way of collaboration. This will establish the condition of interdisciplinary collaboration of the network, as well as identify the most participatory teachers in these modalities, among other representative characteristics of the network.

---

[*] Bachelor's Thesis

[**] Physical-mechanical Engineering Faculty. School of Industrial and Business Studies. Supervisor: Henry Lamos Díaz. Co-supervisor: David Esteban Puentes Garzón

Introducción

Actualmente la Universidad Industrial de Santander-UIS se encuentra en el top 7 en la clasificación de las mejores IES colombianas según indicadores de investigación del Ranking U-Sapiens 2020-1[2] como reflejo de la importancia de la investigación en la universidad y su aporte hacía la sociedad.

En ese sentido, durante años se ha tratado de entender los comportamientos científicos y de investigación para una difusión de conocimiento íntegro ante la comunidad universitaria y la comunidad en general. A través del tiempo el estudio de redes sociales (SN por sus siglas en inglés, *Social Networks*) ha sido útil para describir, distinguir e interpretar las conductas de estas; las redes de colaboración se han enfocado en hallar cómo se comportan las comunidades en líneas disciplinarias, metodologías de investigación, temáticas estudiadas dentro de una disciplina, entre otros.

Si bien, este tipo de redes se han estudiado desde los años 80, de acuerdo con Newman, (2001) en su trabajo "The structure of scientific collaboration networks" afirma que anterior a tal fecha no se habían realizado reconstrucciones detalladas de las mismas y a partir de allí su estudio retrata estructuras puntuales y demarcadas sobre las interrelaciones entre científicos, artículos, colaboraciones y subgrupos de trabajo en campos especializados.

Las conexiones interdisciplinarias permiten tener mayor perspectiva del conocimiento, técnicas, aprendizaje en conjunto y visión integral de las problemáticas, a su vez, este tipo de estudios dan paso a aplicaciones como "la red basada en citas bibliográficas" (Wang et al., 2016),

[2] SRG, S. R. G., & CEO., C. R. P. (n.d.). Ranking de las mejores universidades de Colombia 2020. *Ranking U-Sapiens 2020-1*. Retrieved from https://www.srg.com.co

que pueden optimizar los resultados de búsqueda en plataformas académicas, con el fin de que se recomiende literatura más valiosa sobre el tema requerido.

De acuerdo con ello, el presente trabajo pretende destacar aquellas áreas y tendencias de mayor impacto mediante el estudio de patrones de colaboración científica o colaboración de áreas de conocimiento entre proyectos realizados con base en el problema de Detección de Comunidades-CD.

En el presente proyecto se aborda el problema de detección de comunidades (CD), haciendo uso de un algoritmo genético basado en coeficiente de clustering (CC-GA) a una red de colaboración conformada por docentes de los distintos grupos de investigación de la Universidad Industrial de Santander mediante la dirección y codirección de trabajos de grado interdisciplinarios.

Tabla 1

Cumplimiento de objetivos del proyecto

Objetivo	Cumplimiento
Realizar revisión de literatura de métodos de solución al problema de detección de comunidades en redes sociales.	Capítulo 2
Implementar el algoritmo genético mejorado en el lenguaje Python para el problema de detección de comunidades en redes sociales.	Capítulos 5 y 6
Validar el algoritmo desarrollado usando instancias del benchmarking.	Capítulo 5
Aplicar el algoritmo genético mejorado para una red social de la Universidad Industrial de Santander (UIS)	Capítulo 6

1. Generalidades del Problema

1.1.Planteamiento del problema

Una empresa, una sociedad o una red social se arraiga en las relaciones entre empleados e individuos (Easley, 2010); las redes sociales se encuentran involucradas en los diferentes campos de investigación, como la sociología, las matemáticas, la antropología, entre otras. El mundo actual evoluciona a gran velocidad y con ello los datos que éste posee, por lo que se hacen necesarios modelos y algoritmos apropiados para trabajar tal magnitud de datos, con el objetivo de lograr una cuantificación de las características propias de una comunidad, y así extraer sus principales cualidades para su posterior análisis.

La Universidad Industrial de Santander con su generación continua de información y datos como la formación de profesionales integrales mediante la difusión de conocimiento, construye procesos colaborativos por medio de la creación de proyectos interdisciplinarios, otorgándoles la profundización de competencias adquiridas en su etapa académica; se convierte en un objetivo clave de estudio para la detección de comunidades pues "es más probable que las personas comparta sus ideas con gente de la misma área, la misma universidad, o al menos que hablen el mismo idioma" (Wang, Li, Zhang, & Lu, 2016, p.2).

Es por esto, de acuerdo con Khan et al. citado en (Feng et al., 2019) se considera:

> "En las redes de colaboración, la información refleja las pautas de colaboración y las relaciones sociales académicas entre los estudiosos de diferentes disciplinas. La detección de comunidades se considera tradicionalmente como una especie de partición de gráficos para descubrir exhaustivos y desarticulados grupos de nodos en una red determinada" (p.3).

A través de la integración de tales conceptos de estudio de redes, se propone dar solución al problema de CD en una red de colaboradores de la UIS por medio de un algoritmo genético

mejorado con una técnica de clustering, en su desempeño, con respecto a la “eficacia, rapidez y simplificación de resultados”(Witten et al., 2011).

1.2.Objetivos

1.2.1. Objetivo general

Diseñar un modelo de detección de comunidades en redes sociales mediante un algoritmo genético mejorado por medio de una técnica de clustering

1.2.2. Objetivos específicos

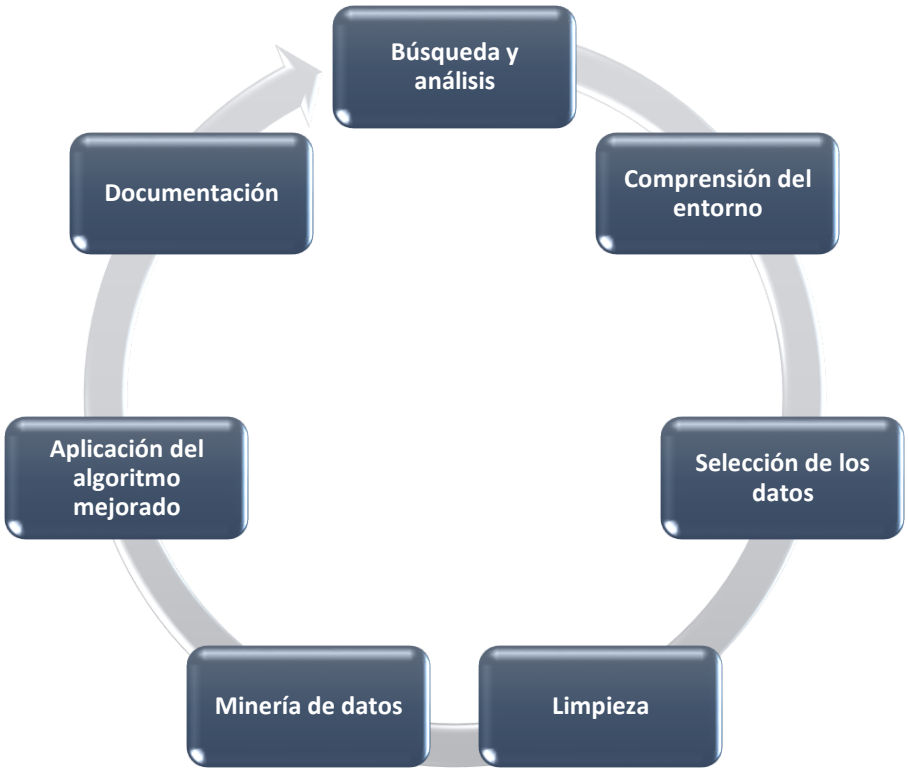
- Realizar revisión de literatura de métodos de solución al problema de detección de comunidades en redes sociales.
- Implementar el algoritmo genético mejorado en el lenguaje Python para el problema de detección de comunidades en redes sociales.
- Validar el algoritmo desarrollado usando instancias del benchmarking.
- Aplicar el algoritmo genético mejorado para una red social de la Universidad Industrial de Santander (UIS)

1.3. Metodología

El desarrollo del presente trabajo de investigación se llevará a cabo en 7 etapas, como se observa en la Figura 1.

Figura 1

Metodología del proyecto. Realizado por los autores del documento



Las etapas se encuentran distribuidas de la siguiente manera:

1.3.1. Etapa 1. Búsqueda y análisis de la literatura

Con el documento base denominado “CC-GA: A clustering coefficient based genetic algorithm for detecting communities in social networks”, se logra el establecimiento de las palabras claves para la realización de ecuaciones de búsqueda especializadas en las bases de datos Scopus y Web Of Science.

Una vez determinadas tales ecuaciones, es posible la obtención, clasificación y depuración de documentos útiles para la siguiente etapa.

1.3.2. Etapa 2. Comprensión del entorno/escenario

Posterior al hallazgo de los documentos adecuados, se realiza una exhaustiva lectura y comprensión de cada uno de éstos para así abordar de manera precisa el problema CD, el algoritmo genético y la técnica de clustering; detectar características relevantes y posibles soluciones propuestas por los autores a lo largo del tiempo que se puedan adaptar a las particularidades de la presente investigación. También, esta etapa es de gran importancia ya que con ella se definen las limitaciones, así como los conocimientos complementarios para el desarrollo del proyecto, tal como las bases del lenguaje Python.

1.3.4. Etapa 3. Selección de los datos

Mediante la plataforma de ColCiencias, "Currículum Vitae para Latinoamérica y el Caribe-CvLAC" se logra la obtención de las hojas de vida de docentes vinculados a los grupos de investigación de las facultades, donde posteriormente se extraen los proyectos que han estado asociados con otras facultades, dado que estas conexiones hacen parte del grupo de interés para la definición y análisis de la red.

1.3.5. Etapa 4. Limpieza

Una vez realizada la etapa anterior se procede a unificar el conjunto de datos que permitirá la depuración de datos, eliminación de información y atributos no útiles, asegurando la integridad y calidad de esta, para dar un formato único de presentación y utilización. Adicional a ello, para una eficiente lectura por parte del lenguaje se otorga una identificación especial a estos datos para su presentación y utilización.

1.3.6. Etapa 5. Minería de Datos

Haciendo uso de la información ya decantada, se procede a la selección de la tarea apropiada de minería de datos, de acuerdo con el objetivo de descripción que consta de la observación de los datos obtenidos, asimismo, el algoritmo genético modificado mediante una técnica de clustering se selecciona como el método utilizar para la solución del problema CD en la universidad.

Enfocado en los códigos utilizados para el CD vistos en la literatura base, se formula el algoritmo en lenguaje Python diseñado para las características de red propuestas.

1.3.7. Etapa 6. Aplicación del algoritmo mejorado

Subsiguiente a la realización y análisis de la etapa anterior se procede a aplicar el algoritmo a la base de datos decantada para la descripción y obtención de la red social en la universidad. Para ellos se realiza de antemano una validación del mismo por medio de benchmarking entre otras redes de la literatura que aportarán la estimación de parámetros eficientes para su implementación en la red universitaria.

1.3.8. Etapa 7. Documentación

Acto seguido a la realización de las anteriores etapas se procede a revisión de resultados y hallazgos de conclusiones plasmándolos en el libro final de presentación de tesis.

2. Revisión de la literatura.

Tomando como base para el desarrollo de la presente investigación el artículo “CC-GA: A clustering coefficient based genetic algorithm for detecting communities in social networks” realizado por Said, Abbasi, Maqbool, Daud, & Aljohani, se destacan los elementos y áreas de aplicación al problema propuesto por los autores con el fin de determinar las ecuaciones de

búsquedas idóneas para el mismo. La búsqueda de los términos se realizó usando dos de bases de datos de la UIS: Scopus y Web Of Science (WOS), en dos fases distintas.

Fase 1: La ecuación de búsqueda de esta fase tiene como fin la obtención de antecedentes sobre el desarrollo en las distintas temáticas existentes relacionadas con las palabras claves definidas, así como la evolución de los algoritmos que buscan la solución al problema de *Community Detection-CD*. En la Tabla 2 se aprecian las ecuaciones de búsqueda iniciales utilizadas en esta fase.

Tabla 2

Ecuaciones de búsqueda Fase 1

Base de Datos	**Ecuación de Búsqueda**
Web of Science	TEMA: ((communit?$ detection OR communit?$ identification) AND (genetic algorithm) IN (social networks OR complex networks))
Scopus	TITLE-ABS-KEY ((community AND detection) AND (genetic AND algorithm) AND (social AND network OR complex AND network))

Los resultados de las anteriores ecuaciones fueron: WOS 146, Scopus 128.

Fase 2: En esta fase, como se aprecia en la Tabla 3, se realizaron modificaciones a la ecuación original, agregando los términos “friend, “lunch”, “collaboration” y “knowledge” que dirigen el enfoque de los artículos hacía dichos asuntos. En Scopus se elimina el término “complex network” al igual que no se contemplan “friend” y “lunch” para evitar que dicha herramienta vuelva a expandir resultado hacía áreas no relacionadas en la búsqueda.

Tabla 3

Ecuaciones de búsqueda Fase 2

Base de Datos	Ecuación de Búsqueda
Web of Science	TEMA: ((communit?$ detection OR communit?$ discover?$*) AND (genetic algorithm) AND (social network$ OR complex network$) AND (collaboration OR friend OR lunch OR knowledge))
Scopus	TITLE-ABS-KEY ((community AND detection) AND (genetic AND algorithm) AND (social AND network OR complex AND network) AND (collaboration OR knowledge))

Mediante los anteriores arreglos se obtuvieron los siguientes resultados: WOS 21 y Scopus 26

Nota: Si bien en el artículo base mencionado se contempla como término principal el "cluster", en las ecuaciones de búsqueda no se utiliza en conjunto con los demás términos mencionados en la fase 2, puesto que la combinación de los 3 factores sesga los resultados a una mínima cantidad de resultados, impidiendo un análisis completo de la literatura. Por ello se toman en funciones separadas para facilitar la similitud de resultados que sean de utilidad.

