RNPS No. 2143
ISSN 2072-6260
Versión Digital

CENATAV
Centro de Aplicaciones de Tecnologías de Avanzada
MINISTERIO DE LA INDUSTRIA BÁSICA

REPORTE TÉCNICO
Minería de Datos

SERIE GRIS

Detección de comunidades con traslape en redes complejas: Un estado del arte

Gustavo Lara-Alvarez y Airel Pérez-Suárez

RT_028 **enero 2015**

7ma. No. 21812 e/218 y 222,
Rpto. Siboney, Playa;
La Habana.
Cuba. C.P. 12200
www.cenatav.co.cu

RNPS No. 2143
ISSN 2072-6260
Versión Digital

CENATAV
Centro de Aplicaciones de Tecnologías de Avanzada
MINISTERIO DE LA INDUSTRIA BÁSICA

REPORTE TÉCNICO
Minería de Datos

SERIE GRIS

Detección de comunidades con traslape en redes complejas: Un estado del arte

Gustavo Lara-Álvarez y Airel Pérez-Suárez

RT_028 **enero 2015**

7ma. No. 21812 e/218 y 222,
Rpto. Siboney, Playa;
La Habana.
Cuba. C.P. 12200
www.cenatav.co.cu

Tabla de contenido

Lista de figuras

Detección de comunidades con traslape en redes complejas: Un estado del arte

Gustavo Lara-Alvarez y Airel Pérez-Suárez

Equipo de Investigaciones en Minería de Datos, Centro de Aplicaciones de Tecnologías de Avanzada (CENATAV),
La Habana, Cuba
{glara,asuarez}@cenatav.co.cu

RT_028, Serie Gris, CENATAV
Aceptado: 14 de enero de 2015

Resumen. Tanto en la naturaleza como en la sociedad se pueden identificar numerosos sistemas complejos y fenómenos de interacción, los cuales pueden ser modelados efectivamente usando redes complejas. En vista de esto, el análisis estructural de este tipo de redes resulta de gran importancia práctica. La estructuración en comunidades es una propiedad inherente de las redes complejas reales. Los vértices que pertenecen a la misma comunidad suelen compartir características similares o desempeñar roles semejantes en el funcionamiento de la red. En este reporte, se describe el problema de la detección de comunidades con traslape en redes complejas y se lleva a cabo un análisis estructurado de las principales técnicas propuestas hasta la fecha para el descubrimiento de estos grupos. Además, se lleva a cabo una comparación cualitativa de dichas técnicas, en función de un conjunto de características deseables en este tipo de algoritmos. Finalmente, se proponen varias líneas de investigación que se consideran de gran interés práctico y deben ser abordadas como trabajo futuro.

Palabras clave: comunidad, red compleja, agrupamiento, detección de comunidades, estructuración jerárquica de comunidades, comunidades dinámicas, comunidades con traslape, comunidades difusas.

Abstract. Many complex systems and interaction phenoma, observed in both nature and society, can be accurately modeled using complex networks. The structural analysis of such complex networks is very important for many practical applications. An inherent property of such networks is their arrangement into communities; in this arrangement, nodes belonging to the same community usually share similar features or they have the same role in the network. In this report, the problem of overlapping community detection is addressed and a critical analysis of the most relevant algorithms proposed for this purpose is presented. In addition, a qualitative comparison of these algorithms, regarding a list of desirable characteristics of this kind of algorithms, is presented. Finally, some future lines of research, considered of great importance, are proposed.

Keywords: community, complex networks, clustering, community detection, hierarchical community structure, dynamic communities, overlapping communities, fuzzy communities.

1. Introducción

Una *red compleja* es un modelo matemático basado en grafos que permite modelar fenómenos de interacción, presentes en múltiples escenarios del mundo real. Este tipo de fenómenos se ha convertido en un asunto trascendental y cotidiano en nuestra sociedad, lo que ha desatado un creciente interés en el análisis de sus propiedades y las leyes que rigen su evolución [3].

