PONTIFICIA UNIVERSIDAD CATÓLICA DE VALPARAÍSO
FACULTAD DE INGENIERÍA
ESCUELA DE INGENIERÍA INFORMÁTICA

DETECCIÓN DE COMUNIDADES EN REDES COMPLEJAS O GRAFOS USANDO METAHEURÍSTICAS

MARÍA PAZ CONTRERAS WARZ

JUAN ALEJANDRO ZÚÑIGA VERDUGO

INFORME FINAL DE PROYECTO
PARA OPTAR AL TÍTULO DE
INGENIERO CIVIL EN INFORMÁTICA.

DICIEMBRE, 2015

PONTIFICIA UNIVERSIDAD CATÓLICA DE VALPARAÍSO
FACULTAD DE INGENIERÍA
ESCUELA DE INGENIERÍA INFORMÁTICA

DETECCIÓN DE COMUNIDADES EN REDES COMPLEJAS O GRAFOS USANDO METAHEURÍSTICAS

INFORME FINAL
PROYECTO 2

MARÍA PAZ CONTRERAS WARZ
JUAN ALEJANDRO ZÚÑIGA VERDUGO

Profesor Guía: **WENCESLAO PALMA MUÑOZ**
Profesor Co-Referente: **IGNACIO ARAYA ZAMORANO**

Carrera: **Ingeniería Civil Informática**

DICIEMBRE 2015

Agradezco a cada de una de las personas que formaron parte de este camino, en especial a mis padres y hermanas por su incondicional apoyo, a Jaime por estar a mi lado siempre, a mi compañero de proyecto Juan que fue un pilar fundamental en el desarrollo de este trabajo y a los profesores guías por iluminar mi camino y a cada profesor que fue parte en mi formación académica. Eternamente agradecida.

- María Paz

A mi familia que siempre estuvieron apoyándome, a mi madre que me motivó a creer en mí y mis capacidades, a mi padre que con su manera de ser fue un ejemplo a seguir y me entregó los mejores valores que mantengo hasta hoy, a mis profesores a lo largo de mi vida académica que fueron una guía en mi desarrollo y me brindaron las herramientas para sacar adelante este desafío. Eternamente agradecido.

.

- Juan

Resumen

En las últimas décadas se ha observado que la estructura de muchas redes reales (redes sociales, tráfico aéreo, redes neuronales) se puede modelar de manera efectiva usando redes complejas, es decir, un conjunto de nodos que se relacionan entre sí de acuerdo a ciertas propiedades topológicas no triviales. Una de las propiedades que poseen los grafos que caracterizan estos sistemas complejos es la estructura de comunidad. La detección de comunidades tiene como objetivo la identificación de los módulos o grupos con alguna o varias propiedades en común. La detección de comunidades es importante no sólo para caracterizar el grafo, sino que además ofrece información sobre la formación de la red, así como sobre su funcionalidad. La estructura de conectividad de estas redes se manifiesta por la presencia de comunidades o grupos, es decir, conjuntos de nodos que comparten propiedades comunes o poseen roles similares en la red.

Este trabajo se centra principalmente en el problema de la detección de comunidades en sistemas complejos o grafos, mediante la aplicación de una metaheuristica basada en enjambres de abejas artificiales. Además, se utiliza la modularidad como guía para la detección de comunidades, y se agrega la detección de comunidades solapadas a través de una extensión de modularidad.

Palabras Clave: comunidad, redes, algoritmo, grafo, metaheuristica, modularidad, Artificial BeeColony

Abstract

Most systems can be represented as complex networks, namely a set of nodes which relate to each other according to certain non-trivial topological properties. One of the properties they own graphs that characterize these complex systems is the structure of community. Detection of communities aims to identify the modules or groups with one or several properties in common. Detection of communities is important not only to characterize the graph, but also provides information on network formation as well as their functionality. The connectivity structure of these networks is shown by the presence of communities or groups, that is, sets of nodes that share common properties or have similar roles in the network.

This work is mainly focused on the problem of detecting communities in complex systems or graphs, by applying a metaheuristic based on artificial swarms of bees. Furthermore, modularity to guide communities detection is used, and detecting overlapping communities through a modular extension is added.

Keywords: community networks, algorithm, graph, metaheuristic, modularity, Artificial BeeColony

Índice

Lista de Figuras

Lista de tablas

1 Introducción

El término comunidad tiene su origen en el vocablo latino *communitas*, se refiere a un conjunto, una asociación o un grupo de individuos, pueden ser de seres humanos, de animales o de cualquier otro tipo de vida, que comparten elementos, intereses, propiedades u objetivos en común, por ejemplo, el idioma, las costumbres, la visión del mundo, el trabajo (empresa), los problemas y/o los intereses. Actualmente encontramos comunidades en las llamadas redes sociales, gracias a las plataformas de internet que facilitan la comunicación de las personas. El aumento exponencial de usuarios de las diferentes redes sociales ha incrementado el interés en el estudio de estas.

Para llevar a cabo el estudio de redes sociales se aplica la llamada teoría de grafos, donde se identifican las entidades como "nodos" o "vértices" y las relaciones como "enlaces" o "aristas". La estructura del grafo resultante es a menudo una red compleja, en su forma más simple una red social es un mapa de todos los lazos relevantes entre todos los nodos estudiados. Al ser una estructura compleja, otro factor relevante es estudiar cuando algún nodo pertenece a más de algún grupo, lo que se conoce como solapamiento.

Las metaheurısticas son esquemas generales de heurísticas que permiten abordar un amplio espectro de problemas adaptándose a sus particularidades. En los últimos años la investigación en metaheurísticas ha crecido en forma sustancial, sustentada en los buenos resultados obtenidos debidos a la aplicación de esas técnicas a problemas de optimización, formulándose una gran cantidad de propuestas de nuevas metaheuristicas. Sin embargo, en la práctica solamente un grupo pequeño de esas propuestas ha logrado consolidarse, demostrando una amplia aplicabilidad sobre problemas de diversas características y adquiriendo la madurez necesaria para ser una alternativa real al momento de resolver un problema de optimización.

El diseño de algoritmos cada vez más eficientes para resolver problemas complejos (tanto de optimización como de búsqueda) ha sido tradicionalmente uno de los aspectos más importantes de la investigación en Informática. El objetivo perseguido en este campo es, fundamentalmente, el desarrollo de nuevos métodos capaces de resolver los mencionados problemas complejos con el menor esfuerzo computacional posible, mejorando así a los algoritmos existentes. En consecuencia, esto no sólo permite afrontar problemas actuales de forma más eficiente, sino también tareas vedadas en el pasado debido a su alto coste computacional. Finalmente se presenta en este informe la solución aplicada para la detección de comunidades usando un algoritmo basado en el comportamiento de enjambres de abejas. En este informe presentamos métodos de búsqueda tanto para comunidades disjuntas como también para comunidades solapadas.

2 Objetivos

2.1 Objetivos Generales

Estudiar e implementar una metaheurística basada en enjambres de abejas (Algoritmo ABC) para la detección de comunidades solapadas en redes complejas.

2.2 Objetivos Específicos

- Generar conocimiento en base a la investigación de trabajos previos similares para así comprender de mejor manera la problemática.
- Desarrollar un método para la detección de comunidades solapadas en redes complejas basados en el algoritmo ABC.
- Implementar un algoritmo basado en el algoritmo ABC que detecte comunidades solapadas en redes complejas.
- Realizar pruebas para comprobar rendimiento y contrastar la performance con otros métodos.

3 Comunidades

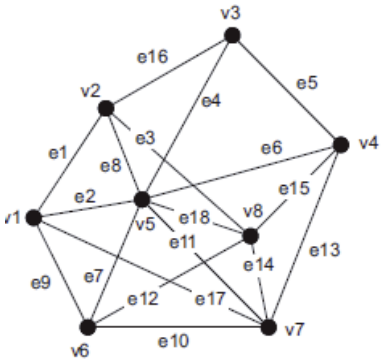
3.1 Elementos de Teoría de Grafos para Comunidades

Los grafos constituyen una herramienta básica para modelar fenómenos discretos y son fundamentales para la comprensión de las estructuras de datos y el análisis de algoritmos. En matemáticas e informática, la teoría de grafos estudia las propiedades de los grafos, que son colecciones de objetos llamados vértices (o nodos) conectados por líneas llamadas aristas (o bordes) que pueden tener orientación (dirección asignada). Típicamente, un grafo está diseñado por una serie de puntos (los vértices) conectados por líneas (las aristas). Utilizando una notación formal, un grafo se define como sigue:

Definición 3.1.: un grafo **G** consiste de una colección de V vértices y de una colección de E aristas, denotándose como G= (V, E). Cada arista e ∈ E, se dice que une dos vértices, que son denominados como **puntos extremos**. Si *e* es el enlace *u, v* ∈ V, entonces se escribe como *e*= ‹*u,v*›. Los vértices u y v en este caso se dice son adyacentes. La arista e se dice que es incidente con los vértices u y v respectivamente.

De forma general se escribe **V (G)** y **E(G)** para denotar el conjunto de vértices y aristas asociados al grafo G. Un grafo que no tiene enlaces o aristas se denomina **grafo simple**, de la misma manera un grafo que no contenga vértices se le denomina **grafo vacío**, en contra parte un grafo que tenga todos los vértices enlazados se dice que es un **grafo completo**. Un grafo completo de *n* vértices, comúnmente se denota como k_n [1].