2.1. Análisis Bibliométrico

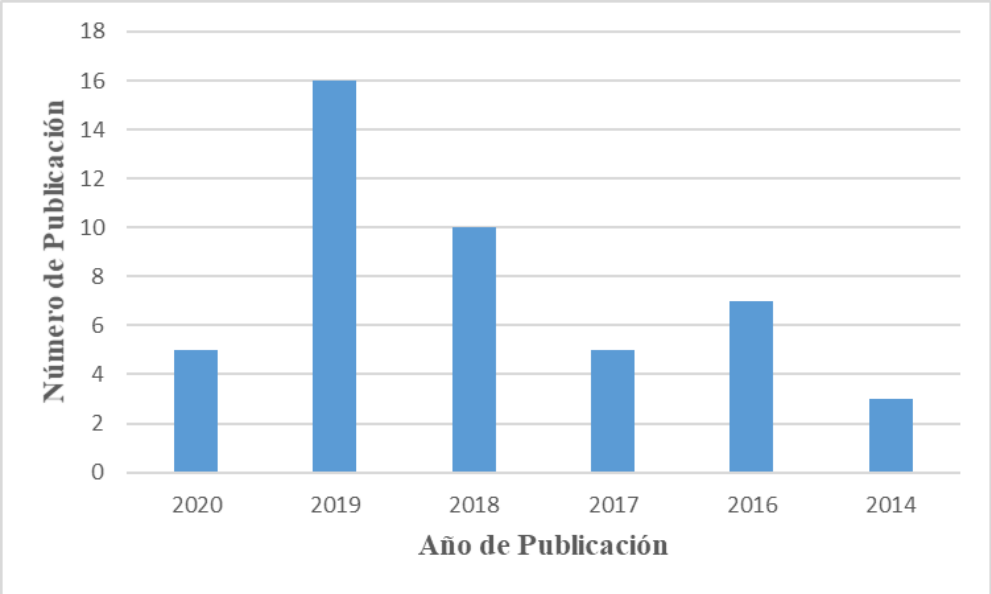
Finalizado las fases de búsqueda, se logran destacar 46 artículos de mayor relación. Dicha clasificación se realiza por medio de WOS a través de las ecuaciones utilizadas, clasificando artículos en: relevancia con en el tema, cantidad de citaciones, área de investigación y año de publicación. De acuerdo con estos artículos se puede destacar lo siguiente:

2.1.1. Año de Publicación.

Con el fin de llevar una cronología concreta de las publicaciones, se analizó la cantidad de publicaciones en cada uno de los años como se observa a continuación:

Figura 2

Publicaciones por año. Tomado del análisis de WOS



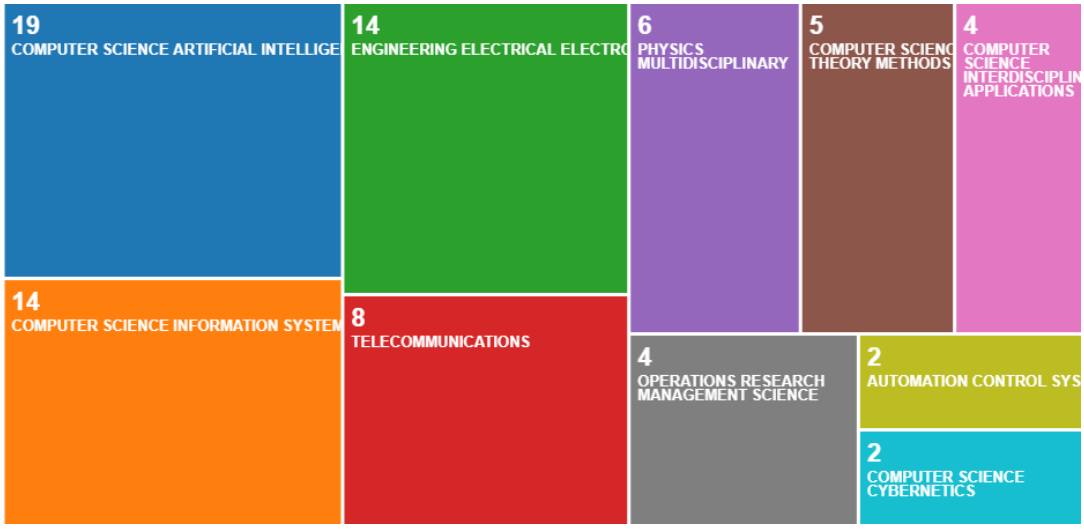
En la Figura 2 se puede observar que los temas de relevancia están más concentrados a partir del año 2014 hasta la actualidad, cada año presenta diferente número de publicaciones; 3 en 2014, 7 en 2016, 5 en 2017 y 10 en 2018. Del mismo modo se aprecia el interés de investigación creciente en el área, puesto que para el 2019 se publicaron 16 artículos en total, finalmente en lo que va corrido del año 2020 van 5 publicaciones al respecto.

2.1.2. Área de Investigación

Enfocándonos ahora en las temáticas y áreas de investigación en tales documentos con la finalidad de conocer las diversas aplicaciones se encuentra lo siguiente:

Figura 3

Área de aplicación. Tomado del análisis de WOS



En la Figura 3 se aprecia que gran parte de publicaciones se enfocan en la ciencia de computación mediante 5 áreas complementarias a ésta; 19 artículos de inteligencia artificial, 14 sistemas de información, 5 teoría y métodos de ciencia de la computación, 4 de aplicaciones interdisciplinarias y 2 de ciencia de la computación cibernética. En otras áreas de importancia se puede enfatizar: 14 publicaciones en Ingeniería Eléctrica y Electrónica, 8 en Telecomunicaciones, 6 en Física Multidisciplinaria, 4 en Investigación de operaciones y 2 en sistemas de controles automáticos.

Con estas últimas observaciones se resalta el interés por estudiar e investigar sobre estos conceptos, iniciando mediante la conceptualización y análisis de la aplicación de los distintos algoritmos empleados en el CD, la comparación de resultados alcanzados entre ellos y el por qué, de la evolución en la estructura de estos, así como sus ventajas en la implementación de acuerdo con el tipo de red estudiada. A raíz de ello se destacan para efectos de esta investigación, los Algoritmos Genéticos y las técnicas especializadas de agrupamiento.

2.2. Análisis Preliminar

Entender y predecir el comportamiento de distintos grupos sociales es un tema de interés para la ciencia, los negocios, la política, entre otros. Sin embargo, no es una tarea sencilla de realizar por medio de técnicas cotidianas de fácil obtención en el entorno debido al gran dinamismo de dichas redes. Esto ha dado vía libre para el desarrollo de investigaciones que traten de llegar a una solución óptima con base en distintos objetivos de caracterización bajo el estudio de redes sociales y redes complejas.

A través de CD se logra la caracterización de particularidades de las redes objeto de estudio, ya que presenta diversas aplicaciones; el funcionamiento de sistemas en una organización (Yang et al., 2014), en grupos de decisión (Bedi & Sharma, 2016), influencia de las personas entre sus redes (Kaur et al., 2016), tipos de relaciones entre grupos (Girdhar & Bharadwaj, 2019), entre otras variedades de empleos y usos de acuerdo al análisis deseado por el investigador.

Dado lo anterior, la siguiente revisión de literatura presenta la evolución del uso de diversos algoritmos y técnicas utilizadas en diferentes aplicaciones del CD. Esta temática fue estudiada inicialmente (de acuerdo con las citaciones realizadas en la mayoría de los artículos y la cronología de los mismos), por Girvan & Newman, (2002) mediante la aplicación de detección de estructuras comunitarias, por medio del algoritmo de Girvan y Newman (GN) utilizando el método conocido de agrupamiento jerárquico, aplicado a distintas “redes ficticias”, comparando los resultados con otras técnicas tradicionales de menor complejidad. Estos autores demostraron que su algoritmo obtenía mejores respuestas en menor tiempo, gracias a que su algoritmo también comprende el concepto de Modularidad el cual es un aspecto que permite la evaluación de conexiones entre los nodos de la red, logrando la optimización de este y alcanzando mejores niveles de conexión intrared y entrered.

Este mismo método de agrupamiento jerárquico, fue usado por Jiang, Liu, & Wang, (2016) acoplándose al Problema del agente viajero (TSP) otorgándole a tal problema un enfoque complementario, demostrando que además de encontrar la ruta más corta de viaje y menos costosa, todos los individuos visitados y atendidos por el "vendedor" pueden caracterizarse por medio de comunidades que representan características similares. Sin embargo, de acuerdo con Arasteh & Alizadeh, (2019) esta técnica presenta algunos inconvenientes, debido que el proceso de eliminación de bordes va cambiando constantemente el valor de modularidad y agrupación inicial, lo cual afecta de manera considerable el valor óptimo del mismo al finalizar la iteración, encontrando posiblemente valores indeseables en el resultado final.

A raíz de ello, con intención de optimizar los resultados obtenidos, al problema de CD se han introducido distintos tipos de algoritmos, los cuales también dependen "en gran medida de la topología de la red, ya sea dinámica o estática." (Javed et al., 2018). Esto se debe a que cada algoritmo funciona de distinta forma en cada área de aplicación lo cual resume que, si en una red el algoritmo funciona bien, para otra red con características muy similares, la eficiencia de este puede no ser suficiente.

Dichos algoritmos se han separado en Algoritmos Evolutivos (EA) entre estos se presenta una extensión como lo son los Algoritmos Genéticos (GA). Attea, Hariz, & Abdulhalim, (2016); Folino & Pizzuti, (2014); Mishra, Hota, Kumar, & Nayak, (2019); Žalik & Žalik, (2018) aplicaron técnicas de EA para el problema de CD, implementando en varios operadores conocidos como operadores cruzados (*crossover),* los cuales funcionan "combinando dos individuos para producir una o dos descendencias. Por ello es un operador muy importante, ya que debe generar una descendencia con las mejores propiedades heredadas de sus padres, mientras que también debe mantener la diversidad de la población." (Žalik & Žalik, 2018, p.3). Cabe mencionar que los

autores previamente mencionados, aplicaron este algoritmo con base en la optimización multi-objetivo, teniendo como principio el concepto de Modularidad aplicado por Girvan & Newman, (2002) testeado en la "red de colaboradores de del Instituto Santa Fe de Nuevo México, constituida por un centro de investigación interdisciplinario", a través de los siguientes criterios: "Bajo número de valores entre clústeres y un alto número de valor intra cluster" (Mishra et al., 2019, p.9), donde el algoritmo determina una clasificación de la red en 2 tópicos; similitudes de tema de investigación y metodología aplicada, demostrando porqué en los departamentos interdisciplinarios se pueden encontrar conexiones frente a otros, e intuyendo un pronóstico sobre posibles futuros vínculos entre campos de conocimiento vagamente similares.

Por otro lado, los GA son algoritmos más utilizados para la optimización de problemas complejos, éstos se adaptan completamente a la dificultad requerida en la CD. "Se basan en mecanismo con los procesos evolutivo como la reproducción, selección natural entre otros, pueden determinar automáticamente el número de clústeres en una red, lo que los hace útiles para redes del mundo real". (Kaur et al., 2016). Este mismo autor realizó una comparación entre diversos GA aplicándolos a redes sociales; Algoritmo Genético Multiobjetivo (MOGA-net), Densidad de Modularidad y Modularidad Rápida, usando Información Mutua Normalizada (NMI), la cual calcula la calidad de las particiones realizadas a través de estos métodos, concluyendo para este caso que el mejor nivel de NMI encontrado entre dichos métodos es alcanzado por el método de Densidad de Modularidad.

Adicional a ello la gran variedad de GA permite la integración de diversas técnicas de agrupamiento mejoradas, entre ellas la más conocida y usada en la investigación es el *Community Coefficient-CC*; un concepto implementado desde 1998 para determinar y comprobar la existencia de una propiedad común de las redes sociales reales, Newman, (2001) ejemplifica tal característica

en redes de colaboración, donde los centros de conocimiento podrían formar comunidades científicas importantes y a su vez conjuntos de investigadores que trabajen en subcampos especializados particulares. Nascimento, (2014) utilizó dicha técnica con el fin de alcanzar una maximización adaptativa a redes ficticias, sin embargo, también concluye que con base en el algoritmo principal no es factible el uso de ellas sobre el análisis de redes o grafos con un tamaño reducido. Posteriormente Deng, Zhai, Lv, & Yin, (2017) implemento la CC en medio de redes superpuestas, lo que da paso a encontrar no solo mejores valores de agrupamiento entre nodos, sino también la diversidad de conexiones que pueden tener éstos de acuerdo a características que no son comunes para todos los individuos pertenecientes a un grupo, esta característica de comunidades superpuestas mediante CC, se explica mejor a través del CD en Multicapas estudiados por Kim, Lee, & Lim, (2016) en "Differential Flattening: A Novel Framework for Community Detection in Multi-Layer Graphs" y por Li, Xu, & Tang, (2018) en "Community detection for multi-layer social network based on local random walk" quienes determinan que cada capa representa una característica de una comunidad en especial. Por ejemplo: "Trabajo, Amigos, Facebook, Coautor y Comida" (Li et al., 2018), sin embargo, como es de inferir, un individuo puede pertenecer a más de una comunidad al mismo tiempo, por ello estos autores evalúan el CC local, donde determina agrupaciones en una sola capa del problema y el CC multicapa, tratando de conectar en grupos a individuos existentes en todas las capas el mismo tiempo sin tener que separarlos de su capa principal.

Si bien el CC se puede usar como técnicas de evaluación y optimización extras al GA, Said, Abbasi, Maqbool, Daud, & Aljohani, (2018) en su artículo "CC-GA: A clustering coefficient based genetic algorithm for detecting communities in social networks", como bien su nombre lo indica, integraron el CC a un GA (CC-GA), con el fin de que las iteraciones de mismo vayan evaluando

todas las condiciones para alcanzar no solamente identificación de comunidades sino encontrar también grupo de gran cohesión entre distintos nodos de la red, esto les permite evaluar redes más grandes y complejas con niveles de eficiencia más altos. Con base en los resultados obtenidos por tales autores el CC-GA alcanza mejores resultados en el análisis de redes sociales, donde en 6 de las 10 redes analizadas obtuvo un mayor valor de modularidad comparado contra otros algoritmos de GA; (TGA, LGA y MENSGA), en este mismo orden de ideas este algoritmo presenta la capacidad de crear conexiones con gran modularidad, alcanzando los valores máximos en menos cantidad de tiempo con respecto a los demás algoritmos.

En resumen, el CD es un problema que ha logrado el interés de investigación en distintas áreas de aplicación, el cual, si bien empezó a trabajarse desde 2002, se ha llevado mejoras a la solución de este problema por medio de gran diversidad de algoritmos desde el 2014, facilitando en gran medida las tareas del análisis de data. Se observa que entre los algoritmos de gran uso en tal problema están los GA, los cuales debido a su gran variedad de operadores logran ligarse completamente con las características propias de las redes sociales. Cabe destacar algunos aspectos importantes en la aplicación de la CC a dichos algoritmos, puesto que logran conseguir agrupaciones más concentradas durante la identificación de comunidades, con mejores valores de conexión en menor tiempo. Por ende, dado el gran uso de estos dos últimos casos mencionados, el presente trabajo llevará a cabo mediante la aplicación del CC a un GA para así fortalecer los conocimientos en las distintas áreas de los involucrados en los proyectos interdisciplinares, encontrando posibles focos de conocimiento asociado.

3. Marco teórico

3.1. Red social:

Las redes sociales son colecciones de sociedades interconectadas, representada computacionalmente por medio de grafos, estos son herramientas básicas para el correcto estudio de las mismas, según Hansen, Shneiderman, & Smith, (2011) "están compuestas por 2 bloques: nodos (agentes de conexión) y bordes (lazos de conexión)" (p.3). Tal estructura permite simplificar y moldear de forma eficiente la búsqueda local y global las tendencias de conexión entre los individuos pertenecientes a cada uno de los nodos de la red.