Nuestra propia existencia se basa en el funcionamiento de varias redes bioquímicas. Redes tecnológicas y de información inciden cada vez más en las distintas esferas de la ciencia. La toma de decisiones en muchas aplicaciones comerciales necesita tener en cuenta distintos tipos de redes, tales como sitios de comercio electrónico y tiendas virtuales. Como se observa, descubrir conocimiento útil en redes complejas ha sido el centro de atención de académicos de diferentes especialidades.

En la mayoría de las redes reales, el orden coexiste con el desorden [3]. La noción de desorden en un grafo fue introducida por Erdös y Rényi [4] con el grafo *aleatorio*[1]. En este tipo de grafo, la probabilidad de que un par de vértices estén conectados es igual para todos los posibles pares de vértices. La distribución de las aristas entre los nodos de un grafo *aleatorio* es altamente homogénea, por lo que la mayoría de los vértices tienen grados similares. En cambio, las redes reales muestran un alto grado de orden y organización. Debido a esto, se pueden definir los distintos roles que desempeñan los nodos en la red, de acuerdo a su interacción con la misma. No obstante, el análisis de la distribución de las aristas en estas redes no se puede realizar únicamente desde una perspectiva global, debido a otra propiedad que cumplen la mayoría de las redes que modelan escenarios reales. Esta propiedad es la estructura basada en *comunidades*.

Generalmente, la forma en que los nodos de una red se interrelacionan provoca la formación de grupos de vértices o módulos estructurales que son conocidos como *comunidades*. Los nodos de estos módulos suelen compartir propiedades comunes o desempeñar roles similares en la red. Los vértices pertenecientes a la misma comunidad tienen una alta concentración de aristas entre ellos y poca interacción con el resto de la red. En la figura 1 se puede observar una red pequeña con comunidades bien definidas.

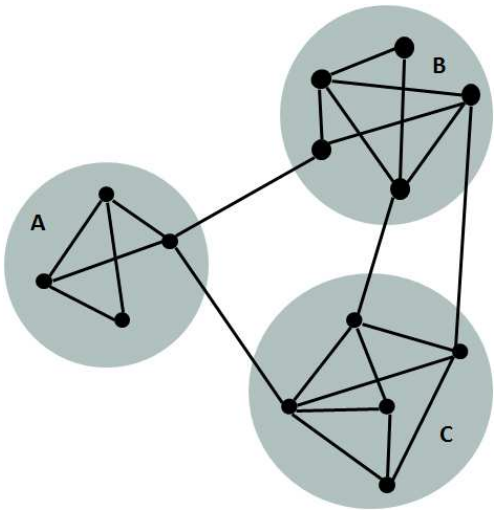

Fig. 1. Ejemplo de una red pequeña con tres comunidades delimitadas por la zona sombreada.

La existencia de comunidades en redes sociales es bastante natural, la propia sociedad nos ofrece una gran variedad de organizaciones posibles tales como familia, ubicación geográfica, especialidad profesional, ideología política, religión, comunidades virtuales como los grupos de Facebook, entre muchas otras. En redes bioquímicas o neurológicas las comunidades pueden verse como grupos funcionales, de manera que al delimitar tales grupos en la red se puede simplificar considerablemente el análisis funcional [5,6]. La detección de comunidades virtuales en la web, permite el refinamiento de sistemas de recomendación para los sitios de venta, y la realización de búsquedas más completas en los sistemas de recuperación de

[1] del inglés *random graph*

información [7,8]. En el contexto de las telecomunicaciones conocer los grupos de usuarios con servicios similares y ubicados en la misma región geográfica, facilita un servicio personalizado y de mayor calidad [9].

Las primeros trabajos en la detección de comunidades se remontan a 1927 con la búsqueda de grupos de tendencias políticas similares basándose en los patrones de votación de pequeñas organizaciones políticas [10]. En los últimos años, este ha sido un tema ampliamente tratado y muchos esfuerzos han sido encaminados en ese sentido [3,11,12,13,14], incorporando novedosos enfoques provenientes de distintas especialidades como sociología, física y ciencias de la computación. La mayoría de estos trabajos se enfocan en la detección de comunidades disjuntas. En este tipo de detección, los vértices de la red se agrupan en comunidades, de manera que cada vértice pertenece a una única comunidad.