Con esta definición básica de un grafo se puede empezar a tener más claridad de cómo están construidas las comunidades. También se debe tener en cuenta los conceptos de secuencia de grado, matriz de adyacencia, grado de un vértice y clique de un grafo ente otros [1].



$$V(G) = \{v_1, \ldots, v_8\}$$
$$E(G) = \{e_1, \ldots, e_{18}\}$$

$e_1 = \langle v_1, v_2 \rangle$	$e_{10} = \langle v_6, v_7 \rangle$
$e_2 = \langle v_1, v_5 \rangle$	$e_{11} = \langle v_5, v_7 \rangle$
$e_3 = \langle v_2, v_8 \rangle$	$e_{12} = \langle v_6, v_8 \rangle$
$e_4 = \langle v_3, v_5 \rangle$	$e_{13} = \langle v_4, v_7 \rangle$
$e_5 = \langle v_3, v_4 \rangle$	$e_{14} = \langle v_7, v_8 \rangle$
$e_6 = \langle v_4, v_5 \rangle$	$e_{15} = \langle v_4, v_8 \rangle$
$e_7 = \langle v_5, v_6 \rangle$	$e_{16} = \langle v_2, v_3 \rangle$
$e_8 = \langle v_2, v_5 \rangle$	$e_{17} = \langle v_1, v_7 \rangle$
$e_9 = \langle v_1, v_6 \rangle$	$e_{18} = \langle v_5, v_8 \rangle$

Figura 3.1 Un ejemplo de un gráfico con ocho vértices y 18 bordes

3.2 Definición general de Comunidades

La idea intuitiva utilizada en la gran mayoría de los trabajos para establecer qué es una comunidad, es que los vértices de la misma deben estar más relacionados entre sí, que con el resto de los vértices de la red. En función de esta idea general se han propuestos numerosos criterios cuantitativos para definir que es una comunidad. Estos criterios pueden clasificarse en:

- Definiciones locales: se analiza la estructura interna de la comunidad, sin tener en cuenta el resto de la red.
- Definiciones globales: se analiza el papel de la comunidad en la estructura global de la red.

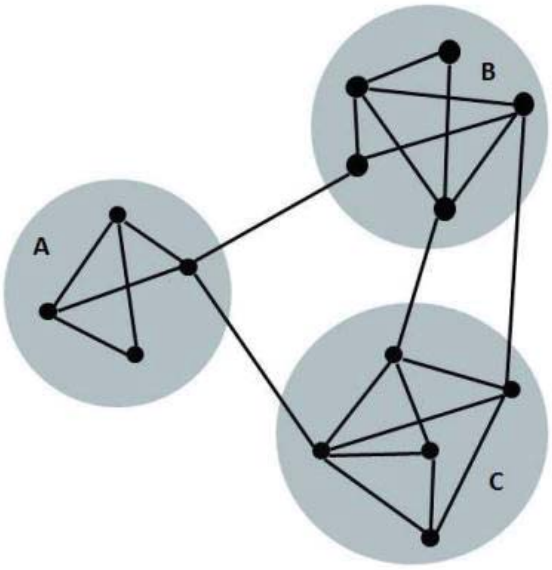


Figura 3.2 Grafo G en donde $k(G) = 2$; $\lambda(G) = 3$; $\delta(G)=4$.

3.3 Comunidades Solapadas

Como ya se definió, una comunidad en una red es un sub grupo de esta, el cual está más densamente conectado entre sí, que con el resto de la red, cuando un nodo de esta red pertenece a más de un sub-grupo a la vez se dice que la red está solapada.

La definición de solapamiento es válida y es precisamente lo que vemos en la realidad. Por ejemplo, vemos en una red social como Facebook en donde una persona cualquiera (la cual representaría un nodo) puede pertenecer a varias comunidades o grupos si tomamos en cuenta sus intereses.

De forma intuitiva podemos deducir que las personas podrían llegar a tener más de un interés, pero en este caso no podríamos asignar cada vértice (persona) a una sola comunidad, es acá donde aparece el termino de **comunidad solapada**, en donde estamos en presencia de comunidades que comparten nodos entre sí [2], es decir, vértices que pertenecen a dos o más comunidades, ver Figura 3.

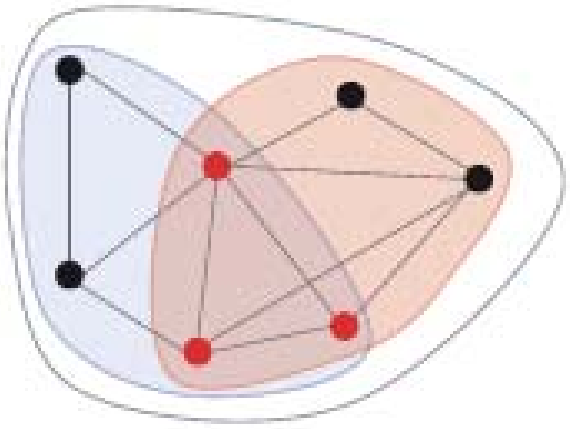


Figura 3.3 Visualización de 2 comunidades con 3 nodos solapados

4 Métodos de detección de Comunidades

4.1 Métodos divisivos

La filosofía de este tipo de métodos consiste en eliminar las aristas que conectan vértices pertenecientes a distintas comunidades, de esta manera quedan aisladas unas de otras. El problema de partición de grafos consiste en dividir los nodos de un grafo ***G(N , A)*** en ***p*** grupos de un tamaño similar y predefinido, mientras se minimiza el número de arcos entre los grupos. En la Figura 5.2 se muestra un clásico ejemplo donde $p = 2$. Los arcos entre los grupos se llaman "tamaño del corte".

Debido a si simpleza un gran número de heurísticas han sido propuestas, entre las primeras y las más conocidas se encuentra el algoritmo de Kernighan-Lin el cual busca mejorar una partición inicial, en general, buscada con otro método, optimizando la diferencia entre los arcos en las particiones y entre ellas. El método explora entre los vecinos buscando mejorar esta diferencia.

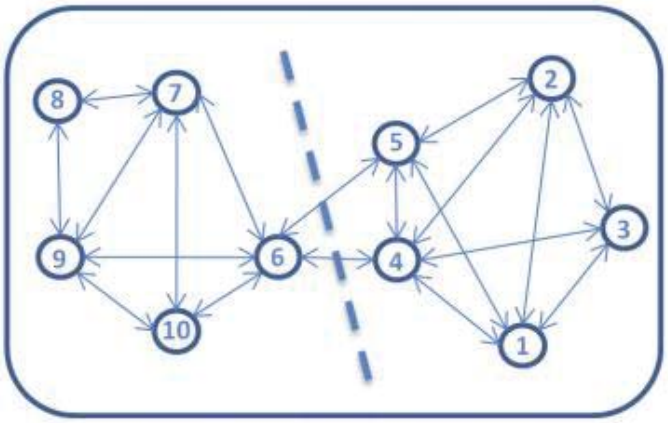


Figura 4.1Método Divisivo

En general el algoritmo de Kernighan-Lin es usado con otras estrategias para optimizar los resultados de búsqueda de particiones o grupos. Una conocida estrategia para encontrar particiones son los métodos de bisección. Estos buscan separar iterativamente la red en 2 subgrafos y luego repetir el procedimiento con cada sub grafo hasta completar el número de grupos solicitado a priori. Para separar los vértices en cada iteración se utilizan los vectores propios de la matriz Laplaciana.

El gran problema de estos métodos es que se debe entregar de antemano el número de grupos y en general esto es una de las interrogantes del problema. Luego estas estrategias resultan no ser del todo adecuadas para la búsqueda de comunidades.

4.2 Métodos de Clustering

Este tipo de métodos busca determinar cómo separar adecuadamente un conjunto de nodos respecto a un número predefinido de clusters o grupos. Para poder utilizar este tipo de métodos es necesario embeber el grafo bajo una métrica espacial, de tal manera de que cada nodo se encuentre a una distancia respecto a cada uno del resto de nodos que componen el grafo. Esta distancia puede ser una medida de similitud o de disimilitud. Luego el problema consiste en, dado un número *k* de clusters, colocar a cada uno de los nodos en uno de los *k* grupos maximizando o minimizando una función de costos basados en la distancia de los nodos a ciertos centroides o puntos en el espacio que representan a estos *k* grupos.

Existen diversas funciones que pueden ser utilizadas para encontrar los clusters, entre ellas, las más conocidas son k-means y Fuzzy c-means que si bien apuntan al mismo objetivo, se diferencian en que esta última otorga a cada nodo un grado de pertenencia a cada cluster, pudiendo pertenecer, según el nivel de corte exigido, a más de una comunidad al mismo tiempo, cosa que en k-means no es posible. Sin embargo, al igual que los métodos de partición de grafos requieren que el número de grupos o comunidades sea definido de antemano.

4.3 Métodos Jerárquicos

Este tipo de métodos tiene por objetivo buscar las divisiones naturales de la red en grupos, basados en la idea de que el grafo tiene una estructura jerárquica, es decir, pequeños grupos de nodos que son parte de grupos medianos de nodos y que a su vez éstos pertenezcan a grupos más grandes y así sucesivamente.