Nodos: Están destinados para la representación de las estructuras sociales concentradas del objeto de estudio, ya sean; grupos de trabajo, organizaciones, instituciones, entre otros.

Aristas, Bordes o enlaces: Representan las distintas formas en las que se vinculan entre si los nodos, las clases de relaciones o vínculos se adaptan según el objetivo principal de la investigación y pueden ser tipo proximidad, colaboración, entre otras.

3.2. Grafo:

La representación gráfica y matemáticas de las redes sociales con sus componentes específicas de nodos y enlaces, de la siguiente forma:

$$\boldsymbol{G = (V, E)} \quad \textit{(1)}$$

Donde V representa los nodos, que como se menciona anteriormente es el actor social del estudio y E representa el borde que conecta un par o más de los actores existentes en la red de acuerdo con sus características en común.

3.3. Red de Colaboración:

Las redes sociales reales presentan miles de características, entre ellas está las conocidas redes de colaboración, donde en este caso la intención es determinar conexiones e interacciones frecuentes entre uno o más autores de acuerdo a un tema, metodología aplicada en sus artículos como reflejo de la colaboración científica entre distintas áreas de investigación, en otros aspectos "además de la distancia entre los autores, hay muchas otras cantidades interesantes que deben medirse en las redes de colaboración, incluyendo el número de colaboradores de los científicos, el número de artículos que escriben y el grado de "agrupación"" (Newman, 2001, p.2).

3.4. Detección de Comunidades:

"Las comunidades en una red social pueden representar verdaderas agrupaciones sociales, tal vez por interés o antecedentes" (Girvan & Newman, 2002, p.1), cuales reflejan una fotografía que posteriormente pueden ser utilizadas con fines investigativos.

3.5. Modularidad:

La modularidad es una medida de aptitud en los problemas CD, que se encarga de medir la calidad que tienen las conexiones encontradas en cada iteración del algoritmo. Newman, (2006) define la modularidad como "el número de bordes que caen dentro de los grupos menos el número esperado en una red equivalente con bordes colocados al azar" (p.2)., por lo tanto, la función determina un menor número de conexiones que las esperadas en la red. Es decir, se presenta entre 2 grupos una cantidad de "bordes" pequeña, y al interior de cada comunidad el equivalente es mayor, lo que concluye que la estructura comunitaria alcanza grandes características atractivas.

$$\boldsymbol{Q} = \sum_{\boldsymbol{c=1}}^{\boldsymbol{n}} [\frac{\boldsymbol{L_c}}{\boldsymbol{L}} - \left(\frac{\boldsymbol{K_c}}{\boldsymbol{2L}}\right)^{\boldsymbol{2}}] \qquad \boldsymbol{(2)}$$

Dónde: n es el número total de particiones, Lc el número de enlaces contenidos en la partición c, L el número totas de enlaces y Kc el número total de grados de todos los nodos de la partición c.

3.6. Algoritmo Genético:

Este algoritmo funciona mediante técnicas de búsqueda y optimización, que mediante heurística se adaptan al aspecto de modularidad. Para efectos de este tipo de problemas, "el algoritmo genético va generando una población inicial y crea una conexión de vecindarios óptimos que luego se evalúan por medio de una función de aptitud la calidad de las conexiones encontradas en el respectivo método" (Said, Abbasi, Maqbool, Daud, & Aljohani, 2018, p.2). Para efectos de esta generación de cromosomas mediante la iteración, su representación será:

$$\boldsymbol{P} = \{\boldsymbol{P_1}, \boldsymbol{P_2}, \dots, \boldsymbol{P_n}\}$$

3.7. Operador cruzado:

Este operador utiliza una metodología similar a la generación de IP, sin embargo, está ligado a las crear de enlaces entre nodos para a partir de ella generar nuevas descendencias, durante esto se evalúan sus "longitudes de vector" con valores de 0 o 1. Durante este procedimiento tal operador escoge los nuevos genes siempre y cuando su valor con el "progenitor" sea de uno, con el fin de que las cadenas de descendencia contengan genes conectados lo cual prioriza la optimización de la estructura por sobre los óptimos locales.

3.8. Coeficiente de *Clustering:*

El coeficiente de agrupamiento (CC por sus siglas en inglés) es una medida de eficiencia de las comunidades. Dicha métrica evalúa las tendencias de agrupamiento de los vértices en el grafo (Nascimento, 2014, p.2). De acuerdo con Said et al., (2018) "el CC calcula la necesidad de

conexión entre los vecinos de un nodo y representa la probabilidad de conexión entre dos vecinos aleatorios de un nodo" (p.3), dicho cálculo se realiza mediante la evaluación de grados de conexión entre un conjunto de vecinos de la estructura de red, si el nodo $\boldsymbol{v_i}$ tiene un único vecino $\boldsymbol{v_j}$ (es decir un valor de 1) entonces estos dos nodos están conectados, sin embargo si el nodo $\boldsymbol{v_i}$ no presenta ningún vecino (o valor de 0), dicho nodo se aislará en una comunidad separada para posterior análisis. La representación matemática es:

$$\boldsymbol{C_{vi} = 2 * \frac{L_i}{K_i(K_i-1)}} \quad \boldsymbol{(3)}$$

Donde: L_i corresponde al número total de enlaces entre vecinos del nodo V_i y K_i representa el grado del vértice *i.*

3.9. Coeficiente de Clustering basado en Algoritmo Genético:

Para optimización de los resultados mediante el GA, el CC-GA se encarga de escoger y evaluar las descendencias mejoradas mediante los distintos operadores y componentes del algoritmo en cada iteración, con el objetivo de maximizar la modularidad. La inicialización del GA se hace mediante definición de los siguientes parámetros: el tamaño de la población, tasas de mutación y el operador cruzado, a partir de allí para cada nodo se calcula primero su CC para empezar la generación de IP y obtención de cromosomas de red esperadas en cada iteración, concluyendo finalmente determinación de una estructura comunitaria robusta.

3.10. Benchmarking:

Son técnicas especializadas para determinar el rendimiento de los algoritmos mediante la comparación con puntos de referencia. En el caso de las redes sociales esto se hace por medio de estructuras de red conocidas anteriormente, con el fin de evaluar el algoritmo progresivamente mediante el contraste entre las comunidades detectadas en cada ajuste efectuado.

4. Preparación de datos y diseño del algoritmo

A continuación, se especifica el proceso de recolección, preparación de la información, así como la creación de la red y diseño del algoritmo.

4.1.Recolección de la información

La recolección de información parte de la familia de docentes adscritos a los grupos de investigación de las facultades de Ingenierías Fisicomecánicas (FIFM) y Fisicoquímicas (FIFQ), elegidas por su grado de representación de la información, de acuerdo con los criterios específicos para la elección óptima de los datos que implique tantas interacciones interdisciplinarias como sea posible.

Los datos utilizados se obtuvieron a partir del Sistema Nacional de Ciencia y Tecnología de Minciencias, se recolectaron los listados de los docentes vinculados a los grupos de investigación explorados en la página UIS en cifras, verificando que la información exportada estuviera dentro del periodo de vinculación vigente del docente en cada grupo, dando como resultado 19 grupos de investigación para FIFM y 11 para FIFQ. Cada grupo de investigación cuenta con un promedio de 100 docentes, posteriormente se procedió a consultar la hoja de vida electrónica de Colciencias CvLac. En primera instancia, con la finalidad de encontrar relaciones entre los miembros respecto a la dirección y codirección de trabajos de grupo con otras carreras distintas a las de vinculación del grupo de investigación arrojó un resultado total de 257 registros, en este paso se tuvo en cuenta la misma relación adicionándole si había colaboración con otras universidades, no obstante después de identificar que estos hallazgos no simbolizaban un peso representativo entre el conjunto de datos recolectados, se optó por conservar únicamente los datos de relaciones dentro de la UIS dejando un total de 220 registros.

Luego se usó como criterio de selección los docentes que poseían colaboraciones en la dirección o codirección de proyectos de grado de pregrado con otras carreras distintas a la escuela de vinculación del grupo de origen, concluyendo para la Facultad de Ingenierías Fisicomecánicas (FIFM) 93 docentes con distintas colaboraciones interdisciplinarias. En esta etapa se observan un total de 536 trabajos de grado dirigidos, concentrado en 31 docentes como los más representativos del grupo mencionado, haciendo constancia como las personas con más colaboraciones hechas en la facultad, véase apéndice C página 1. Por otra parte, los docentes que han colaborado con al menos un profesor dan una totalidad de 40 de ellos, de los cuales el grupo representativo se centra en 20 docentes con 80 colaboraciones realizadas con otras personas. Es importante resaltar que en esta información existe la posibilidad que se haya realizado más de una colaboración con el mismo docente, pero para este registro de información sólo se tomó en cuenta el vínculo de colaboración docente-docente y no la cantidad de colaboraciones entre ellos, véase apéndice C página 2. Finalmente, como se detalla en el apéndice C página 3, el mayor registro de colaboración se encuentra en el grupo de investigación Cómputo Avanzado de la escuela de Ingeniería de Sistemas, con el programa de Ingeniería Química con un total de 67 trabajos en conjunto. Seguido de este, se encuentre el grupo de investigación INNOTEC de Ingeniería Industrial, con el programa de Trabajo Social con un total de 57 trabajos, finalmente se encuentra el grupo de investigación GISEL de la Escuela de Eléctrica, Electrónica y Telecomunicaciones (E3T), en vinculación con el programa de Ingeniería Forestal con un total de 52 trabajo. Con ello se evidencia del mismo modo que estos son los grupos de investigación con mayores colaboraciones registradas en la FIFM.

Del mismo modo, en la Facultad de Ingenierías Fisicoquímicas (FIFQ) se encontraron un total de 80 docentes que han realizado colaboraciones interdisciplinarias con un total de 560 documentos, véase apéndice C página 4, de ellos 23 docentes representan el mayor porcentaje de

las colaboraciones con una concentración de 447. De la cantidad inicial, tan solo 33 han realizado colaboraciones con al menos otro docente como se puede apreciar en el apéndice C página 5. En este caso, encontramos un docente que ha realizado un total de 20 colaboraciones con otros docentes, posicionándose como el más representativo de esta clase. Por último, el Grupo de Investigaciones en Corrosión vinculado a la Escuela de Ingeniería en Metalúrgica, con el programa de Ingeniería Química, representa más colaboraciones con un total de 184 trabajos en conjunto, véase apéndice C página 6.

4.2.Preparación de los datos

Posterior a la recolección de los datos, fue pertinente la correcta preparación de estos, dado que, como punto de partida, el lenguaje de programación Python no reconoce caracteres especiales tales como las tildes. Con el fin de evitar reprocesos y confusiones con el algoritmo, se eliminó la múltiple aparición de los docentes en el archivo base (ver apéndice E), ya que algunos de ellos se encontraban adscritos a más de un grupo de investigación, a la vez, también se consideró adecuada la asignación a cada docente de un número de identificación para lograr una corrida más eficiente del algoritmo, dado que de esta forma no tendría que reconocer cierta cantidad de caracteres, sino un solo número.

4.3. Creación de la red social universitaria

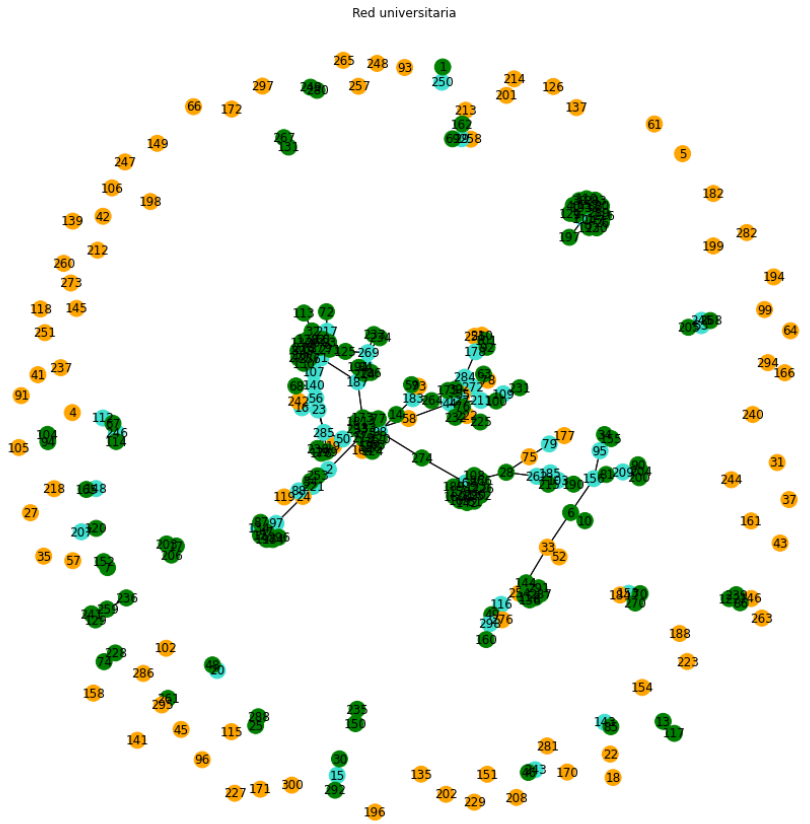
Con base en la información mencionada, se anexó la información en un fichero para plantear la red inicial con la librería Networkx(Hagberg et al., n.d.; *Software for Complex Networks — NetworkX 2.5 Documentation*, n.d.) de Python, donde un total de 166 docentes adscritos a cada uno de los grupos de investigación se extiende a 280 docentes mediante colaboración, definiendo como punto de partida a los docentes como los nodos de la red y los

enlaces como la cantidad de colaboraciones; ello genera un total de 300 nodos con 347 enlaces, no obstante, debido a que muchos de los datos encontrados fueron directores "independientes", se encuentran 146 enlaces entre ellos mismos lo que deja un total de 154 con un docente distinto (verde), cabe mencionar que dentro estos 146 relaciones independientes 95 de ellos no colaboraron con algún otro docente (naranja). Véase Figura 4.

La diferencia generada entre la lista extendida de docentes y el total de nodos encontrados se debe a que existen 20 docentes que no colaboraron con docentes adscritos a los grupos de investigación trabajados al momento, así como tampoco realizaron colaboraciones independientes.

Figura 4

Red universitaria, creada con Networkx



4.4.Diseño del algoritmo

Con base en la información obtenida durante la revisión de literatura, se da paso a la formulación del algoritmo a implementar en la presente investigación mediante la adaptación de elementos importantes del algoritmo planteado por Said et al., (2018) en "CC-GA: A clustering coefficient based genetic algorithm for detecting communities in social networks", el cual, como se ha mencionado anteriormente, tiene como objetivo aplicar un algoritmo genético partiendo del cálculo del CC en cada uno de los nodos, creando pequeños subgrafos de la red en donde cada uno de ellos esté conectado con un único nodo de su vecindario; este último elemento se conocerá como el cromosoma de la red, concepto que, acoplándose a lo encontrado en la revisión de literatura y su definición clásica, un cromosoma tendrá tantos genes como nodos exista, consiguiente a ello el gen se definirá como el único vecino del nodo evaluado por el CC para cada uno de ellos en toda la red y esto cambiará para la creación del siguiente cromosoma.