Uno de los más recientes métodos es el propuesto por Girvan y Newman. Este método usa como medida la *betweenness* de un arco, la cual se define como la suma de todos los caminos mínimos que existen entre todos los pares de nodos de la red que pasan por este arco. La finalidad es remover iterativamente los arcos con mayor *betweenness* hasta que no quede ningún arco o hasta que se cumpla un criterio de detención adecuado para el problema en particular. La gran desventaja de este algoritmo es que resulta poco eficiente, pues se hace necesario recalcular los caminos mínimos entre los nodos que pasan por el arco retirado en cada iteración.

A pesar que no es necesario definir a priori el número de grupos se presentan ciertas desventajas:

En primer lugar los métodos aglomerativos tienden solo a encontrar el core de las comunidades y a descuidar los nodos periféricos debido a la forma en que operan.

Otro problema de los métodos jerárquicos es que no proveen de una medida que permita determinar cuáles de las soluciones entregadas es mejor y dependerá de la medida de similitud utilizada cuál o cuáles son las mejores soluciones. Para resolver esto Newman y Girvan proponen la “modularidad”.

4.4 Métodos basados en la Modularidad

Según señala Fortunato la modularidad representa uno de los primeros intentos por lograr entender los principios del problema de clustering, integrando en su función de calidad todos los elementos esenciales, desde la definición de comunidad, pasando por la elección de un modelo nulo de comparación, hasta la expresión de solidez o fortaleza de las comunidades y particiones encontradas.

Dado que la modularidad representa una función que refleja qué tan buena es una partición de una red, entendiendo que entre mayor sea el valor, mejor es la partición de la red encontrada, parece ser una buena estrategia buscar maximizar esta función. Sin embargo, se ha determinado que, dado la gran cantidad de posibles particiones de una red, el problema de maximizar la modularidad es del tipo NP-Completo. Así resulta natural pensar en heurísticas para poder lograr valores aproximados al óptimo de modularidad, en particular, técnicas basadas en algoritmos voraces (greedy).

Uno de los primeros intentos en esta área fue el método voraz de Newman, posteriormente este algoritmo fue mejorado por Clauset quien realizó cambios en la manera en que era calculada la modularidad entre los grupos, eliminando todas las operaciones innecesarias y mejorando la elección de la máxima modularidad en cada iteración utilizando max-Heap.

Otro método voraz conocido es el propuesto por Blondel et al. Este método funciona en dos etapas: En la primera se busca encontrar la máxima modularidad con una estrategia local y en la segunda, para mejorar los resultados, se itera de la misma manera pero embebiendo las comunidades en un solo nodo.

El algoritmo voraz de Blondel, también conocido como algoritmo de Louvain, tiene mejores resultados en términos de modularidad máxima encontrada que el algoritmo de Clauset y sus mejoras. Aun así, el resultado obtenido con este método depende significativamente del orden de los nodos escogidos. El método en detalle será explicado en siguiente sección.

5 Índices de calidad para evaluar y detectar comunidades

Al momento de intentar detectar comunidades en alguna red en particular, se debe considerar que los métodos propuestos hasta hoy en día tienen su base en alguna métrica para determinar la calidad de las comunidades detectadas.

5.1 Índices de calidad en la detección de comunidades disjuntas

5.1.1 Modularidad

La función de calidad más popular se denomina Modularidad y fue propuesta por Newman y Girvan en 2004 [3]. Se basa en la idea de que una distribución en clusters no es lo que se espera por azar en una red, y por tanto, trata de cuantificar la intensidad de esta estructura de comunidades comparando la densidad de link dentro y fuera de cada comunidad con la densidad que esperaríamos si los links estuviesen distribuidos aleatoriamente en la red. A este modelo estadístico se le debe dotar de un modelo nulo que especifique qué es lo que se espera por azar. En este caso, los autores parten de la base de que la distribución de grados de los nodos es una propiedad intrínseca de la red y proponen un modelo nulo siguiendo este principio. La siguiente fórmula es la utilizada para calcular la Modularidad (Q) de una partición determinada (P):

$$Q(P) = \frac{1}{2m} \sum_{ij} \left[A_{ij} - \frac{d_i d_j}{2m} \right] \sigma(C_i, C_j) \qquad (1)$$

En donde:

- A_{ij} : Es la matriz de adyacencia del grafo
- d_i : Es el grado del nodo *i*
- d_j : Es el grado del nodo *j*
- m : Numero de aristas o links
- $\frac{d_i d_j}{2m}$: Según el modelo nulo, corresponde al número esperado de aristas entre los nodos *i* y *j*
- La función $\boldsymbol{\delta}$ corresponde a una función binaria que toma como valor uno si es que los nodos *i* y *j* están en la misma comunidad (C_i=C_J) y cero en caso contrario.

Dado que los únicos pares de vértices que aportan a la sumatoria son los que aparecen en una misma comunidad, la expresión anterior se puede reagrupar de manera que, en vez de analizar el aporte puntual de cada par de vértices, la modularidad de una partición queda definida mediante la suma de la modularidad de sus grupos. Esto se logra mediante la expresión de modularidad (equivalente a la anterior):

$$Q(P) = \sum_{C \in P} \left[\frac{E_C}{m} - \left(\frac{d_{total(C)}}{2m} \right)^2 \right] \qquad (2)$$

Una característica muy útil de la modularidad es que no solamente indica qué tan buena es una partición mediante valores positivos, sino que también puede describir qué tan malo es un agrupamiento mediante valores negativos. Una partición con valores negativos implica la existencia de grupos de baja densidad interna que interactúan mucho más con el resto de la red. Si en una red no hay particiones con modularidad positiva, entonces se asume que dicha red no posee una estructuración basada en comunidades. Para cualquier red, existe una partición trivial con modularidad igual a cero, que resulta de considerar toda la red como una única comunidad.

En la definición inicial de modularidad se requiere que cada vértice pertenezca a una única comunidad, por lo que no se puede evaluar el solapamiento. Otro de los problemas es el límite de resolución, tal como Fortunato y Barthélemy demostraron matemáticamente, la optimización de esta medida sufre de un límite de resolución. Esto significa que es incapaz de detectar comunidades de tamaño menor que un límite que viene determinado por el número de links del grafo y su patrón de conexiones.

5.1.2 NMI

La Información mutua normalizada o NMI [4], mide los resultados en relación a los que deberían haber sido. En grafos sintéticos se puede manejar como debería estar organizada la red, desde la cantidad de vértices y aristas, factor de mezclado hasta como se estructuran las comunidades, NMI mide esta información contrastando los resultados de algoritmo versus los resultados que deberían haber sido según la configuración del grafo sintético, por lo tanto, si X es la partición detectada e Y la partición conocida a priori, la comparación por medio la NMI seria:

$$I_{norm}(X,Y) = \frac{2(H(X)+H(X|Y))}{H(X)+H(Y)} \qquad (3)$$

Dónde:

$$H(X) = -\sum_x P(X) log P(X) \qquad (4)$$

$I_{norm}(X,Y)$ es igual a 1 si las particiones son idénticas e igual a 0 si son independientes.

5.2 Índices de calidad en la detección de comunidades solapadas

5.2.1 Extensión de Modularidad.

En el caso de las comunidades de la superposición, un vértice puede pertenecer a más de una comunidad. La fuerza de su apego a cada comunidad puede ser diferente en función del número de conexiones que tienen con cada comunidad. Sobre la base de esta observación, la definición original de la modularidad para evaluar las comunidades superpuestas se extiende como:

$$Q_0 = \frac{1}{2m}\sum_n \sum_{i \in c_n, j \in c_n} \left[\frac{k_i^c k_j^c}{k_i\, k_j}\right]\left[\frac{a_{ij}}{k_i k_j} - \frac{1}{2m}\right] \qquad (5)$$

Donde $k_i^c = \sum_{p \in V_n} a_{ip}$ y $k_j^c = \sum_{p \in V_n} a_{jp}$ y V_{c_n} es el conjunto de vértices en la comunidad c_n. De manera similar a la modularidad tradicional Q, la modularidad extendida propuesta Q_0 = 0 cuando todos los nodos pertenecen a la misma comunidad y obtiene un valor más alto para indicar una estructura de comunidad más fuerte [5].

6 Algoritmos para la detección de comunidades disjuntas

A continuación, se presentan algoritmos altamente efectivos a la hora de realizar la compleja labor de detectar comunidades. Se dividirán tanto en redes disjuntas como en redes solapadas.

6.1 Algoritmo de Blondel.

También conocido como Louvain [20] es una de las técnicas más eficientes entre las basadas en optimización de modularidad. Este algoritmo es capaz de procesar redes ponderadas y establecer una jerarquía entre las comunidades detectadas. La estructura general de Louvain consta de dos fases principales, de manera que la primera es la encargada de formar las particiones y en la segunda se procesan las comunidades obtenidas previamente para identificar sus relaciones jerárquicas.

Inicialmente, cada nodo de la red constituye una comunidad. Luego, se lleva a cabo un proceso iterativo donde, en cada paso se sigue una estrategia greedy para reajustar la partición detectada en el paso anterior. Con este objetivo, se recorren de manera aleatoria todos los vértices y se actualiza la comunidad asignada a cada uno de ellos. Para actualizar la comunidad de un vértice, se verifica si moverlo a la comunidad de alguno de sus vecinos, incrementa la modularidad de la partición. En tal caso, se realiza el movimiento que reporte el mayor aumento de modularidad. En caso contrario, el vértice sigue perteneciendo a la misma comunidad. Este proceso termina cuando las particiones obtenidas en dos iteraciones sucesivas sean iguales.

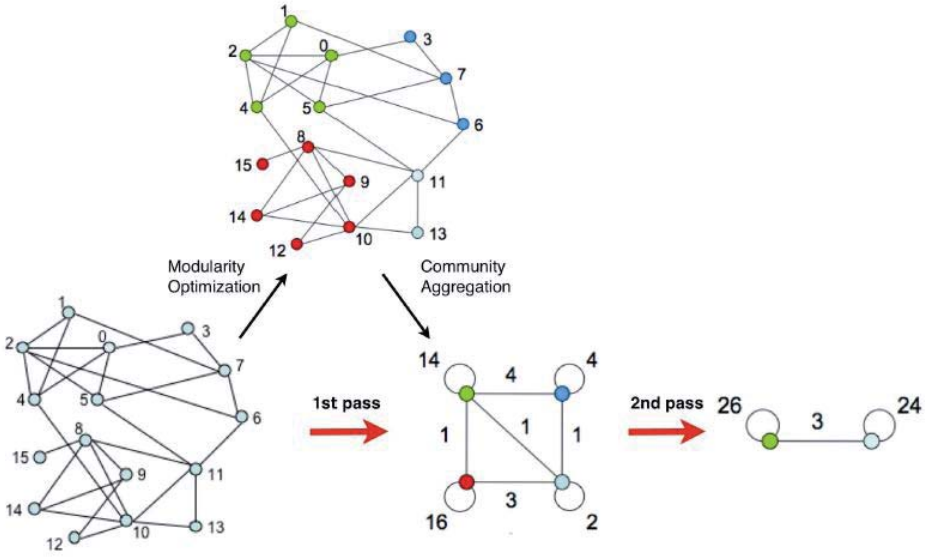


Figura 6.1 Fases Algoritmo de Blondel

7 Algoritmo para la detección de comunidades solapadas

7.1 FOCS (Búsqueda de estructuras de comunidades solapadas)

A continuación, se presenta el algoritmo FOCS [19] y sus correspondientes etapas, pero antes es necesario definir la función de densidad, la cual es usada como métrica de calidad en el núcleo del algoritmo para ir aceptando o no las comunidades.

7.1.1 Función de Densidad

Con el fin de cuantificar la bondad de una comunidad identificada, es decir, medir la proporción de aristas internas a la comunidad C respecto al total de sus posibles aristas internas, se utiliza la conocida función de densidad ψ definida como:

$$\psi(C) = \frac{|C^{in}|}{\binom{|C|}{2}} \; donde \; C \subseteq V \qquad (6)$$

Donde C^{in} es la cantidad de aristas internas de la comunidad C y |C| es la cantidad de nodos existentes. Mientras más se aproxima C a un clique de su tamaño, mayor es su valor de densidad $\psi(C)$. Para que C se considere una comunidad local se propone la función

$$\tau(C) = \frac{\sigma(C)}{\binom{|C|}{2}} \; donde \; \sigma(C) = \binom{|C|}{2}^{1-\frac{1}{\binom{|C|}{2}}} \qquad (7)$$

Aquí $\sigma(C)$ es el umbral del número de aristas internas que basta para que C se considere una comunidad local. Particularmente, un subgrafo inducido por C es una comunidad local si y sólo si $\psi(C) \geq \tau(C)$ o equivalentemente $\left|C^{in}\right| \geq \sigma(C)$. Las funciones $\tau(C)\; y \; \sigma(C)$ son procedimientos locales en la comunidad candidata C por lo que no se requiere de ningún parámetro de entrada por parte del usuario, además por la Proposición 1 dichas funciones van en aumento.

Proposición 1. La función $f(n) = n^{1-\frac{1}{n}}$ es estrictamente creciente para $n \geq 4 \; y \; \lim_{n\to\infty} f(n) = n$.

7.1.2 Búsqueda de estructuras de comunidades solapadas

Así como son variados los criterios para definir una comunidad, también lo son las técnicas y algoritmos que intentan hacer una correcta detección de las comunidades. Por un lado hay algoritmos que se enfocan en agrupar las comunidades sin considerar posible solapamiento y también están los que se enfocan en buscar solapamiento en las

comunidades detectadas, ya sea buscando este en el momento de detectarlas [1] o una ves ya detectadas, con distintas técnicas intentar ve la relación entre ellas y combinarlas según la información que tengan en común.

En relación al segundo caso, en donde se determina el solapamiento una vez ya detectadas las comunidades, uno de los algoritmos efectivos en realizar esta tarea es el llamado FOCS, de la sigla en inglés Búsqueda de estructura de comunidades solapadas.

FOCS consta de tres etapas fundamentales, la primera es en donde se detectan las comunidades usando la función de densidad previamente definida, en la segunda etapa se combinan las comunidades según la información las aristas en común y por último se revisa cualquier nodo que no se haya logrado agregar a alguna comunidad y así asignarlo. Estos tres procesos se detallan a continuación.

7.1.3 Búsqueda de comunidades locales

Las comunidades locales conectan partes de la red cuyas densidades internas son mayores o al menos iguales a cierto parámetro. FOCS determina este parámetro de manera automática vasado en la función $\tau(.)$. Particularmente, una comunidad local se define sobre la base de la conexión (u, v) cuando el número de conexiones internas dentro del subgrafo inducido por$C = \{u, v\} \cup \big(N(u) \cap N(v)\big) excede\ \sigma(C)$, o en otras palabras cuando $\psi(C) \geq \tau(C)$. Sin embargo, eventualmente podría surgir un problema durante este procedimiento: la existencia de subcomunidades en una comunidad más grande. En la práctica, se detectan comunidades grandes unificadas por comunidades más pequeñas (subcomunidades) si la comunidad grande es en sí densamente conectada. Con el propósito de mitigar este caso no deseado, se impone que $\psi(\cup_{i=1}^{s} C_i) < \tau(\cup_{i=1}^{s} C_i) \forall s = 1 \ldots |C|$.

FOCS detecta aquellas comunidades locales compuestas por cuatro o más nodos, identificando comunidades pequeñas al final del proceso de búsqueda.

El Algoritmo 1 describe el procedimiento de búsqueda, en donde su tiempo de complejidad es del orden de $O(dM)$ con $d = max_{v \in v}\, d_v$.

Algoritmo 1 Búsqueda de comunidades locales

Input: $G = (V, E)$
Output: Una Colección de comunidades Ω_r

```
for (u, v) ∈ E do
  if Com(u) ∩ Com(v) = ∅ then
    Dejar que C = {u, v} ∪ N(u) ∩ N(v);
    if |C^in| ≥ σ(C) and |C| ≥ 4 then
      Definir C como una comunidad local;
      /*Se incluye C dentro de la estructura de comunidades*/
      Ω_r = Ω_r ∪ {C};
    end if
  end if
end for
```

Figura 7.1 Búsqueda de comunidades locales

7.1.4 Combinación de comunidades solapadas

Una vez realizado el proceso de detección de comunidades, la estructura de comunidad del grafo se representa como una colección de (posiblemente solapadas) partes densas de la red junto con los valores atípicos encontrados. Es posible que alguna de esas partes densas puedan compartir subestructuras importantes, por lo que deben unirse si es que están muy solapadas entre sí. Para lo anterior, se propone una función de puntuación de solapamiento para dos comunidades, definida como:

$$OS(C_i, C_j) = \frac{|I_{ij}|}{\min\{C_i, C_j\}} + \frac{I_{ij}^{in}}{\min\{C_i^{in}, C_i^{in}\}} \quad (8)$$

Donde $I_{ij} = C_i \cap C_j$. La función $OS(C_i, C_j)$ valora la importancia de nodos y aristas en común entre C_i y C_j. EN comparación con la métrica de distancia que se sugiere en la ecuación. La función OS(.,.) no solo tiene en consideración la fracción e nodos en común, si no también considera la cantidad de aristas en común entre dos comunidades lo que se vuelve esencial para su combinación. En este proceso, las comunidades C_i y C_j se fusionan si $OS(C_i, C_j) \geq \beta$, donde β es el umbral de solapamiento.

Algoritmo 2 Combinacón de comunidades solapadas

Input: Estructura de comunidades Ω_r
Output: Una refinada estructura de comunidades Ω_f

```
1: Ω_f ← Ω_r
2: for C_i, C_j ∈ Ω_r do
3:    if OS(C_i, C_j) ≥ β = ∅ then
4:       C ← Combinar C_i y C_j;
5:       /*Actualizar la estructura actual*/
6:       Ω_f = (Ω_f {C_i ∪ C_j}) ∪ C;
7:       done ← false
8:    end if
9: end for
```

Figura 7.2 Combinación comunidades solapadas

7.1.5 Revisión de vértices no asignados

En la primera etapa del algoritmo puede que algunos nodos no hayan sido asignado a ninguna comunidad, esto puede ser debido a la poca relación que estos tienen con el resto, o porque son grupos muy pequeños (menos a 4). Si algún nodo no fue considerado en la etapa inicial, inevitablemente no se tratara este en la segunda etapa, por lo cual en una tercera etapa se trata todos estos vértices no asignados con la función fitness ya antes utilizada en la ecuación de combinación. En caso de haber nodos aun sin asignar, estos son agrupados y catalogados como nodos atípicos.

$$F_s = \frac{|S^{in}|}{2\,|S^{in} + S^{out}|} \; Donde\; S \subseteq V. \quad (9)$$

7.2 Estructuras Jerárquicas y Extensión de la Modularidad

Este método consta de un algoritmo de aglomeración de dos fases, la primera para encontrar la estructura jerárquica y luego para determinar la superposición de la comunidad en una red. Este algoritmo se basa en dos conceptos fundamentales: máximo cliqué y la modularidad extendida. Un clique puede ser identificado como un subconjunto de vértices en una red de tal manera que cada dos vértices en el subconjunto están conectados por un borde. Un máximo cliqué es un cliqué que no es un subconjunto de cualquier otro cliqué.