La utilización de la medida del CC tiene como fin encontrar un valor de ajuste de la red óptimo, conocido para los problemas de redes y CD como Modularidad, el cual se toma como valor de referencia para comenzar la tarea de selección y cruce de padres por medio del operador cruzado del GA, mediante la metodología descrita en la sección 3.7 del marco teórico, en el cual se utiliza un vector binario de bits aleatorios para la selección de genes de cada padre que irán en la nueva generación, con el objetivo de incrementar dicho valor progresivamente hasta encontrar una red final de más alta modularidad posible frente al valor de inicio, encontrando comunidades representativas. Cabe mencionar que de principio en dicho algoritmo se hizo uso de la librería de Python: Networkx, una librería con una diversidad de herramientas, funciones y demás utilidades para el estudio y análisis de redes.

Teniendo en cuentas estas consideraciones, el concepto metodológico del algoritmo, el estudio de la librería mencionada y demás aspectos importantes, se realizaron pequeñas modificaciones en momentos específicos del algoritmo para llevar a cabo nuevas consideraciones en la formulación del algoritmo con el lenguaje Python 3.8, escogido debido a su sintaxis de alto nivel, elemento que lo ha llevado a ser tan popular en los últimos años razón por la cual contiene gran variedad de librerías útiles para atender diversas necesidades. Para la escritura se utilizó el entorno de desarrollo integrado Spyder disponible en la plataforma de distribución de usuario *Anaconda Navigator*, junto con la importación de algunas librerías de análisis de redes, graficar de resultados entre otros (véase *Figura 5*).

Figura 5

Librerías importadas

```
import networkx as nx
import networkx.algorithms.community as nxc
from networkx.algorithms import community
import random
import matplotlib.pyplot as plt
import datetime
import math
```

Función calcCC:

Tal como se mencionó anteriormente, el principio del algoritmo comprende el cálculo del coeficiente de agrupamiento de cada nodo ante sus vecinos, motivo por el cual es necesario presentarlo como la primera función a crear. En este caso, gracias a la librería Networkx, acuñado con las siglas nx en su importación, contiene una función propia para tal fin llamada *clustering* cuyo recurso contiene todos los componentes de la ecuación (3)

Figura 6

Cógido función calcCC

```
def calcCC(red):
    CC=nx.clustering(red)
    #for nj in CC:
        #print(CC[nj])
    return CC
```

Función ReconsRed

Tal como se mencionó, la tarea principal del CC es crear un enlace único con alguno de sus vecinos y así crear un listado de cromosomas que cumpla con tales características, siguiendo dicha temática, los autores del artículo base para el presente proyecto, propusieron un modelo de construcción de nuevos subgrafos para ello.

Figura 7

Cógido función ReconstRed

```
def ReconstRed (rd):
    NuevoGrafo = nx.Graph()
    for nd in rd.nodes(rd):
        NuevoGrafo.add_nodes_from(nd)
        NuevoGrafo.remove_node(None)
        continue
    ListaNodos = []
    ListaNodos = NuevoGrafo.nodes()
    #print(ListaNodos)
    ListaEnlaces = []
    for node in ListaNodos:
        if (node != None):
            maxCC = -1
            EnlaceNodo = None
            Vecinos = list(iv.neighbors(node))
            #print(Vecinos)
        for vc in Vecinos:
            CCdeNodo = CC[vc]
            if (random.random()>0.5):
                if (CCdeNodo>maxCC):
                    maxCC = CCdeNodo
                    EnlaceNodo = vc
            else:
                if (CCdeNodo>=maxCC):
                    maxCC = CCdeNodo
                    EnlaceNodo = vc
        if (EnlaceNodo != None):
            NuevoGrafo.add_edge(node,EnlaceNodo)
            ListaEnlaces.append(EnlaceNodo)
        else:
            ListaEnlaces.append("sólo")
    gDic = {}
    gDic[NuevoGrafo] = ListaEnlaces
    #print(gDic)
    return gDic
```

Se inicializa un grafo vacío que se irá llenando con información de la red original y nuevos componentes que se adicionarán progresivamente tal como se observa en la *Figura 7*. Dada la situación de red vacía del principio es necesario eliminar elementos inconclusos como el nodo fuera de rango creado, para listarlos en una nueva variable correctamente. Después, así como en este caso, la lista generada por "ListaEnlaces" es una cadena de los nodos escogido entre el vecindario por su propio valor de CC; a partir de la conexión creada se construyen los genes, que conforman el cromosoma. En este caso, esta es una función de naturaleza iterativa, por lo cual en cada corrida de este habrá genes distintos en alguna posición de la cadena si su vecindario lo permita. Estos valores se guardan en el diccionario gDic con el fin de que cada ListaEnlaces (cromosoma) se identifique directamente con su identidad de subgrafo creado.

Función calcMOD y crearcomunidades.

Para el desarrollo del GA se debe relacionar en cada generación de subgrafos la medida de calidad “modularidad”, que como se afirmó con anterioridad es la encargada de medir la capacidad que tiene la red para poder dividirse a partir de particiones del grafo que se crearán con la función crearcomunidades, da paso a la determinación de parámetros, como el valor óptimo de referencia con el cual el GA empezará su tarea de optimización de resultados.

Figura 8

Código función calacMOD y función crearcomunidades

```
def calcMOD(NuevoGrafo):
    MOD=[]
    MOD=nxc.modularity(NuevoGrafo,crearcomunidades(NuevoGrafo))
    #print(MOD)
    return MOD

def crearcomunidades(NuevoGrafo):
    Comms = []
    communities_generator = community.girvan_newman(NuevoGrafo)
    Comms = next(communities_generator)
    #Comms = nxc.asyn_lpa_communities(NuevoGrafo)
    #print(Comms)
    return(Comms)
```

Cabe mencionar que la librería antes mencionada, también posee la función *modularity* que atiende perfectamente a lo planteado por la ecuación (2). Por otro lado, el fin principal de la segunda función, crearcomunidades, es determinar particiones de red iniciales para el cálculo de modularidad, tal como se observa se trabajará con 2 modelos de particiones de comunidad encontradas durante el estudio de Networkx, creando a su vez una modificación al modelo propuesto en el formato original del algoritmo. La primera de ellas son las particiones clásicas postulada por Girvan y Newman, conocido en la literatura como pioneros en este tipo de estudio de redes, mientras que la segunda es el modelo LPA o propagación de etiquetas asincrónico, de forma en que crea comunidades de forma probabilística por medio del establecimiento de etiquetas a cada nodo hasta encontrar las etiquetas de mayor frecuencia en los distintos vecindarios creados. Ambos modelos de partición se utilizarán con el fin de realizar comparaciones entre sus resultados como apoyo profundo del análisis. Adicional a ello esta última función se utilizará posteriormente para el algoritmo para determinar, del mismo modo, nuevas particiones en la red temporal con mejor Modularidad y evaluar su evolución por medio de los cruces genéticos.

Función GuardarGeneraciones

Dada la necesidad de mantener en un listado las generaciones creadas de forma aleatoria con las anteriores funciones se crea la presente función, por medio de elemento tipo diccionario con el fin de establecer correctamente las relaciones de los Subgrafos generados y los cromosomas de los mismos, con su respectiva modularidad calculada (Q). Adicional a obtener datos correctamente relacionados, es importante mantener un orden de valores descendente, para lo cual se usan las funciones *sorted* y *sort reverse* propias del lenguaje.

Figura 9

Código función GuardarGeneraciones

```
def GuardarGeneraciones (Q, NuevoGrafo,ListaEnlaces):
    Generaciones[Q] = NuevoGrafo
    ListaEnlacesDic[Q] = ListaEnlaces
    if (Q not in Modularidades):
        Modularidades.append(Q)
        sorted(Modularidades)
        Modularidades.sort(reverse=True)
        #print(Generaciones,ListaEnlacesDic)
    return Generaciones
```

Función retTopMOD

Retoma le mejor valor de modularidad de entre las generaciones creadas y guardadas, valor de referencia y partida del GA.

Figura 10

Código función retTopMOD

```
def retTopMOD ():
    if (not Modularidades is None):
        Tmod = Modularidades[0]
        #print("Mejor Modularidad:",Tmod)
    return Tmod
```

Función SeleccionaryCruzarGens

Definidas ya las funciones anteriores, se llamarán en la presente función que se encargará en acceder a cada uno de los cromosomas y subgrafos por medio del valor su modularidad de forma ordenada para la selección de “padres” que se cruzarán para crear las nuevas generaciones del algoritmo. De primera instancia se inicia a la función por medio de parámetros de entrada del mayor valor de modularidad encontrada en pasos anteriores para determinar el valor de referencia de inicialización. Así como se observa, la función contiene los parámetros de entrada de tasa de

cruce de generaciones y cromosomas acorde al tamaño de la red, facilitando la operación y proceder con la creación de pequeños subgrafos temporales para encontrar la mejor solución en la detección de comunidades. Estos valores de entrada, según lo planteado en el algoritmo original, se mantienen con el fin de que en la etapa de validación de este se pueda realizar el análisis con mayor facilidad para ajustar los parámetros de cruce en la red universitaria, pues se basará únicamente con la línea de código: *if (Ugen>=100)*

Figura 11

Código función SeleccionaryCruzarGens

```
def SeleccionaryCruzarGens (Ugen,red,TasadCruce):
    GQ = retTopMOD()
    CruceGen = math.floor(Ugen*0.1)
    if (CruceGen<10):
        if (Ugen>10 and Ugen<100):
            CruceGen = 10
        if (Ugen>=100):
            CruceGen = Ugen*TasadCruce
        else:
            CruceGen = Ugen
    TasaCruce = math.floor(CruceGen*TasadCruce)
    if (TasaCruce<2):
        TasaCruce = 2
    Pn = {}
    PnListaEnlaces = {}
    for tn in range(TasaCruce):
        index = 0
        while (index<CruceGen):
            g1 = nx.Graph()
            g2 = nx.Graph()
            listenG1 = []
            listenG2 = []
            ListaEnlacesSalida = []
            g1 = accedMejoresGens(index)
            g2 = accedMejoresGens(index+1)
            listenG1 = retMejoresEnList(index)
            listenG2 = retMejoresEnList(index+1)
            if (not g1 is None and not g2 is None):
                RedSalida = nx.Graph()
                RedDic = CruzarGens(g1,g2,listenG1,listenG2)
                RedSalida = nx.Graph()
                for etRed, ListSalida in RedDic.items():
                    RedSalida = etRed
                    ListaEnlacesSalida = ListSalida
                newMod = calcMOD(RedSalida)
                #print(newMod)
                Pn[newMod] = RedSalida
                PnListaEnlaces[newMod] = ListaEnlacesSalida
            index += 2
    for kk,vv in Pn.items():
        if (kk>GQ):
            GQ = kk
            ListaEnlaces = PnListaEnlaces[kk]
            GuardarGeneraciones(kk,vv,ListaEnlaces)
    Mutación(TasadMutacion,red)
```

Definidos los parámetros del cruce de generaciones y la creación de variables vacías para las nuevas generaciones, se procede con acceder a aquellos "padres" que se seleccionarán por medio de las funciones *retMejoresEnList* y *accedMejoresGens,* el valor "Index" va iterando acorde a la cantidad de generaciones creadas, el cual corresponde a la variable nombrada como Modularidades cuyos valores fueron guardados respectivamente con la función *GuardarGeneraciones*.

Figura 12

Código función retMejoresEnList y accedMejoresGens

```
def retMejoresEnList (index):
    if (index >= len(Modularidades)):
        return None
    tr = Modularidades[index]
    eListr = ListaEnlacesDic[tr]
    #print(tr)
    #print(eListr)
    return eListr

def accedMejoresGens (index):
    if (index >= len(Modularidades)):
        return None
    else:
        gf = nx.Graph()
        tr = Modularidades[index]
        gf = Generaciones[tr]
        #print(gf)
        return gf
```

Una vez seleccionados aquellos padres, se procederá con el respectivo cruce entre ellos, cuyos resultados se irán enlistando sobre generaciones temporales "Pn" en formato diccionario, respondiendo a la necesidad de mantener las relaciones de identidad entre los subgrafos y sus enlaces unitarios, para luego hacer comparación con el valor de modularidad de referencia mencionado al principio de la codificación donde se escogerán únicamente aquellas generaciones

que cumplan la condición de mejores que la anterior se guardarán como nuevos elementos con la función *GuardarGeneraciones* que procederán a la etapa de mutación.

Función CruzarGens:

Procediendo con los postulados del algoritmo original, esta función se encargará de sobreponer los enlaces de los padres seleccionados sobre la metodología de Said et al., el cruce de los cromosomas que se realiza a través de una lista aleatoria de "bits" binaria aleatoria y así conocer la descendencia de conexión de la red. Una vez creada la lista de bits, se irá evaluando uno a uno los valores del mismo, y se comparará con la posición específica de los nodos a los cuales hace referencia en la lista de ambos cromosomas, si el valor de bit es igual a uno se tomará el nodo y el gen de padre g1 y se creará un enlace entre ellos, en el caso contrario se hará el mismo procedimiento con el padre g2, garantizando que las nuevas generaciones contengan los mejores genes de cada uno de los padres.

Figura 13

Código función CruzarGens

```
def CruzarGens (g1,g2,listenG1,listenG2):
    ListaEnlacesSalida = []
    RedDic = {}
    RedSalida = nx.Graph()
    CantNodos = g1.number_of_nodes()
    bits = [0,1]
    randombits = []
    g1Nodos = list(g1.nodes())
    g2Nodos = list(g2.nodes())
    for i in range(CantNodos):
        randombits.append(random.choice(bits)) #Esta línea se encarga de
                                               #generar el listado
                                               # de bits aleatorio
    #print(randombits)
    for ct in range(CantNodos):
        bit = randombits[ct]
        if bit == 1:
            nede = g1Nodos[ct]
            eListaNodo = listenG1[ct]
            RedSalida.add_edge(nede,eListaNodo)
            if (eListaNodo != None):
                ListaEnlacesSalida.append(eListaNodo)
                continue
            ListaEnlacesSalida[eListaNodo]
        else:
            nede = g2Nodos[ct]
            eListaNodo = listenG2[ct]
            RedSalida.add_edge(nede,eListaNodo)
            if (eListaNodo != None):
                ListaEnlacesSalida.append(eListaNodo)
                continue
            ListaEnlacesSalida[eListaNodo]
    #print("enlace final ",ListaEnlacesSalida)
    #print(RedSalida.edges())
    RedDic[RedSalida] = ListaEnlacesSalida
    return RedDic
```

Función Mutación:

Para finalizar el proceso del algoritmo se establece la mutación de este, en este caso se hace uso de la codificación del elemento *performSimpleMutation* que se centra en los objetivos de mutación esperados en el artículo base, el cual se encarga de garantizar que las comunidades pequeñas traten de fusionarse con otras para crear comunidades más grandes, y en medida de que sea posible aumentar así el valor de aptitud en un rango de iteraciones menor. Para ello se da inicialización de 2 redes, la primera contendrá las 10 mejores generaciones encontradas, mientras que la segunda será una copia de estos elementos originales para empezar a realizar las mutaciones. Luego se escoge de ante mano una comunidad aleatoria en la red mutada, ya dentro de ella, escoger aleatoriamente alguno de ellos, verificando posteriormente que no haya nodos en común con otras comunidades, buscando que cada nodo pertenezca a una única comunidad. Una vez establecido esto, se continua con la creación de enlaces entre los nodos escogidos y un nuevo vecino acorde a tales exigencias, para finalmente encontrar una nueva red temporal mutada en donde se evaluará su valor de aptitud, se comprará con el mejor valor encontrado hasta el momento y se guardarán finalmente las mejores mutaciones.