En la primera fase del algoritmo, encuentra todos los máximos cliqués en la red. Aquí, se utiliza el algoritmo de Bron-Kerbosch que se basa en un procedimiento recursivo backtracking [9]. Aquí se proporciona un conjunto de cliqués máximos para una red determinada. Hay que señalar que un solo vértice se puede incluir en varios cliqués máximos, se denominan a los vértices como solapados.

En este caso, no todos los máximos cliqués se tienen en cuenta. Si un cliqué máximo comparte sus vértices de algunos otros clique máximos, los eliminamos. (Por ejemplo: Si {1, 2, 3, 4, 5}, {5,6, 7, 8}, y {2, 3, 5, 7} son tres máximos cliqués en una red dada, se descartaría { 2, 3, 5, 7} como todos sus vértices están ya incluidos en los dos primeros cliqués. En la implementación, los máximos cliqués se almacenan en un arreglo ordenado en orden basado en el número de vértices en cada cliqué de forma descendente. Luego, se itera a través de la matriz y descarta ciertos cliqués en base a los criterios antes mencionados. Esta primera fase reduce considerablemente el tamaño del problema para la segunda fase del algoritmo.

En la segunda fase del algoritmo, los máximos cliqués generados en la fase anterior se consideran como las comunidades iniciales para la detección de la estructura de la comunidad jerárquica de la red. De manera similar al algoritmo introducido en [8], estas comunidades iniciales se unen entre sí en pares de tal manera que resulta un mayor incremento o disminución más pequeña en la modularidad de la red, luego se obtiene un dendrograma. En contraste con el algoritmo se explica en [8], aquí no se inicia desde los vértices únicos, además trata de maximizar Q_0 en lugar de P. Si una comunidad se puede representar como un subconjunto de otra comunidad que se genera al unirse a otras dos comunidades, la comunidad subconjunto también se absorbe en la comunidad en general. (Por ejemplo: Si {1, 2, 3, 4, 5, 6} se genera mediante la unión comunidades {1, 2, 3} y {4, 5, 6} juntos, y si hay otra pequeña comunidad {2, 3 , 6}, entonces serán absorbidos en {1, 2, 3, 4, 5, 6}.) El nivel de corte del dendrograma se decide de acuerdo con el valor de $Q_{0.}$

8 Algoritmo Artificial BeeColony

Dentro de las ciencias de la computación y de la investigación de operaciones, el algoritmo de Colonias de Abejas Artificiales (Artificial BeeColony) es una metaheurística, propuesta por Dervis Karaboga en el año 2005, la cual se basa en el comportamiento inteligente de búsqueda de fuentes de alimento de las colonias de abejas.

En el algoritmo ABC, el espacio de búsqueda se simula como la búsqueda de alimento en el medio ambiente, donde cada punto en el espacio de búsqueda corresponde a una fuente de alimento que las abejas artificiales podrían explotar. La posición de una fuente de alimento representa una posible solución al problema, y la cantidad de néctar que ésta posea se asocia con la calidad (fitness) de dicha solución.

Dentro de la colonia de abejas, existen tres tipos de abejas: Abejas Empleadas, Espectadoras y Exploradoras. Las primeras vuelan sobre la fuente que están explotando y retornan a la colmena para compartir la información recogida sobre la cantidad de néctar en el área de baile. Las abejas espectadoras esperan en la colmena y eligen una fuente para explotar basándose en el baile realizado por las abejas empleadas. Las abejas empleadas cuya fuente de alimento ha sido abandonada se convierten en abejas exploradoras y buscan, aleatoriamente, nuevas fuentes de alimento en el entorno. Esto se controla por medio de un parámetro denominado "límite" [4], el cual es único parámetro de control aparte de los comúnmente utilizados en los algoritmos basados en poblaciones, tales como son el tamaño de la población o colonia (ColonySize, CS) y el número máximo de iteraciones (MaximumCycleNumber, MCN).

En el modelo del algoritmo ABC propuesto por Karaboga [6], la mitad de la colonia se compone de abejas empleadas, mientras que la otra mitad corresponde a abejas espectadoras. Además, se asume que cada fuente de alimento es explotada por solo una abeja empleada. En otras palabras, el número de fuentes de alimento es igual a la cantidad de abejas empleadas, que a su vez es igual a de abejas espectadoras.

En el algoritmo ilustrado en la figura 8.1, se muestra el pseudocódigo correspondiente al algoritmo de colonias de abeja. En esta se puede apreciar con mayor detalle las fases que lo componen, las cuales se explican de manera detallada a continuación.

8.1 Fase de Inicialización

Para comenzar con el algoritmo, se generan las fuentes de alimento correspondientes a las soluciones en el espacio de búsqueda. Dichas fuentes se producen al azar dentro de un rango de límites de variables, por medio de la ec (10):

$$x_{ij} = x_{ij}^{min} + \ rand(0,1) * \left(x_{ij}^{max} - x_{ij}^{min}\right), i \ \in [1, FS], j \in \ [1, Dim] \qquad (10)$$

Donde *FS* es el número de fuentes de alimento y *j* corresponde a la dimensión de la solución $x_i Dim$ es la cantidad de variables a optimizar, y por último,x_j^{max} y x_j^{min} son los límites superiores e inferiores, respectivamente, del *j-ésimo* parámetro de la solución *i*.

Una vez generadas las *FS* fuentes de alimento se determina el fitness [2] de cada una de ellas utilizando la ecuación (5), donde es el valor de costo de la solución por maximizar o minimizar.

$$fitness_i = \begin{cases} \frac{1}{1+f_i}, & si\ f_i \geq 0 \\ 1 + abs(f_i), & si\ f_i < 0 \end{cases} \qquad (11)$$

Para finalizar esta primera etapa, los contadores de intentos de abejas por encontrar una fuente de alimento con mejor fitness que la que ya poseía en su memoria, se inicializan en cero.

8.2 Fase Abejas Empleadas

En esta fase, cada abeja empleada intenta mejorar su solución modificando una de sus dimensiones, por medio de la ecuación (12):

$$v_{ij} = x_{ij} + \emptyset(x_{ij} - x_{kj}) \qquad (12)$$

Donde k es una fuente de alimento seleccionada al azar distinta de *i*, *j* es una dimensión seleccionada aleatoriamente y ∅es un número aleatorio distribuido uniformemente dentro del rango [-1,1].

Luego se calcula el fitness de la fuente de alimento modificada, v, y se aplica una selección codiciosa entre la nueva fuente y la que ya existía en su memoria. Si el fitness de la nueva solución candidata es mejor que la antigua, entonces la abeja empleada memorizará la posición de la nueva fuente y su contador de intentos se reseteará a cero. En caso contrario, la abeja empleada desechará la nueva fuente, manteniendo en su memoria la posición de la fuente anterior y su contador de intentos se incrementará en uno.

Una vez que todas las abejas empleadas han completado el proceso de búsqueda de mejores fuentes de alimento, vuelven a la colmena y comparten la información de la calidad de sus fuentes por medio de un baile.

8.1 Fase Abejas Espectadoras

Las abejas espectadoras reciben la información de las fuentes de alimento compartida por las abejas empleadas y escogen una basándose en una probabilidad asociada a la cantidad de néctar que ésta posea. Debido a como se lleva a cabo selección, una o más abejas espectadoras podrían optar por la misma fuente de alimento si esta posee un buen fitness. La probabilidad de selección de la fuente de alimento se calcula por medio de la ecuación (13):

$$P_i = 0{,}9 * \frac{fitness_i}{fitness_{best}} + 0{,}1 \qquad (13)$$

Donde $P_i$es la probabilidad de optar por la solución de la *i-esima* abeja empleada, $fitness_i$ es la calidad asociada a dicha fuente de alimento, y a, $fitness_{best}$es la mejor solución encontrada dentro de la población.

Para escoger una fuente, las abejas espectadoras utilizan un método de selección al azar [12], el cual ofrece mejores candidatos para tener una mayor probabilidad de ser seleccionadas. Luego de elegir la fuente de alimento, al igual que en la fase anterior, cada abeja intenta mejorar su solución por medio de la ecuación (12) y aplica una selección codiciosa entre la nueva fuente y la original con el fin de memorizar la fuente de mejor calidad. El contador de intentos se incrementará en uno si selecciona la nueva fuente o se reiniciará a cero si se escoge la fuente anterior.