Figura 14

Código función Mutación

```
def Mutación (TasadMutacion,red):
    rf = nx.Graph()
    redMutada = nx.Graph()
    for nt in range(0,10):
        rf = accedMejoresGens(nt)
        EnListar = retMejoresEnList(nt)
        if (rf is None):
            break
        else:
            EnlacesRedMutada = list(EnListar)
        redMutada = rf.copy()
        TMutar = TasadMutacion
        nwcom = crearcomunidades(redMutada)
        for j in range(0,5):
            if (len(nwcom)>1):
                nuevoVecino = None
                ComunNodoAleatorio = None
                while(True):
                    if (len(nwcom)<2):
                        break
                    ComunNodoAleatorio = random.randrange(0,len(nwcom))
                    Comunidadn = nwcom[ComunNodoAleatorio]
                    if (len(Comunidadn)>1):
                        ComunNodoAleatorio = random.choice(list(Comunidadn))
                        nodosVecino = red.neighbors(ComunNodoAleatorio)
                        VecinosExternos = [x for x in nodosVecino if x not in Comunidadn]
                        if (len(VecinosExternos)>0):
                            nuevoVecino = random.choice(VecinosExternos)
                            break
                        else:
                            break
                    fl = True
                    if (0 in g1Nodos):
                        VecinoActual = EnlacesRedMutada[ComunNodoAleatorio]
                        fl = False
                    else:
                        VecinoActual = EnlacesRedMutada[int(ComunNodoAleatorio)-1]
                    if (nuevoVecino !=None):
                        redMutada.add_edge(ComunNodoAleatorio,nuevoVecino)
                        if (fl==False):
                            EnlacesRedMutada[ComunNodoAleatorio] = nuevoVecino
                        else:
                            EnlacesRedMutada[int(ComunNodoAleatorio)-1] = nuevoVecino
                deltaMod = calcMOD(redMutada)
                TopModR = retTopMOD()
                if (deltaMod>TopModR):
                    GuardarGeneraciones(deltaMod, redMutada, EnlacesRedMutada)
```

4.5.Código del algoritmo

Una vez definidas las funciones a utilizar en el algoritmo final, cada una de ellas se irán llamando de forma secuencial y ordenada con tal de cumplir los fines específicos de cada una de ellas. De principio de inicializan variables de entrada como la Tasa de Cruce y la Tasa de Mutación, ambas dadas en los mismos valores del artículo original para la programación del algoritmo; este consta de 2 etapas.

La primera etapa, tal como se observa en la *Figura 15*, se concentra la utilidad e importancia del CC para la creación de los cromosomas de la siguiente manera:

- Dada la red a analizar se procede con el cálculo del CC de cada uno de sus nodos
- Se determina la cantidad de generaciones a usar.
- En esas generaciones se crearán los subgrafos temporales nombrados como NuevoGrafo
- Se procede con la creación de los subgrafos por medio de la función ReconstRed.
- Se calcula la modularidad de cada subgrafo creado.
- Finalmente se guardan los resultados ordenados con su respectiva generación

Figura 15

Etapa 1 implementación del algoritmo

```
iv = nx.read_gml("InvestigadoresU.gml")
print('Red',iv, iv.number_of_nodes(), 'nodos y', iv.number_of_edges(), 'enlaces')
DensI = print("Densidad inicial:",nx.density(iv))
crt = 100    #Criterio de terminación para cuando no hay mejoría en la modularidad
tiempoIn1 = datetime.datetime.now()
CC = calcCC(iv)
for itr in range(0,100):
    NuevoGrafo = nx.Graph()
    nuevoDic = ReconstRed(iv)
    for k,val in nuevoDic.items():
        NuevoGrafo = k
        ListaEnlaces = val
        Q = calcMOD(NuevoGrafo)
        Generaciones = GuardarGeneraciones(Q, NuevoGrafo, ListaEnlaces)
        break
```

La segunda etapa viene la aplicación neta del componente genético del algoritmo:

- Tomar el valor modularidad más alto encontrado en la primera etapa.
- Crear una red temporal, que será aquella a la que le corresponda el valor tomado anteriormente

- Se crean comunidades temporales en esta red temporal
- Siempre y cuando se cumpla la condición del criterio de parada, proceder con la selección y cruce de padres.
- En este punto, se calcula constantemente el valor de aptitud alcanzado por cada una de las nuevas generaciones y se compararán con el valor tomado al principio de esta etapa.
- Si se encuentra un mejor valor que el anterior, éste se actualizará, de lo contrario seguirá siendo hasta que se encuentre un mejor valor o sucedan más iteraciones que el criterio de parada establecido.
- Una vez se comprueba que no hay oportunidad de mejora, se termina el ciclo condicional del algoritmo.
- Finalmente se procede a guardar con nueva nomenclatura los resultados alcanzados y crear las particiones que dan a la red la característica de una estructura comunitaria robusta con respecto a su forma original.

Figura 16

Etapa 2 Implementación del algoritmo

```
Ugen = len(Generaciones)
GQ = retTopMOD()
M1 = GQ
print("Modularidad inicial:", round(GQ,5))
RedFin = nx.Graph()
RedFin = Generaciones[GQ]
crearcomunidades(RedFin)
tiempoIn2 = datetime.datetime.now()
cont = 1
iteraciones = 1
while(True):
    SeleccionaryCruzarGens(Ugen, iv, TasadCruce)
    newMod = retTopMOD()
    if (newMod <= GQ):
        cont += 1
        if (cont >= crt):
            break
    else:
        GQ = newMod
        cont = 0
    iteraciones += 1
    fig2 = plt.figure("Cruces")
    plt.xlabel("Iteraciones")
    plt.ylabel("Modularidad")
    fig2 = plt.title("Red Universitaria Modularidad vs Iteraciones")
    fig2 = plt.plot(iteraciones,GQ,"p-",color="orange")
print("t de cruce:",(datetime.datetime.now ()-tiempoIn2))
M2 = GQ
print("Modularidad Red Final:",round(GQ,5))
print("porcentaje de variación modularidad:", (M2-M1)/M1)
RedFinal = nx.Graph()
RedFinal = Generaciones[GQ]
ComunidadesFinales = crearcomunidades(RedFinal)
print(list(ComunidadesFinales), len(list(ComunidadesFinales)))
print("CC promedio de la red:",round(nx.average_clustering(RedFinal),5))
DensF = print("Densidad final:",nx.density(RedFinal))
```

Cabe destacar que varios elementos que se observan en ambas etapas facilitarán la tarea de análisis de los resultados que son inherentes a la estructura del algoritmo. Adicional ello, los valores de criterio de parada (crt) y cantidad de generaciones (range (0,100) de la *Figura 15*), son valores temporales, pues se mantendrán para la etapa de validación mediante benchmarking para posterior ajuste de los mismos al final de la misma.

5. Validación del algoritmo

Con el fin de comprobar el correcto funcionamiento del algoritmo antes de proceder con el análisis de la red de la universidad, se procedió a hacer una validación por medio de problemas del benchmarking para 3 redes de distintas dimensiones, a las que se pudo acceder en el formato requerido, utilizadas igualmente en el artículo base. Estas redes fueron Red Zachary's Karate Club, Red de Delfines y Red de Libros políticos[3].

5.1. Validación con partición Girvan y Newman (GN)

En primera instancia se realizan pruebas con las particiones de comunidad del método Girvan y Newman, no obstante, tal como se puede observar en el apéndice I este modelo de particiones en las redes mencionada presenta la desventaja que no otorga oportunidad de mejora en el valor de aptitud de la red pese a que se genere una gran cantidad de generaciones e iteraciones en cada una de las prueba, si bien los resultados de la misma son altos y representan un valor muy concreto a sus estructuras comunitarias, adicional a eso, un tiempo de convergencia que no supera los 30 minutos con la red más extensa, el hecho de que no se genere una optimización durante las iteraciones en el resultado puede no generar la suficiente confianza en la estructura del algoritmo utilizado, sin embargo el hecho que se encuentre una estructura comunitaria mejor que el punto de partida demuestra aún aspectos positivos del mismo.

[3] A estas redes se pudo acceder por medio de 'Network Data' <http://www-personal.umich.edu/~mejn/netdata/> [accessed 28 December 2020]. La primera de ellas, bien conocida en la literatura, donde cuyos integrantes del club después de un conflicto entre los administradores, deciden autónomamente irse con alguno de ellos, separando los bandos del club en 2. La segunda es una red de delfines que conviven en Nueva Zelanda representando la asociación frecuente entre ellos. Mientras que la última se basa en libros de política estadounidense que estudia de qué forma fueron comprados.

Complementario a ello, se procuró la observación de los tiempos de convergencia del algoritmo con una tasa de cruce mayor, pronosticando algunas demoras en el mismo debido a la dependencia de la cantidad de generaciones creadas, acorde a lo propuesto en la etapa de cruce se decide disminuir el criterio de parada para poder analizar con mayor facilidad tales relaciones, esto último con el fin de observar la existencia de alguna variación de resultados. Sin embargo, este caso también fue negativo, dado que se alcanzan resultados inmediatos del valor de modularidad óptimo (ver apéndice J). Adicionalmente, cuando el tamaño de red es más grande, Libros Políticos, a pesar de la disminución en la cantidad de generaciones creadas, se encontró que el tiempo de convergencia es considerablemente grande con respecto a las pruebas anteriores, lo cual lleva a la decisión de no aumentar la tasa de cruce si se tiene por objetivo menores tiempos de convergencia.

Aun así, estas particiones otorgan una referencia de estructura de comunidad representativa, pues su valor de aptitud está cerca del 1 y la cantidad de comunidades encontradas demuestra una buena cohesión en cada red en general. Realizando una comparación entre los resultados del artículo con las soluciones obtenidas, véase tabla 4, también se encuentra una variación importante a considerar pues todas están por encima del 50% de mejoría, con ello se garantiza una estructura comunitaria robusta entre los resultados, que procederá en no ignorar estas particiones para efectuar pruebas en la red universitaria.

Tabla 4

Benchmarking con particiones GN

	Modularidad alcanzada		
Red	**CC-GA Adaptado**	**CC-GA Original**	**Variación**
Zachary's Karate Club	0.80316	0.42	91.229%
Libros Politicos	0.92181	0.529	74.255%
Delfines	0.80851	0.527	53.417%

Nota: La columna de CC-GA son los alcanzados por el algoritmo creado para el presente documento, mientras que la columna CC-GA original son los resultados obtenidos en el artículo base del proyecto.

5.2. Validación con particiones Asyn-LPA

En consecuencia, con los resultados anteriores para esta etapa se decidió agregar una medida extra para determinar las variaciones de las estructuras comunitarias, en este caso se escogió la densidad de la red, el cual disminuye conforme la modularidad aumenta representando que las particiones encontradas presentan pequeñas o nulas conexiones entre las distintas comunidades, mientras que al interior de cada comunidad la conexión de los nodos es fuerte.

En este caso, los resultados en cada prueba fueron más fructíferas que en la situación anterior, a lo cual este proceso se separó en 2 partes, validación por medio de las redes pequeñas Karate y Delfines, mientras que la red de libros políticos se trabajó a parte para el ajuste progresivo de los parámetros a utilizar con la red universitaria.

5.3. Validación redes pequeñas

Tomando de primera instancia las redes de menor tamaño, dados sus tiempos de convergencia rápida, sin variar los parámetros de criterios de parada, con este modelo de comunidades los resultados fueron concretos, pues con base en lo deseado se obtuvo evolución en el valor de ajuste de estas, garantizando que siempre puede haber una mejor estructura en ellas. Véase apéndice K.

Tabla 5

Modularidad alcanzada particiones LPA redes pequeñas

	Modularidad alcanzada				
Red	**CC-GA Adaptado**	**Mod prom**	**DesvEst**	**CC-GA Original**	**Variación**
Zachary's Karate Club	0.80658	0.80487	0.0024	0.42	92.043%
Delfines	0.79114	0.78907	0.0029	0.527	49.334%

Nota: Para estas redes se realizó el análisis sobre 2 pruebas hechas, a lo cual se calcula el promedio del valor de aptitud alcanzada y la desviación estándar de las mismas. Criterio de parada 200 iteraciones.

Ahora bien, con lo observado en la *Tabla 5* se determina que las modularidades encontradas en las redes no varían demasiado con respecto las observadas en la *Tabla 4*, así como su comparación con los resultados del artículo, la variación sigue siendo positiva y de porcentajes de gran valor, no obstante, en cuanto a la red de delfines hubo una disminución del 4.09% con respecto al modelo de partición GN. Cabe mencionar dentro de los aspectos nuevos en la evolución de resultado, en el presente modelo, la variación de los mismo es más fructífera, pues con la presencia de nuevos fenómenos da paso a entender de mejor manera la estructura de la red, lo que conllevó anticipadamente a realizar análisis más amplios y a disminuir a la mitad el criterio de parada para posteriormente, volver a realizar pruebas.

Tabla 6

Modularidad alcanzada

	Modularidad alcanzada				
Red	**CC-GA Adaptado**	**Mod prom**	**DesvEst**	**CC-GA Original**	**Variación**
Zachary's Karate Club	0.82275	0.8147	0.0114	0.42	95.893%
Delfines	0.79512	0.7886	0.0092	0.527	48.397%

Nota: En esta fase se mantuvo la cantidad de pruebas anterior con un criterio de parada de 100

Con los criterios de parada reducidas a la mitad, se evidencia que no perjudica el hallazgo una modularidad óptima final, en estos descubrimientos (ver últimas 2 páginas del apéndice K) se destaca una disminución considerable en las iteraciones requeridas para encontrar resultados, característica que a su vez disminuye en gran medida los tiempos de convergencia, e incluso encuentra valores de aptitud levemente más altos. Dados los comportamientos observados, en estas redes cada prueba se realizó 2 veces en cada una de estas, razón por la cual se exhiben el promedio y desviación estándar de las modularidades encontradas para exponer que existe bajo rango de variación en cada prueba garantizando la calidad de los resultados, no obstante se percibe que las variaciones de modularidad durante evolución no superan el 10% entre ellas, lo cual, si bien no demuestra mejoría representativa de este valor de aptitud, cuando se hace comparación de las densidades de inicio y final en cada red ésta es superior al 60% en cada una, afianzando así que las estructuras comunitarias son provechosas.