8.2 Fase Abejas Exploradoras

Por último, si una fuente de alimento ha sido abandonada por una abeja empleada o espectadora, la abeja empleada de dicha fuente se convierte en abeja exploradora y sale en busca, aleatoriamente, de una nueva fuente de alimento. Para determinar si una fuente fue abandonada, al final de cada iteración se evalúa el contador de intentos de cada solución. En otras palabras, si el valor del contador de intentos por mejorar una fuente de alimento es mayor que el parámetro conocido como "límite" [4], entonces se reemplaza dicha fuente por una nueva utilizando la ecuación (10), y el contador de intentos de aquella abeja se reseteará a cero.

Cabe destacar que las fases de las abejas empleadas, espectadoras y exploradoras se repetirán mientras no se alcance la condición de término. Además, por cada iteración se almacena la mejor solución obtenida hasta el momento, por lo que al cumplirse la condición de término y concluir el ciclo iterativo, el algoritmo da como resultado inmediato, la mejor solución encontrada.

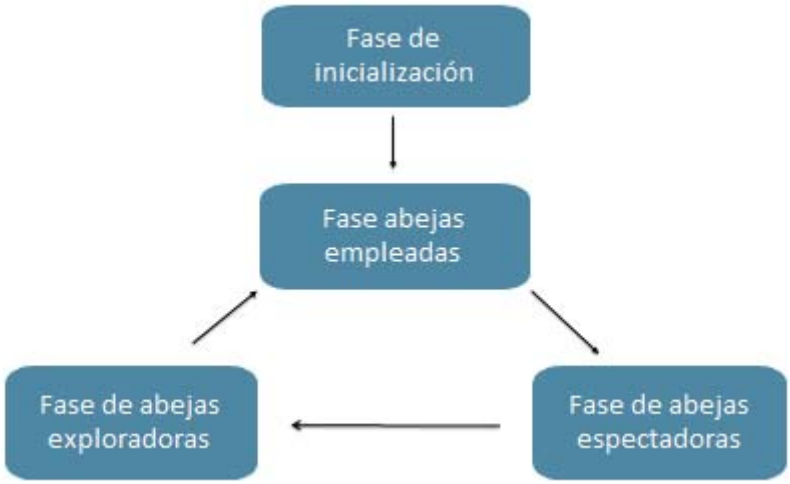


Figura 8.1 Flujo Algoritmo ABC.

8.3 Pseudocódigo ABC Original

1: Inicialización de las fuentes de alimentos. Éstas son generadas de manera aleatoria.

2: Se evalúa a cada fuente de alimento generada, es decir se obtiene su valor de rentabilidad, mediante su valor en la función objeto del problema (ecuación en cap. 2).

3: **Generación=1;**

4: **REPETIR**. Inicia el ciclo de búsqueda.

5: Se producen nuevas soluciones por las **abejas empleadas** para cada posición o fuente usando la siguiente evaluación:

6: $u_{ij}=x_{ij}+\phi_{ij}(x_{ij}-x_{kj})$

7: En donde:

8: x_{ij} es la posición actual (fuente de alimento actual).

9: x_{kj} es una posición tomada aleatoriamente conforme al número de fuentes existentes.

10: ϕ_{ij} valor aleatorio entre -1 y 1.

11: Se evalúa la nueva posición u_{ij} y si el valor que se obtiene es mejor al valor de la fuente actual x_{ij}, entonces se sustituye la posición actual por la nueva posición: $x_{ij}=u_{ij}$

12: Se producen nuevas soluciones por las **abejas en espera** bajo los siguientes criterios:

13: **Torneo**: consiste en seleccionar aleatoriamente *t* posiciones para cada posición o fuente conforme al número de fuentes existentes y evaluar cuál de todas es la mejor. Una vez determinada, se repite el proceso de las abejas empleadas utilizando los datos de la posición seleccionada y la posición actual (pasos 5 y 6).

14: Se determina el abandono de las fuentes de alimento por las abejas scouts si el límite permitido para abandonarla es rebasado, por lo tanto se reemplaza la posición actual por una nueva posición que es generada aleatoriamente.

15: Una vez procesados los pasos anteriores se evalúa y memoriza cuál de todas las fuentes de alimento actuales es la mejor.

16: Generación=Generación + 1;

17: **HASTA (Generación=NUM_MAX_GEN).**

18: **FIN.**

Figura 8.2 Pseudocódigo Algoritmo ABC Colony

9 Propuesta de Métrica basada en Algoritmo Artificial BeeColony para la detección de comunidades solapadas

Se implementó una adaptación del algoritmo BeeColony (ABC) para la detección del mejor conjunto de comunidades [7] usando como función a maximizar una adaptación modularidad. Tomando como base el algoritmo original, se presentan las siguientes modificaciones:

- La restricción del espacio de búsqueda
- La representación de la Solución
- Adaptación de la Modularidad como función objetivo

9.1.1 Restricción del espacio de búsqueda.

El espacio de búsqueda determinado para la detección de comunidades en grafos completos, se determinó como los nodos y enlaces existentes en la red analizada, como modo de facilitar el desarrollo del algoritmo es que se utiliza la matriz de adyacencia de la matriz.

Una segunda restricción se incluyó al momento de generar nuevas soluciones de manera aleatoria, en esta instancia se restringió que la nueva solución obtenida de manera aleatoria sea consistente a nuestro espacio de búsqueda, es decir que los nuevos enlaces sean consistentes a los de la red tratada

9.1.2 Representación de la solución

Usualmente las redes complejas son representadas como grafos, los cuales están compuestos por nodos y enlaces, por lo que las comunidades corresponden a un grupo de nodos. En este caso las distintas comunidades serán representadas como una representación genética, es decir, como un genotipo. En esta representación, las soluciones serán definidas como un vector el cual se le asigna un valor, el cual representa un nodo adyacente a este (ver imagen xxx).

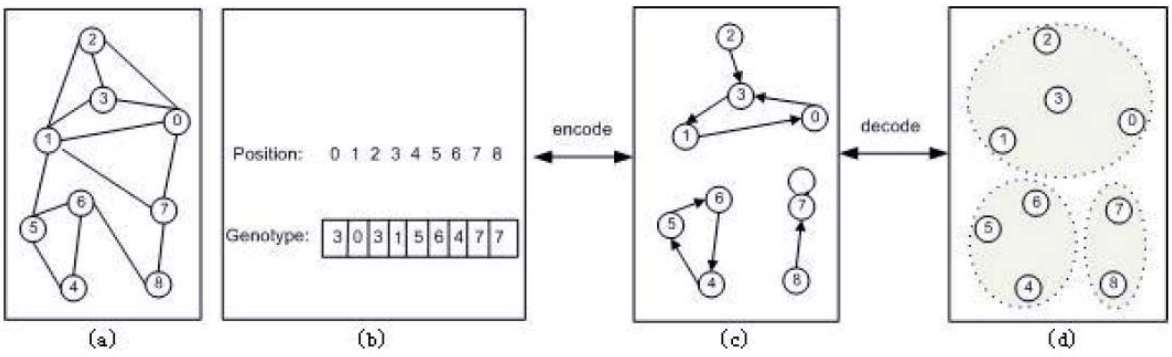


Figura 9.1 Representación de la Solución

En la imagen se pueden observar 4 etapas, en la etapa (a) se grafica un grafo ejemplo que cuenta con 9 nodos conectados entre sí. En la etapa (b) se genera una solución (genotipo) en donde cada posición del vector corresponde a un nodo y cada valor asignado al vector corresponde a un nodo adyacente a la posición en que se ubica. En la parte (c) se puede observar el mismo grafo ejemplo pero graficado como se determinó en la solución expuesta en (b). Finalmente en (d) y por medio de un algoritmo es que se puede determinar qué nodo corresponde a qué comunidad, por lo que quedan diferenciadas claramente.

Dicha representación, por definición, funciona exclusivamente para redes disjuntas, por lo que para esta nueva etapa se agregó una lógica extra, la cual se definen a continuación:

- Se transforma desde una representación vectorial a una representación matricial NxN en donde la dimensión 1xN representa a que comunidad pertenece dicha posición, mientras que las siguientes dimensiones 2xN hasta NxN corresponden a otras comunidades que pertenezca dicha posición (nodo), lo que significará que existe solapamiento. En caso de no existir solapamiento dichas dimensiones se completarán con un valor por defecto (-1). A continuación, se presenta un ejemplo:

	0	**1**	**2**	**3**	**4**
0	0	1	1	0	2
1	-1	0	-1	-1	1
2	-1	-1	-1	-1	0
3	-1	-1	-1	-1	-1
4	-1	-1	-1	-1	-1

Figura 9.2 Representación matricial de la solución

9.1.3 Adaptación de la Modularidad como función objetivo

La modularidad de Newman se utiliza para medir la calidad de la estructura de comunidades disjuntas de una red. Sin embargo, es más realista que los nodos en las redes pertenecen a más de una comunidad, dando lugar a la superposición de las comunidades. Por ejemplo, es muy común que las personas en las redes sociales se caracterizan naturalmente por una representación múltiple de la comunidad en función de sus familias, amigos, profesiones, etc. Por esta razón, el descubrimiento de las comunidades superpuestas es más realista, lo que ha derivado a que en este último tiempo se propusieron varias extensiones de modularidad en comunidades solapadas para medir la calidad.

La idea básica detrás de la modularidad de Newman para la cuantificación de las comunidades es la densidad en un subgrafo de una red en comparación con un modelo-nulo. Aquí, el modelo-nulo se define como un subgrafo con mismo número de vértices, mismo número de bordes, y el mismo grado de distribución como el subgrafo original, pero

los enlaces se colocan al azar. En un gráfico tal azar, la probabilidad de tener vértice ***i*** conectado al vértice ***j*** viene dada por $P_{ij} = \frac{k_i k_j}{4m^2}$, donde ***m*** es el número total de enlaces en la red, y k_i *y* k_j son los grados de vértices *i* y *j*, respectivamente.

En el caso de las comunidades superpuestas, un vértice puede pertenecer a más de una comunidad. La fuerza de su adhesión a cada comunidad puede ser diferente dependiendo del número de conexiones que tienen con cada comunidad.

Teniendo en cuenta dicha observación, la definición de la modularidad para evaluar la superposición de las comunidades se extiende como:

$$Q_o = \frac{1}{2m}\sum_n \sum_{i\in c_n, j\in c_n}\left[\frac{k_i^c k_j^c}{k_i k_j}\right]\left[a_{ij} - \frac{k_i k_j}{2m}\right],$$

$$= \frac{1}{2m}\sum_n \sum_{i\in c_n, j\in c_n} k_i^c k_j^c \left[\frac{a_{ij}}{k_i k_j} - \frac{1}{2m}\right].$$

Aquí, $k_i^c = \sum_{p\varepsilon V_{cn}} a_{ip}$ y $k_j^c = \sum_{q\varepsilon V_{cn}} a_{jp}$, donde v_{cn} es el conjunto de vértices de la comunidad cn.

De manera similar a la modularidad tradicional *Q*, en la modularidad extendida propuesta Q_0 = 0 cuando todos los nodos pertenecen a la misma comunidad y se obtendrá un valor más alto para indicar una estructura de comunidad más fuerte.

9.2 Algoritmo.

1. Inicialización de las fuentes de alimentos. Éstas son generadas de manera aleatoria, contemplando restricciones del espacio de búsqueda (Sección 9.1.1).
2. Se decodifica cada vector solución generando un vector solución decodificado (sección 9.1.2).
3. Generar matriz solución por cada fuente de alimento.
    a. Se evalúa cada punto crítico para determinar si existe o no solapamiento.
    b. Un punto crítico corresponde a un nodo que no pertenece a una comunidad, pero es adyacente a esta.
    c. Se utiliza la función tradicional de modularidad (Sección 5.1.1) para evaluar el fitness de la comunidad con y sin el punto crítico. De mejorar el valor del fitness incluyendo este punto, se infiere existencia de solapamiento agregando este nodo a la comunidad evaluada.
4. Calculamos el fitness de cada matriz solución generada, a través de la función Adaptación de la Modularidad (sección 9.1.3)

5. Se selecciona como mejor solución la matriz con mejor fitness, según la evaluación del paso 4.
6. Se compara la mejor solución obtenida con la solución global almacenada, de ser mayor la solución obtenida esta reemplaza la solución global.
7. Se producen nuevas soluciones en base a la mejor solución. Se repite el ciclo hasta encontrar la solución óptima o hasta alcanzar el máximo de ciclos.

10 Resultados experimentales

10.1 Dataset

Para el análisis del código propuesto más adelante se presentan los conjuntos de datos que se han usado, los grafos se han ejecutado directamente en el algoritmo de detección de comunidades desarrollado.

10.1.1 Zachary Karate Club

Zachary es una red que muestra las relaciones de los miembros de un club de Karate de una universidad americana en los años 70. Contiene 34 miembros. Después de la construcción de esta red, el club en cuestión se dividió en dos tras disputas internas [10].

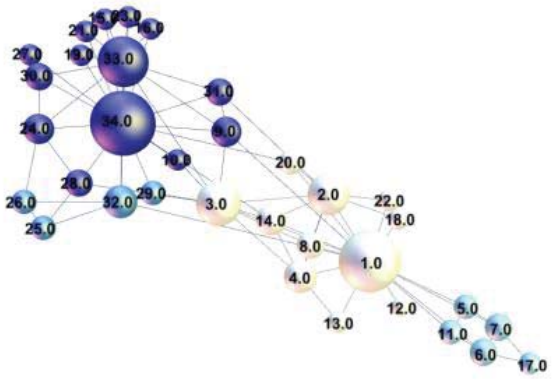


Figura 10.1 Representación Zachary Karate Club

10.1.2 American College football

La red ’football’ recoge los partidos de los equipos universitarios de la división IA de fútbol americano universitario, durante la temporada regular del año 2000. Los nodos representan los equipos y los enlaces, partidos entre ellos. Esta red es característica por su conocida estructura de comunidades, en el cual, los 115 equipos están divididos en conferencias de aproximadamente 8-12 equipos. Debe saberse que los partidos entre equipos de la misma conferencia son más frecuentes que entre equipos de diferente conferencia [12].

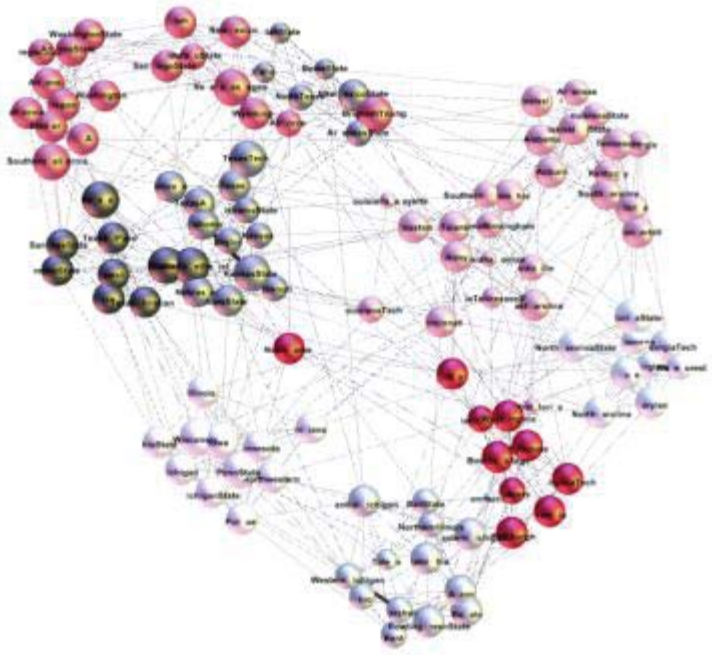


Figura 10.2 Representación American College football

10.1.3 Jazz

Los datos fueron obtenidos de la base de datos digital Red Hot Jazz [17]. En este análisis se incluyeron 198 bandas que llevan a cabo entre 1912 y 1940, con la mayoría de las bandas en la década de 1920. La base de datos se enumeran los músicos que tocaban en cada banda sin distinguir qué músicos tocaban en diferentes momentos, por lo que no es posible estudiar la evolución temporal de la red de colaboración. Las bandas contienen 1275 nombres diferentes de músicos. Sin embargo, es importante subrayar que este número no representa necesariamente el número de individuos, similar a la observación señala en [18]. El mismo músico podría aparecer con diferentes nombres en la base de datos. Por ejemplo, el músico Henry Allen aparece como Henry Allen, Red Allen o Henry Red Allen. También en algunas bandas no se conocen los nombres de algunos músicos. Como consecuencia de ello se citan en la base de datos con el mismo nombre: desconocido. En la Fig. 10.4 se muestra la distribución de los músicos que han jugado en una banda. Presenta una forma sesgada, con un pico alrededor de 5 - 10 músicos y un puñado de grandes bandas que incluyen hasta 171 músicos.

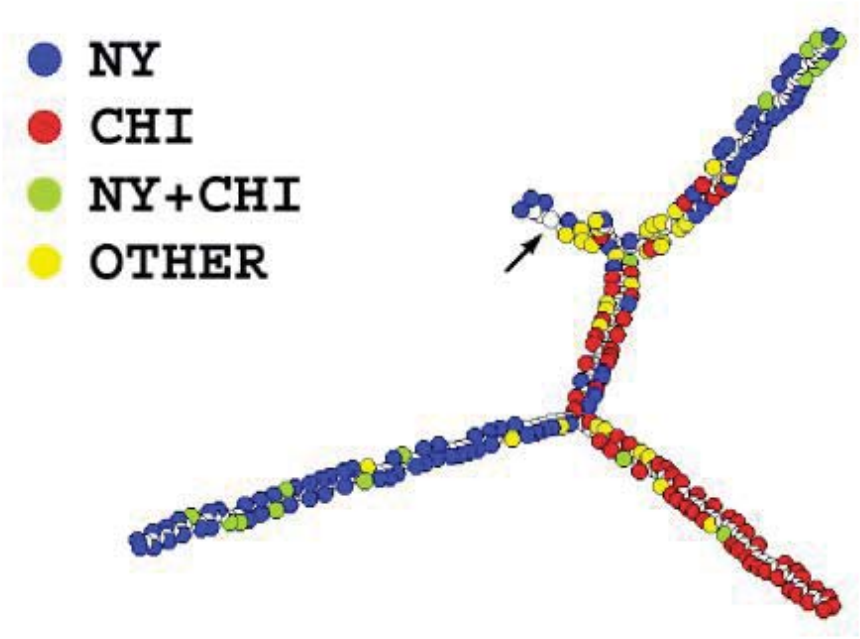


Figura 10.3 Representación Red hot Jazz

10.2 Resultados

10.2.1 Tablas

Modularidad

	Paper [14,15,16]	Experimental
Zachary Karate Club	0.72	0.70
America College Football	0.61	0.57
Red Hot Jazz	0.70	0.48

Tabla 1 Resultados Modularidad

Cantidad de Comunidades

	Paper [14,15,16]	Experimental
Zachary Karate Club	3	3
American College Football	10	12
Red Hot Jazz	4	6

Tabla 2 Resultado Comunidades

NMI

	Experimental
Zachary Karate Club	0.92
America College Football	0.85
Red Hot Jazz	0.57

Tabla 3 Resultado NMI

10.2.2 Representación grafica

A continuación, se comparan los resultados del dataset correspondiente a la Red Zachary Karate Club, en donde se puede observar gráficamente el resultado obtenido.