5.4. Validación y ajuste de parámetros con red Libros Políticos

La razón de utilizar la presente red por separado se debe a que por su tamaño los parámetros del algoritmo a utilizar en el momento de la función seleccionar y cruzar generaciones será la misma, que en resumen tomará la cantidad de generaciones como el porcentaje de la tasa de cruce brindad.

Así, se procede con el criterio de parada inicial de 200 iteraciones, y una tasa de cruce de 0,2, lo cual quiere decir que si esta red tiene un total de 105 nodos en cada iteración se evaluarán 21 generaciones creadas para encontrar mejoría en el valor de aptitud. En estas primeras pruebas los tiempos de convergencia están por debajo de los 30 minutos, un resultado alentador para este tipo de red, además, los porcentajes de mejora en la modularidad en rangos están del 3% al 8%, rescatando que en cuanto a la densidad el incremento es constante con 80.0454%. Observando los

gráficos de evolución de esta red, véase página 1 del apéndice L, no se requiere de una gran cantidad de iteraciones para llegar a ella, necesitando cerca de la mitad comparándose con las redes pequeñas, del mismo modo se identifican zonas donde el mismo valor lleva una gran cantidad de iteraciones antes de tomar un mejor valor. Con esto último se lleva a contemplar la modificación parámetro de parada a la mitad y observar el nuevo comportamiento.

Con base en estos nuevos resultados, se observa que, si bien hay una disminución en los tiempos de convergencia, la variación modularidad acorde a las iteraciones se mantiene entre los rangos anteriores, por lo cual también se encuentra un mejor valor de aptitud rápidamente, sin contar pruebas en los que el valor de aptitud óptimo se encontraba desde el principio de la operación. Analizando las gráficas de los mismos, se observan nuevas áreas de evolución escalable debido a que se encuentran rangos "cortos" donde no hubo variación considerable o el crecimiento alcanzado fue bajo al momento de incrementar la aptitud, que conduce a elegir el parámetro de parada a 60 para observar el comportamiento y luego apreciar los escenarios con menor cantidad de generaciones a crear. Con estos últimos resultados, visualmente se puede ver con mayor detenimiento cómo se comportan las variaciones acordes a una cantidad de iteraciones menor, lo cual no impide que la mejoría sea "representativa" en cuanto a la medida de aptitud y densidad de la red, no obstante, el tiempo de convergencia se ha reducido considerablemente, lo cual puede ser positivo comparándose con las variaciones encontradas con un criterio de parada alto. En este punto recordando que se ha trabajado con 100 generaciones creadas y que para este tamaño de red la cantidad de generaciones a contemplar en la etapa de cruce es tan solo del 20%, se encuentra interés en realizar observaciones con este último valor de criterio de parada y una cantidad de generaciones menor.

Los tiempos de convergencia son muy buenos, pues con base en los ajustes se ha pasado de tener tiempos de 30 minutos a menores de 2 minutos en encontrar resultados positivos, no obstante, en los valores de modularidad esto no ha representado que el valor de aptitud alcanzada sea menor con menores tiempos, sino que han sido valores muy similares entre las modificaciones demostrando poca dependencia a la misma.

Finalmente efectuando las respectivas comparaciones de los resultados ante los hallados en el artículo base, con los 3 últimos criterios mencionados.

Tabla 7

Modularidad alcanzada con red Libros Políticos

	Modularidad alcanzada				
Red	**CC-GA adaptado**	**Mod prom**	**Desv Est**	**CC-GA original**	**Variación**
Libros Políticos crt 100	0.84885	0.84676	0.0015	0.527	61.072%
Libros Políticos crt 60	0.84924	0.84532	0.0028	0.527	61.146%
Libros Políticos crt 60, 50 gens	0.83936	0.83622	0.0022	0.527	59.271%
Desv Est	0.0056	0.0057			

Nota: Para cada escenario se recolectó la información sobre 3 resultados de las pruebas realizadas.

Visualizando los resultados de la *Tabla 7* se evidencia que la variación de parámetros no demuestra mucha diferencia en los resultados de aptitud alcanzados, los cuales, comparándolos con los alcanzados en el postulado original, la mejoría de este modelo de partición representa una variación altamente positiva, llegando a ser superior del 60% en los dos primeros escenarios.

6. Aplicación del algoritmo en la red universitaria

Una vez validado el algoritmo en redes tradicionales de la literatura y con base en los hallazgos de la etapa de ajustes de parámetros y considerando las dimensiones de esta red con respecto a los tiempos de convergencia se decide mantener el criterio de parada en 60 iteraciones, mientras que las generaciones a crear serán 70 para el modelo de particiones Asyn-LPA con el fin de obtener respuestas en un rango de menos de 30 minutos.

Por otro lado, para el modelo de partición GN, se realizaron pruebas con una gran cantidad de iteraciones, con el fin de identificar su comportamiento y lograr el ajuste de los parámetros con base en tales referencias. En este último caso se evaluaron únicamente los comportamientos visuales, véase apéndice M, donde se alcanzaron 650, 340 y 220 iteraciones para obtener el resultado óptimo, los tiempos de convergencia en estos casos fueron superiores a 2 horas, obteniendo incrementos de valor de aptitud poco menores al 1%, y valores muy cercanos entre cada escenario. Basándose en la observación de gráficas y tiempos de convergencia se decide mantener los parámetros usados con las particiones Asyn-LPA para realizar comparaciones en esta red. No obstante, dados los tiempos de convergencia tan amplios y los resultados observados en tales pruebas, los ensayos finales se realizan sin fase de mutación del algoritmo, pues se asume que este proceso infiere fuertemente en este tipo de particiones, adicional a ello dado que se alcanzan mejores valores de aptitud se encuentra redundante la operación para alcanzar mejorías en la red.

6.1. Análisis de resultados

Se realizaron 3 pruebas con cada tipo de partición, cuyos resultados se presentan en la *Tabla 8*, donde se evidencia que acorde a la desviación estándar, por más pruebas que se realice para la misma red, no habrá lugar para variaciones drásticas en estos en las métricas establecidas. En primera instancia se observa que, a comparación con las redes utilizadas en la etapa de benchmarking, la evolución en la densidad de esta nueva red no superó el umbral de 25%, el cual como se mencionó durante la aplicación Asyn-LPA, mide las conexiones existentes en la red, buscando que las conexiones entre comunidades sean las mínimas posibles, y si bien retomamos lo que se observa en la *Figura 4*, la información recolectada otorga una gran dispersión entre los docentes, característica que otorga sentido al resultado, no obstante, esto aún conlleva que en que la red existe una fuerte conexión entre los nodos al interior de cada una de las comunidades.

Tabla 8

Resultados en Red universitaria

Tipo de partición	Mod Máx	Mod prom	Desv Est Mod	VarMáx de densidad	Prom Var	DesvEst Dens	Cant Comunidades de mejor Mod
Particiones LPA	0.95655	0.95206	0.00021	23.9193%	23.4390%	0.0044	140
Particiones GN	0.9738	0.97365	0.00017	22.7666%	22.6705%	0.0017	133

Ahora bien, centrándose en la evolución del valor de aptitud en las pruebas hechas, ver apéndice N, se haya una mejoría cercana al 1% para las particiones Asyn-LPA y menos del 0,1% para las particiones GN, esto significa que para el primer modelo existe más oportunidad de mejora por medio de los operadores genéticos del algoritmo así como demuestran más escalabilidad durante las iteraciones realizadas, mientras que el modelo GN sigue presentando la desventaja

mencionada anteriormente, en contraste a ello la leve mejoría y los valores un 1.8% mayores con respecto al primer modelo y la notable cantidad menor de comunidades detectadas manifiesta buena calidad en su estructura comunitaria. Si bien en ambos casos la cantidad de iteraciones necesarias para obtener resultados no es mayor a 100, los tiempos de convergencia de las particiones GN son altos, ya que durante la etapa de cruce la comparación de valores es bastante minuciosa debido a la alta solidez y baja evolución del valor de aptitud representa mayor esfuerzo de procesador para realizar la labor.

Debido a la utilización de librerías propias de Python para llevar a cabo la fase de crear las comunidades en la red, se decide contrastar los resultados que se obtienen con estos sin la intervención del algoritmo.

Tabla 9

Comparativo de modularidad antes y después de CC-GA

	Modularidad alcanzada		
	CC-GA adaptado	**Antes de CC**	**Variación**
Particiones LPA	0.95206	0.82045	16.0407%
Particiones GN	0.97364	0.90240	7.8935%

Nota: La columna antes de CC son resultados del valor de modularidad alcanzada con el uso de las funciones de crear comunidades sin la ayuda del CC, funciones propias de Networkx.

Por consiguiente, como se observa en la Tabla 9, se verifica que el proceso del CC y su vinculación al GA otorgan una mejoría significativa de la estructura comunitaria de las redes con respecto a las comunidades creadas sin su uso, llegando a una variación del 16.04% para las particiones LPA y 7.89% para las GN, esto último demuestra la superioridad de aptitud que presenta el modelo pionero de creación de comunidades frente a los demás.

6.2. Análisis de comunidades

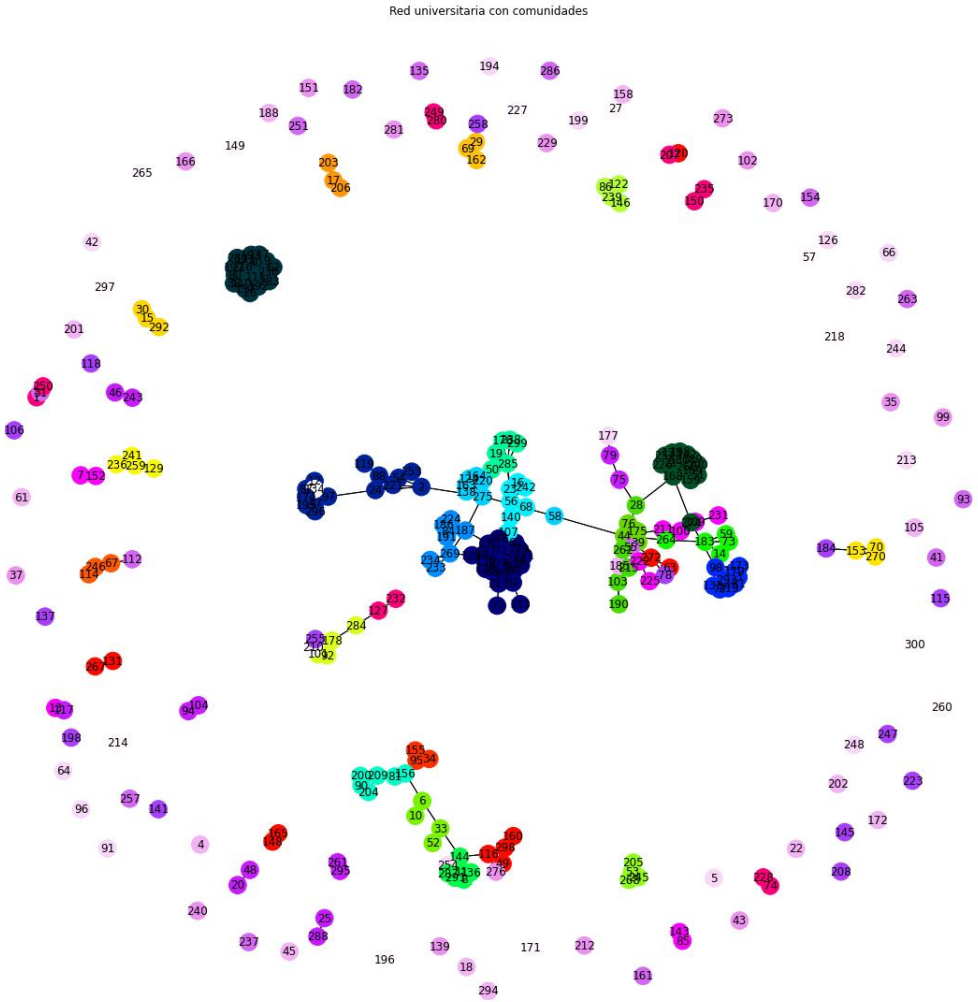
Con el fin de concentrar eficientemente el análisis de comunidades encontradas se procede inicialmente a determinar con base en los resultados hallados en la *Tabla 8*, recordando que cuando la modularidad es más cercana a 1, la calidad de las estructuras comunitarias encontradas es mejor, así como una menor desviación estándar entre las pruebas realizadas y un crecimiento estable de la variación de la densidad, se opta por escoger los resultados del modelo de partición GN como las comunidades a estudiar dado el complimiento de las anteriores características, lo que comprueba la fuerza en los enlaces de la red, con un total de 133 comunidades encontradas después de 93 iteraciones, donde se halla el mejor valor de aptitud.

En la

Figura 17 se observa la red universitaria con sus comunidades mapeadas por colores, con el fin de poder diferenciar las comunidades más grandes, se agrupan de manera especial las comunidades que tienen 2 y 1 individuos en 4 y 6 conjuntos respectivamente, ya que representan más de la mitad del total de comunidades. Así las comunidades grandes son aquellas que toman colores tierra y azules, los intermedios colores fríos entre azules claros y verdes, las medianas colores cálidos mientras que los conjuntos especiales se distinguen entre morados y blancos.

Figura 17

Red Universitaria con 133 comunidades encontradas. Generada con Matplotlib y Networkx de python



En la *Tabla 10* se muestra las comunidades que poseen más de un integrante, con los líderes de cada una, así como la cantidad de participantes en ella. Para observar la totalidad de las comunidades, véase el apéndice O.

Tabla 10

Comunidades encontradas

Comunidad	Tamaño comunidad	Integrante con más colaboraciones entre docentes
1	21	DIONISIO ANTONIO LAVERDE CATANO
2	18	GUSTAVO EMILIO RAMIREZ CABALLERO
3	15	JOSE ANTONIO HENAO MARTINEZ
4	14	FERNANDO VIEJO ABRANTE
5	9	FRANCISCO ALBERTO VELANDIA PATINO
6	8	JULIO ANDRES PEDRAZA AVELLA
7	7	SAMUEL FERNANDO MUNOZ NAVARRO
8	7	JAVIER ALBERTO SANABRIA CALA
9	6	JORGE ENRIQUE MENESES FLOREZ
10	6	VIATCHESLAV KAFAROV
11	6	JENNY MABEL CARVAJAL JIMENEZ
12	5	JULIAN GUSTAVO RODRIGUEZ FERREIRA
13	5	CARLOS ALBERTO RIOS REYES
14	4	CLARA ISABEL LOPEZ GUALDRON
15	4	ALEXANDER MARTINEZ RAMIREZ
16	4	DANIEL ALFONDO SIERRA
17	4	NELSON EDUARDO DIAZ DIAZ
18	4	JUAN DIEGO COLEGIAL
19	4	RAFAEL CABANZO HERNANDEZ
20	3	JORGE ANDRES RODRIGUEZ TORO
21	3	ANA RAMIREZ SILVA
22	3	CARLOS ALFONSO MANTILLA DUARTE
23	3	DIEGO FERNANDO VILLEGAS BERMUDEZ
24	3	JOHN FABER ARCHILA DIAZ
25	3	RUTH ZARATE RUEDA

Dentro de las 133 comunidades encontradas, se puede observar que la de mayor tamaño, cuenta con su líder Dionisio Antonio Laverde Cataño, adscrito al Grupo de Investigaciones en Corrosiones, donde gran parte de las colaboraciones fueron realizadas con la escuela de Ingeniería Química, seguido de Ingeniería Electrónica. De otro modo, es importante resaltar la existencia de

comunidades con un solo individuo dentro de ellas, esto debido a la gran dispersión de los datos y la incapacidad del algoritmo para la creación de enlaces en estos casos específicos, cabe resaltar que los docentes presentes en estas comunidades individuales se encontraban adscritos a un grupo de investigación de una escuela diferente a la del proyecto en cuestión, algunos de ellos en el rol de codirectores, pero sin información suficiente en la hoja de vida electrónica para la creación del vínculo pertinente.

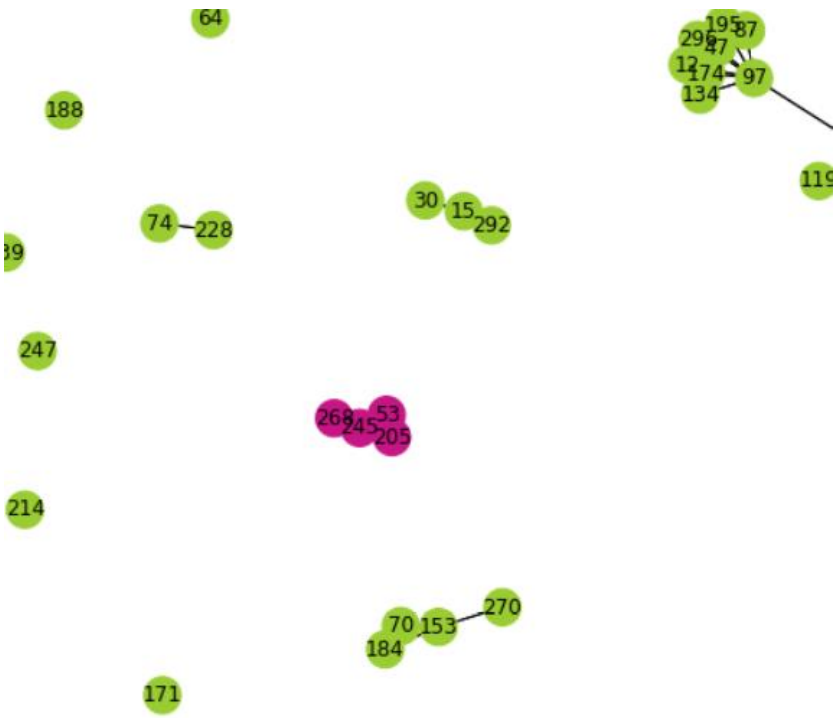
La detección de comunidades es una parte fundamental del análisis de redes sociales, ya que con ésta se puede obtener claridad acerca de los patrones de comportamiento en la creación de estructuras comunitarias, que no son características dadas al azar, sino definidas por algún tipo de interés o particularidad en común. Es por esto por lo que se procuró la búsqueda de comunidades en una red de colaboración de proyectos de grado dentro de la universidad, para de esta forma poder definir algunos atributos usuales y así diagnosticar las relaciones colaborativas con más fortaleza y sentar un precedente para posibles investigaciones futuras que pretendan impulsar la colaboración interdisciplinaria dentro del claustro.

Por lo tanto, se han seleccionado cinco comunidades al azar, para comprobar la validez del algoritmo, y asimismo analizar un poco más a fondo una pequeña muestra de la red creada. A continuación, se presentan cada una de ellas.

Comunidad 16: Como se puede observar en la *Figura 18*, esta comunidad se encuentra compuesta por cuatro docentes, identificada con el color magenta oscuro, además se puede notar que es una comunidad aislada de las demás ya que no posee un nodo fronterizo que la conecte con el resto.

Figura 18

Comunidad 16. Generada con Matplotlib y Networkx de python



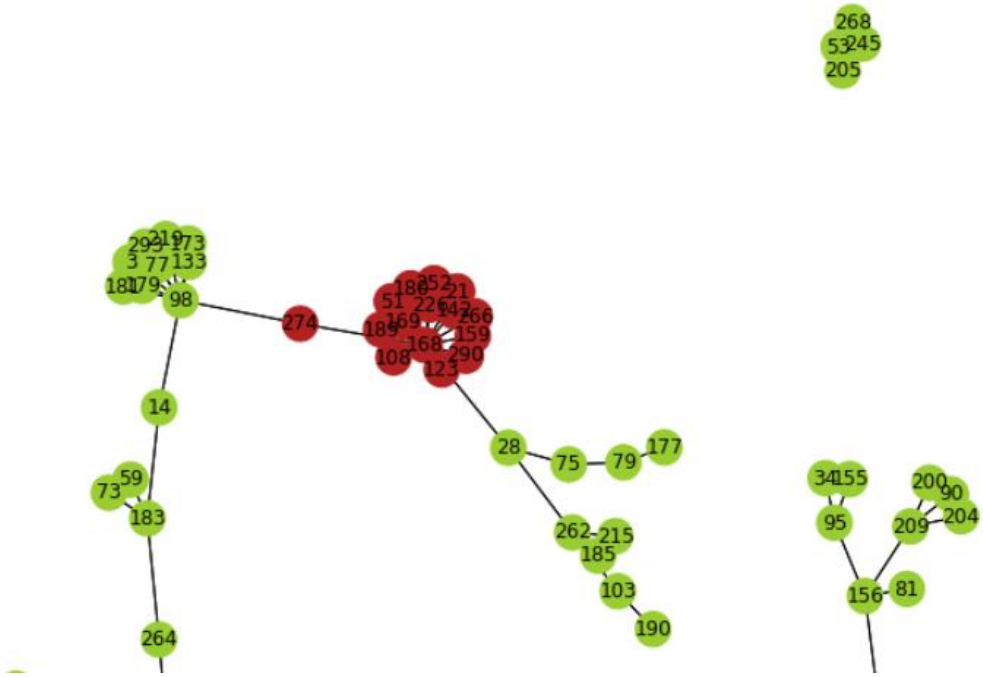
El líder de este grupo se consolida como Daniel Alfonso Sierra, docente adscrito a la Escuela de Eléctrica, Electrónica y Telecomunicaciones (E3T) con el grupo de investigación CEMOS, colaborando con otros tres docentes bajo el rol de director de proyectos para la escuela de Ingeniería de Sistemas. La docente Nydia Paola Rondón Villareal, al momento de realizar la colaboración con el profesor en mención, también se encontraba en vinculación con el grupo CEMOS en el año 2012, lo que nos lleva a visualizar la fuerza de la E3T en los proyectos de la escuela de Ingeniería de sistemas, dado que su dirección y codirección estuvieron a cargo de esta escuela específicamente en este proyecto. También es importante resaltar que el líder de la comunidad en el rango de 2014 a 2017 estuvo vinculado al grupo de investigación Cómputo Avanzado y a Gran Escala de Ingeniería de Sistemas, donde en este periodo realizó la dirección de 17 proyectos para la escuela de Ingeniería Electrónica.

Esta colaboración interdisciplinaria es una de las más frecuentes y esperadas de la red, dado el campo de conocimiento de cada una de las escuelas mencionadas, su constante relación y complementariedad de conocimientos, aunque puede llegar a ser demasiado hermética y descuidar la posibilidad de la transmisión de conocimiento a otras comunidades y disciplinas.

Comunidad 3: Se encuentra formada por 15 docentes, y está demarcada por el color ladrillo como se puede apreciar en la Figura 19.

Figura 19

Comunidad 3. Generada con Matplotlib y Networkx de python



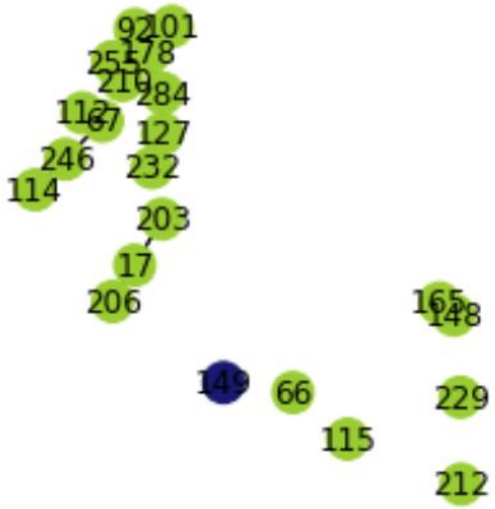
Esta comunidad se encuentra ubicada en la facultad de Ingenierías Fisicoquímicas, teniendo como líder a José Antonio Martínez Henao adscrito a los grupos Energy And Other Non Renewable Resources Research Group y Grupo de Investigación en Geología Básica y Aplicada de la escuela de Geología, dirigiendo en 58 proyectos de la escuela de Química, Ingeniería Química e Ingeniería Metalúrgica. Es bastante esperado este comportamiento, dado que el docente posee

un pregrado en Química y el resto de su trayectoria gira en torno a esta disciplina, también es importante resaltar que, aunque es de las comunidades más grandes se encuentran los nodos muy aislados de los demás, y sus nodos fronterizos se encuentran distribuidos en el líder y Sait Khurama Velásquez. Este último cumple una función importante en las colaboraciones interdisciplinarias, ya que es el conector entre una comunidad perteneciente a la FIFQ con otra de la FIFM, gracias a la colaboración de la escuela de Geología con Ingeniería Civil. A pesar de la cercanía existente entre los distintos conglomerados de nodos que se observan, estos no se agrupan dentro de la misma comunidad debido a la ausencia de colaboración entre los demás integrantes, pues tal como se observa, la comunidad mencionada y las comunidades del entorno se distribuyen a través del nodo líder.

Comunidad 131: Esta es una comunidad particular, dado que se encuentra compuesta por un solo individuo, el cual se puede apreciar de color en la Figura 20 de color azul oscuro.

Figura 20

Comunidad 131. Generada con Matplotlib y Networkx de python



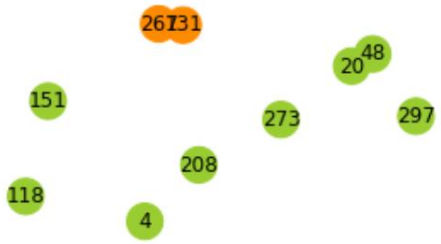
Este es un caso particular y que se generó con bastante frecuencia en la detección de comunidades de la red en cuestión; se trata de una estructura comunitaria individual generada por la dirección del docente John Franklin Cerón Abril adscrito a Energy And Other Non Renewable Resources Research Group, con una validez en el grupo de investigación desde 2018 a la actualidad, de la escuela de Geología en colaboración con Ingeniería de Petróleos.

El docente cuenta con un pregrado en Ingeniería Civil, un doctorado en la Universidad del Sur de Carolina en Geología y una maestría y especialización en Geofísica en la Escuela de Minas de Colorado, aún con todo lo anterior, el algoritmo no identificó suficientes recursos por lo que generó conexiones débiles de esta comunidad, hasta el punto de dejarla totalmente aislada.

Comunidad 28: Formada por 2 individuos, identificados mediante el color naranja en la Figura 21

Figura 21

Comunidad 28. Generada con Matplotlib y Networkx python



Esta comunidad se encuentra formada por la relación entre Jabid Eduardo Quiroga Méndez, Ingeniero Mecánico de la Universidad Industrial de Santander y PhD en Ingeniería Civil de la Universidad Politécnica de Cataluña y Rodolfo Villamizar Mejía, Ingeniero Eléctrico y

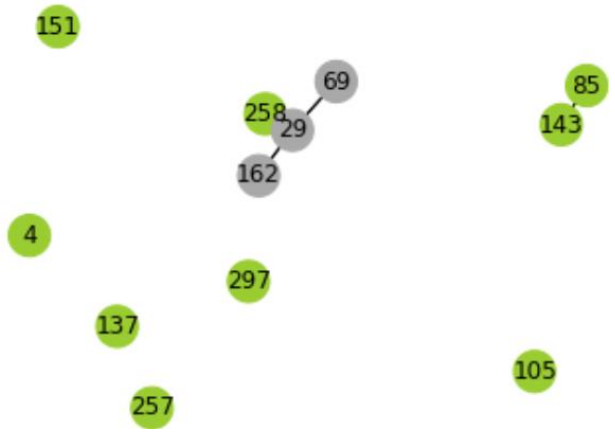
Electrónico de la UIS. La relación entre ellos fue generada mediante la dirección y codirección, respectivamente, de un proyecto de Ingeniería Electrónica bajo la dirección del grupo de investigación DICBOT de la escuela de Ingeniería Mecánica.

Dicho lo anterior, se evidencia un complemento de conocimientos interdisciplinarios, dado que el director del proyecto no posee estudios certificados específicos acerca del área dirigida, pero sí el codirector, aun así, la red al igual que el caso anterior no posee la suficiente fortaleza en los enlaces de la red, por lo que también se evidencia una comunidad aislada sin conexión con otras.

Comunidad 22: Formada por 3 individuos, identificados mediante el color gris en la Figura 22

Figura 22

Comunidad 22. Generada con Matplotlib y Networkx de Python



Así como en la comunidad 16, esta comunidad representa la relación entre las escuelas de Ingeniería de Sistemas e Ingeniería Electrónica, teniendo como líder a Ana Ramírez Silva, Ingeniera Electrónica y PhD. en Ingeniería Eléctrica, adscrita al grupo de investigación en diseño

de algoritmos y procesamiento de datos multidimensionales de la escuela de Ingeniería de Sistemas. Carlos Andrés Angulo Julio y William Alexander Salamanca Becerra también miembros de la comunidad e Ingenieros Electrónicos

Este caso es particular, debido a que la líder de la comunidad no tiene algún estudio específico en la disciplina correspondiente al grupo de investigación, más, no son especialidades aisladas, dado que algunos proyectos de Ingeniería Electrónica pueden llegar a necesitar el soporte de la Ingeniería de Sistemas en cuestiones de programación y automatización de características internas del proyecto en cuestión.

6.3. Análisis general de la red

De forma general de toda la red, las comunidades de un solo individuo conforman el 64% de la cantidad de comunidades encontradas, este grupo se conforma por 85 docentes de los cuales, de la información recolectada suministrada en cada CvLAC, su modalidad de dirección fue independiente, no obstante, en este caso se redujeron 10 agrupaciones esperadas, pues acorde a lo mencionado en la red inicial se habían encontrado 95 docentes que únicamente realizaron colaboración independiente. Adicional a esto, también se conformaron parejas de colaboración con un total de 23 que abarcan el 17%, ambos conjuntos en total comprenden el 81% del total de comunidades definidas, aun así, en el restante 19% los integrantes de cada comunidad presentan patrones de relación y colaboración interesantes en cada conglomerado, ya sea por títulos académicos, experiencia de profesión que brindan enriquecimientos a los resultados alcanzados.