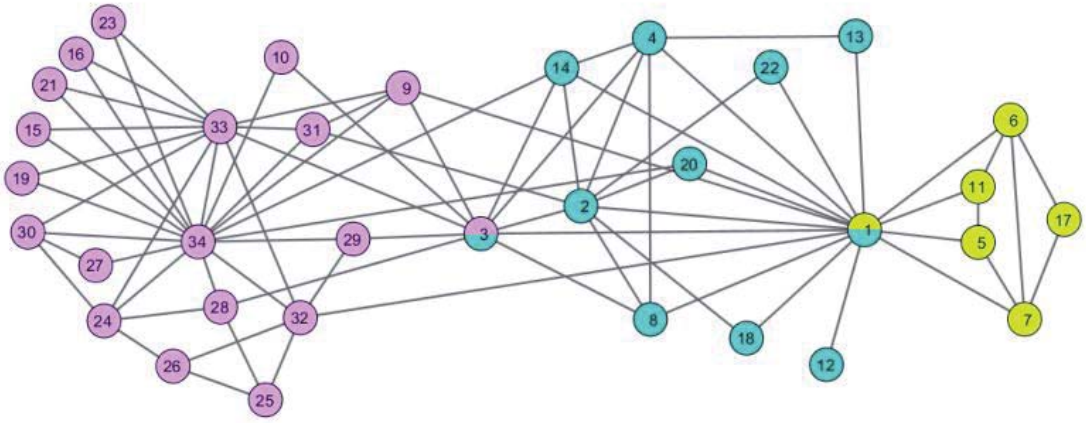


Figura 10.4 Resultados Zachary Karate Club Paper [14]

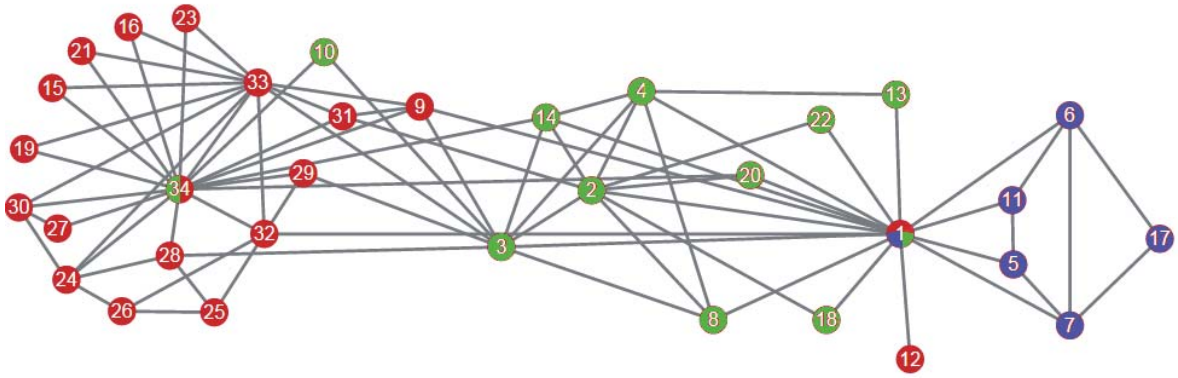


Figura 10.5 Resultados Zachary Karate Club propuesto

11 Conclusiones

El estar inmerso en este proyecto nos ha servido para estar al tanto de las nuevas metodologías en el área del análisis de redes complejas, en donde el estudio de las ciencias y las matemáticas han influido en la creación de una base de conocimientos para comprender el cómo funcionan estas estructuras y de cómo sus propiedades son importantes para el desarrollo de esta área investigativa.

La organización en comunidades es una de las características más significativas de las redes complejas. Por tal motivo, las técnicas propuestas para la detección de tales grupos, juegan un papel fundamental en el análisis estructural de este tipo de redes. En este trabajo se ha llevado a cabo una detallada investigación del estado del arte en el contexto de la detección de comunidades. Como resultado de este estudio, se identificaron las principales limitaciones existentes en los algoritmos que han tenido una mayor relevancia en este dominio de investigación. Con base en esto, se ha podido concluir que el diseño de nuevas y mejores técnicas para resolver este problema, sigue siendo de gran importancia para un mejor entendimiento de las redes complejas.

La base teórica adquirida ha sido útil en el sentido de comprender la problemática en cada una de sus partes: comprender la definición de redes complejas, su significado, definición comunidades y cuando se hace referencia a comunidades solapadas; la teoría de grafos, sus definiciones formales y fórmulas matemáticas y geometría; métodos de detección tanto tradicionales como novedosos, en donde se incluyen una clara diferenciación de los tipos de metodologías, el detalle de los principales indicadores utilizados y finalmente los principales algoritmos con sus alcances, ventajas y desventajas.

En primera instancia se realizó la búsqueda de comunidades disjuntas la cual se utilizó como base para el desarrollo de la búsqueda de comunidades solapadas. Tras los resultados obtenidos se infiere que se pueden utilizar para la detección de comunidades solapadas, previa adaptación de los métodos anteriores.

Finalmente los resultados obtenidos, nos llevan a inferir que el método utilizado resulta útil para la búsqueda de comunidades solapadas, esto en comparación a los métodos estudiados.

12 Referencias

[1] Maarten van Steen. Graph Theory and Complex Networks, 2010.

[2] Santo Fortunato. Community detection in graphs. Physics Reports, page 58, 2010.

[3] M E J Newman and M Girvan. Finding and evaluating community structure innetworks. Physical Review E - Statistical. Nonlinear and Soft Matter Physics, 2004.

[4] A.Lancichinetti Janos Kertesz, S.Fortunato. Detecting the overlapping and hierarchival community structure in complex networks. New journals of Physics, 2009.

[5] Nuwan Ganganath, Guanrong Chen , and Chi-Tsun Cheng, Detecting Hierarchical and Overlapping Community Structures in Networks.

[6] http://mf.erciyes.edu.tr/abc/

[7] Ahmed Ibrahem Hafez, Hossam M. Zawbaa, Aboul Ella Hassanien, Aly A. Networks community detection using artificial bee colony swarm optimization, 2014.

[8] M. E. Newman, "Fast algorithm for detecting community structure in networks," Physical Review E, vol. 69, no. 6, p. 066133, 2004.

[9] C. Bron and J. Kerbosch, "Algorithm 457: Finding all cliques of an undirected graph," Commun. ACM, vol. 16, no. 9, Sep 1973.

[10] Li, Z, Zhang, S, Wang, R, Zhang, X, Chen, L (2008) Quantitative function for community detection. Physical Review E 77:36109.

[11] [9] Lusseau, D et al. (2003) The bottlenose dolphin community of Doubtful Sound features a large proportion of long-lasting associations. Behavioral Ecology and Sociobiology 54:396–405.

[12] Girvan, M, Newman, M (2002) Community structure in social and biological networks. Proceedings of the National Academy of Sciences 99:7821.

[13] Xiang-Sun Zhang a, Rui-Sheng Wang , Yong Wang, Ji-Guang Wang , Yu-Qing Qiu , Lin Wang and Luonan Chen Modularity optimization in community identification of complex networks REFERENCIA DE LOS RESULTADOS

[14] Nuwan Ganganath, Guanrong Chen, and Chi-Tsun Cheng, Detecting Hierarchical and Overlapping Community Structures in Networks

[15] Chien-Yu Chen, A New Approach for Overlapping Community Detection by Adding Node Weight in Modularity Optimization, 2013

[16] Mingming Chen, Konstantin Kuzmin, Boleslaw K. Szymanski, Extension of Modularity Density for Overlapping Community Structure, 2014

[17] The Red Hot Jazz Archive, available at http://www.redhotjazz.com 11

[18] M. E. J. Newman, Who is the best connected scientist? A study of scientific coauthorship networks. Phys.Rev. E 64 (2001) 016131; Phys.Rev. E 64 (2001) 016132

[19] Nam P. Nguyen, Thang N. Dinh, Sindhura Tokala, and My T. Thai. Overlapping communities in dynamic networks: Their detection and mobile applications. In Proceedings of the 17th Annual International Conference on Mobile Computing and Networking, MobiCom '11, pages 8596, New York, NY, USA, 2011. ACM.

[20] Vincent D. Blonde Renaud Lambiottel, Jean-Loup Guillaume and Etienne Lefebvre. Fast unfolding of communities in large networks. Journal of Statical Mechanicstheory and Experimental, page 10, 2008.