6.4. Análisis de longitud de ruta más corta

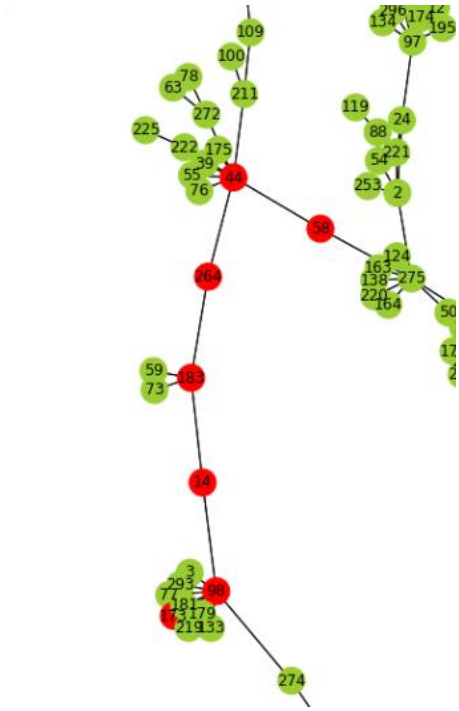
En cuanto a otras métricas de calidad dentro de la red, se encuentra la longitud de ruta más corta, que se define la distancia que se tiene que recorres por la red entre 2 nodos a través de los

enlaces existentes entre los demás, para término de las redes de colaboración esta métrica se traduce como la cadena de referidos con el cual se busca relacionar 2 docentes o autores que no se conocen, buscando crear nuevas colaboraciones entre sí. De principio este concepto tiene pocos resultados en la red universitaria de estudio, dada la ausencia de conexiones entre nodos, la dispersión de individuos, comunidades con baja cantidad de individuos y comunidades con conexiones casi lineales. No obstante, en las zonas más centrales, como se ve en la

Figura *17*, se evidencia la posibilidad de encontrar cadenas sobresalientes, como, por ejemplo, escogiendo el nodo 58, que es nodo frontera de la comunidad 7, y el nodo 173 perteneciente a la comunidad 5.

Figura 23

Cadena de referidos nodos 58 a 173. Generado con Matplotlib y Networkx de python



En la Figura 23 se muestra la ruta de referidos en color rojo entre los nodos mencionados, que es necesario contactar con un total de 6 docentes para que el 58, ubicado en la parte superior, logre contactar al nodo 173 en la parte inferior. Durante esta cadena, se encuentran docentes adscritos a los programas de ingeniería industrial, diseño industrial, ingeniería civil, ingeniería mecánica y geología, lo cual constata la interdisciplinariedad de las distintas comunidades que se conjuntan en esta zona de la red. El nodo 58 hace referencia al docente David Alfredo Fuentes Díaz quien ha pertenecido desde el 2004 al grupo de investigación GIEMA de la escuela de ingeniería mecánica ha presentado fuertes relaciones con docentes de ingeniería industrial, diseño

industrial e ingeniería de petróleos, así como trabajos dirigidos para ingeniería química. El segundo corresponde al docente Jose Ricardo Sandoval Ruíz, Msc en geología de la UIS, que a pesar de ello solo se encontró un trabajo codirigido en la misma área con un docente de ingeniería civil.

7. Conclusiones

En base a la revisión de literatura, el problema de detección de comunidades es ampliamente aceptado y utilizado para realizar estudios en las redes de colaboración, no obstante, la cantidad de artículos de investigación que comprendan la fusión del coeficiente de clustering en los algoritmos genéticos para abarcar esta clase de problemas es muy escaza.

La implementación inicial del algoritmo mediante benchmarking de redes sociales permitió garantizar el correcto funcionamiento del algoritmo planteado, presentando una mejoría representativa en el valor de aptitud para la mayoría de las redes escogidas. Adicionalmente se pudo rescatar las ventajas y desventajas que tienen los modelos de partición de las librerías de networkx, GN y LPA, para cada tipo de red adaptado al algoritmo CC-GA.

Por medio de la red de Libros Políticos se logró identificar correctamente los parámetros óptimos a utilizar en la red universitaria, ya que gracias a ella en esta última se logró obtener resultados en tiempos menores a 30 minutos, con excelente evolución del valor de aptitud. En cuanto a las particiones GN se llevó más trabajo para ajustar los parámetros correctamente, debido a su metodología de partición, no obstante, otorga mejores resultados de modularidad en cualquier escenario, adicional a mejorías del 7,9% con respecto a su aplicación sin instancias del coeficiente de clustering.

El CC-GA con modelo de particiones GN, logró detectar 133 comunidades con un valor de modularidad máximo de 0,9738, garantizando una estructura comunitaria de mejor calidad dada su cercanía con su máximo valor posible, siendo este el 1. El comportamiento general de las comunidades se basó en su gran mayoría en la colaboración entre escuelas de la misma facultad, así como docentes que no realizaron su pregrado en la escuela de origen del grupo de investigación

al que se encuentran adscritos, pero si alguno de sus estudios complementarios, a excepción de algunos casos atípicos, donde no existía concordancia entre la finalidad del grupo y la información acerca de la preparación del docente.

Un 81,79% de las comunidades están compuestas por estructuras individuales aisladas, dada la debilidad de los enlaces para la creación de vínculos con otros nodos, y a la vez conectar con otras comunidades. Si bien, toda información fue exportada de la página oficial de MinCiencias, es preciso resaltar que los resultados podrían variar con la existencia de información más detallada en tal plataforma por parte de los docentes vinculados a la red de investigación. Por otro lado, también se encuentran los trabajos de grado que solo cuentan con la característica de ser una colaboración interdisciplinaria mas no contemplan existencia de un nodo externo de relacionamiento.

En las 3 comunidades con mayor tamaño encontradas, sobresalen distintas escuelas, siendo la de mayor dimensión Ingeniería Metalúrgica, seguido de Ingeniería de Sistemas y Geología, donde se observan colaboraciones atípicas entre Ingeniería de Sistemas con Ingeniería Química a través del líder de la respectiva comunidad, mientras que en los demás casos se realizaron colaboraciones entre escuelas de la misma facultad.

Dentro de los hallazgos, se localiza participación en la transferencia de conocimiento en las facultades de Ingenierías Fisicomecánicas e Ingenierías Fisicoquímicas, pero no es desmesurada respecto a relaciones entre escuelas que no comparten líneas de conocimiento que pueden aportar soluciones sustanciales a los problemas planteados para el desarrollo de los proyectos de grado.

8. Recomendaciones

Dado que gran cantidad de la información recolectada para llevar a cabo este proyecto se indagó en la base de datos de CvLAC, sin embargo, es de anotar que falta actualizar e incluir información las colaboraciones hace que sea una primera aproximación para futuras investigaciones y de esta manera confiar más en los resultados.

Del mismo modo, la situación mundial actual ha dado un giro total, aumentando la necesidad de la búsqueda de información mediante bases de datos secundarias, haciendo imperativa la actualización, verificación y fiabilidad de la información presentada aquí. Por otra parte, de manera más específica, Colciencias evalúa cada hoja de vida electrónica y cada producto ingresado en ella, para otorgar un puntaje a cada grupo de investigación, lo que destaca aún más la importancia de los datos allí presentados.

La programación e implementación del CC-GA fue realizado mediante el lenguaje Python en apoyo con la distribución Anaconda Navigator y el entorno de desarrollo integrado Spyder, con las habilidades y conocimientos sobre programación de los autores, dando lugar a mejoras realizadas por expertos en el tema, enriqueciendo la creación de particiones y de esta forma mejorar los tiempos de convergencia. Es por esto que sería provechoso el incentivo del uso y aprendizaje de lenguajes de programación para los estudiantes de Ingeniería Industrial a manera de adquisición de conocimientos complementarios y facilidad en la realización de proyectos de la magnitud del presentado.

Incentivar la investigación enfocada en la detección de comunidades, dado el amplio campo de apertura al conocimiento en las diferentes áreas de desarrollo de la Ingeniería Industrial, tal como lo es la logística, y de esta forma lograr un análisis considerable de las redes sociales.

Con este modelo de investigación es posible explorar y expandir distintas áreas, que para futuras investigaciones sería interesante estudiar y conocer los comportamientos de la comunidad científica universitaria del país de los distintos grupos de investigación, así como es la participación nacional e internacional de eventos científicos, colaboración y temáticas sobre la creación de patentes, diseño de softwares, etc, pues podría otorgar patrones característicos para indagación sobre nuevos campos investigativos.

Referencias bibliográficas

Arasteh, M., & Alizadeh, S. (2019). A fast divisive community detection algorithm based on edge degree betweenness centrality. *Applied Intelligence*, *49*(2), 689–702. https://doi.org/10.1007/s10489-018-1297-9

Attea, B. A., Hariz, W. A., & Abdulhalim, M. F. (2016). Improving the performance of evolutionary multi-objective co-clustering models for community detection in complex social networks. *Swarm and Evolutionary Computation*, *26*, 137–156. https://doi.org/10.1016/j.swevo.2015.09.003

Bedi, P., & Sharma, C. (2016). Community detection in social networks. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, *6*(3), 115–135. https://doi.org/10.1002/widm.1178

Deng, X., Zhai, J., Lv, T., & Yin, L. (2017). *Efficient Vector Influence Clustering Coefficient Based Directed Community Detection Method*. *5*.

Easley, D. (2010). *Volumen 26 del TCE número 5 Portada y reverso | Teoría econométrica | Cambridge Core*. https://www.cambridge.org/core/journals/econometric-theory/article/ect-volume-26-issue-5-cover-and-back-matter/FA59F3056051664CB2969F82CC6CE0C8

Feng, Y., Yu, S., Zhang, K., Li, X., & Ning, Z. (2019). COMICS: A community property-based triangle motif clustering scheme. *PeerJ Computer Science*, *2019*(3), 1–22. https://doi.org/10.7717/peerj-cs.180

Folino, F., & Pizzuti, C. (2014). An evolutionary multiobjective approach for community discovery in dynamic networks. *IEEE Transactions on Knowledge and Data Engineering*,

26(8), 1838–1852. https://doi.org/10.1109/TKDE.2013.131

Girdhar, N., & Bharadwaj, K. K. (2019). Community Detection in Signed Social Networks Using Multiobjective Genetic Algorithm. *Journal of the Association for Information Science and Technology*, *70*(8), 788–804. https://doi.org/10.1002/asi.24164

Girvan, M., & Newman, M. E. J. (2002). Community structure in social and biological networks. *Proceedings of the National Academy of Sciences of the United States of America*, *99*(12), 7821–7826. https://doi.org/10.1073/pnas.122653799

Hagberg, A. A., Scult, D. A., & Swart, P. J. (n.d.). *Proceedings of the Python in Science Conference (SciPy): Exploring Network Structure, Dynamics, and Function using NetworkX*. Retrieved December 29, 2020, from http://conference.scipy.org/proceedings/SciPy2008/paper_2/

Hansen, D. L., Shneiderman, B., & Smith, M. A. (2011). Chapter 3 – Social Network Analysis: Measuring, Mapping, and Modeling Collections of Connections. *Analyzing Social Media Networks with NodeXL*, 31–50. https://doi.org/10.1016/B978-0-12-382229-1.00003-5

Javed, M. A., Younis, M. S., Latif, S., Qadir, J., & Baig, A. (2018). Community detection in networks: A multidisciplinary review. *Journal of Network and Computer Applications*, *108*(September 2017), 87–111. https://doi.org/10.1016/j.jnca.2018.02.011

Jiang, Z., Liu, J., & Wang, S. (2016). Traveling salesman problems with PageRank Distance on complex networks reveal community structure. *Physica A: Statistical Mechanics and Its Applications*, *463*, 293–302. https://doi.org/10.1016/j.physa.2016.07.050

Kaur, S., Singh, S., Kaushal, S., & Sangaiah, A. K. (2016). Comparative analysis of quality metrics

for community detection in social networks using genetic algorithm. *Neural Network World*, *26*(6), 625–641. https://doi.org/10.14311/NNW.2016.26.036

Kim, J., Lee, J. G., & Lim, S. (2016). Differential flattening: A novel framework for community detection in multi-layer graphs. *ACM Transactions on Intelligent Systems and Technology*, *8*(2). https://doi.org/10.1145/2898362

Li, X. M., Xu, G., & Tang, M. (2018). Community detection for multi-layer social network based on local random walk. *Journal of Visual Communication and Image Representation*, *57*, 91–98. https://doi.org/10.1016/j.jvcir.2018.10.003

Mishra, S., Hota, C., Kumar, L., & Nayak, A. (2019). An Evolutionary GA-Based Approach for Community Detection in IoT. *IEEE Access*, *7*, 100512–100534. https://doi.org/10.1109/access.2019.2923965

Nascimento, M. C. V. (2014). Community detection in networks via a spectral heuristic based on the clustering coefficient. *Discrete Applied Mathematics*, *176*, 89–99. https://doi.org/10.1016/j.dam.2013.09.017

Newman, M. E. J. (2001). The structure of scientific collaboration networks. *The Structure and Dynamics of Networks*, *9781400841*, 221–226. https://doi.org/10.1515/9781400841356.221

Newman, M. E. J. (2006). *Modularity and community structure in networks*. *103*(23), 8577–8582.

Said, A., Abbasi, R. A., Maqbool, O., Daud, A., & Aljohani, N. R. (2018). CC-GA: A clustering coefficient based genetic algorithm for detecting communities in social networks. *Applied Soft Computing Journal*, *63*, 59–70. https://doi.org/10.1016/j.asoc.2017.11.014

Software for Complex Networks — NetworkX 2.5 documentation. (n.d.). Retrieved December 29,

2020, from https://networkx.org/documentation/stable/index.html

Wang, Q., Li, W., Zhang, X., & Lu, S. (2016). Academic Paper Recommendation Based on Community Detection in Citation-Collaboration Networks. *APWeb*, *2*, 56–67. https://doi.org/10.1007/978-3-319-45817-5

Witten, I. H., Frank, E., Hall, M. A., & Christopher, J. P. (2011). Algorithms: The Basic Methods 4. In *Data Mining, 4th edition* (4th ed., pp. 85–145). https://doi.org/10.1016/B978-0-12-374856-0.00004-3

Yang, Y., Sun, P. G., Hu, X., & Li, Z. J. (2014). Closed walks for community detection. *Physica A: Statistical Mechanics and Its Applications*, *397*(37), 129–143. https://doi.org/10.1016/j.physa.2013.11.034

Žalik, K. R., & Žalik, B. (2018). Multi-objective evolutionary algorithm using problem-specific genetic operators for community detection in networks. *Neural Computing and Applications*, *30*(9), 2907–2920. https://doi.org/10.1007/s00521-017-2884-0