Sin embargo, en muchos escenarios prácticos puede existir traslape (vértices en común) entre las comunidades de la red. Los vértices que pertenecen a múltiples comunidades, juegan un papel fundamental en el funcionamiento del sistema, que se modela con la red compleja. Estos vértices facilitan la comunicación entre los distintos grupos e inciden notablemente en la evolución de la red. En vista de esto, se ha incrementado el número de trabajos propuestos para la detección de comunidades con traslape. Esta tarea consiste en identificar las comunidades de la red, permitiendo que exista traslape entre estas.

A pesar de los resultados obtenidos en el perfeccionamiento de este tipo de técnicas, las metodologías existentes presentan limitaciones que impiden su aplicación en numerosas situaciones prácticas o no son capaces de adaptar su funcionamiento a nuevos requerimientos. Por tal motivo, el desarrollo de nuevos y mejores algoritmos para la detección de comunidades con traslape en redes complejas sigue siendo una línea de investigación abierta.

El resto de este reporte técnico está organizado como sigue: en la sección 2 se introducen algunos conceptos preliminares relacionados con las redes complejas y se explican brevemente las características principales de la detección de comunidades en este tipo de redes. Posteriormente, en la sección 3 se realiza un análisis estructurado de los principales algoritmos reportados en la literatura capaces de detectar comunidades con traslape en redes complejas. Luego, en la sección 4 se lleva a cabo una comparación cualitativa de los algoritmos abordados en el documento, de acuerdo a varios aspectos que deben caracterizar a un algoritmo de detección de comunidades. Por último, en la sección 5 se presentan las conclusiones del trabajo y se identifican algunas líneas de investigación que deben ser desarrolladas como trabajo futuro, para lograr un mejor entendimiento en el contexto de detección de comunidades.

2. Conceptos preliminares y características del problema

En esta sección, se presentan los conceptos preliminares necesarios para entender el soporte teórico de este trabajo.

El problema de detectar comunidades puede ser visto como un tipo especial de agrupamiento, ya que en el mismo se realiza una clasificación no supervisada de los vértices de la red. La principal diferencia entre la detección de comunidades y el enfoque tradicional de agrupamiento es que para definir las comunidades no solo se tiene en cuenta el grado de similitud entre los elementos, sino la manera en que estos interactúan entre sí. Por tal motivo, una comunidad no se representa como un grupo de vértices, sino como el subgrafo de la red, formado por dicho conjunto de vértices y las aristas existentes entre estos.

El objetivo de la detección de comunidades es organizar los vértices de la red en grupos, de manera que los vértices pertenecientes al mismo grupo estén lo suficientemente relacionados entre sí, como para inferir que son de la misma comunidad, mientras que vértices que estén ubicados en distintos grupos estén lo suficientemente desvinculados como para poder determinar que son de comunidades diferentes.

Los algoritmos de detección de comunidades pueden clasificarse de acuerdo a las relaciones de pertenencia entre los vértices de la red y las comunidades detectadas. Si los vértices se agrupan de manera que cada uno pertenece a una comunidad, se está en presencia de una *partición* de la red. Los algoritmos que solo son capaces de obtener particiones de la red, se conocen como algoritmos de *detección de comunidades disjuntas* (DCD). Por otro lado, si los vértices pueden pertenecer a varias comunidades, se está en presencia de un *cubrimiento* de la red. Los algoritmos que permiten obtener un cubrimiento de la red, se pueden clasificar de acuerdo al modo en que realizan las asignaciones de los vértices a las comunidades. Si la relación de pertenencia entre los vértices y las comunidades es binaria (pertenece o no pertenece), entonces se dice que el algoritmo es de *detección de comunidades con traslape* (OCD). En cambio, si los vértices se asignan a las comunidades teniendo en cuenta que pueden tener distintos grados de pertenencia, entonces se dice que el algoritmo es de *detección de comunidades difusas* (FCD). En lo adelante, para referirse de manera general al agrupamiento obtenido por un algoritmo de detección de comunidades, se utilizará el término partición.

En los últimos años, un gran número de trabajos se ha presentado para la detección de comunidades en redes complejas. No obstante, estos esfuerzos han sido dirigidos de una manera desorganizada [3]. Esto se debe principalmente a que no existe una plataforma teórica que defina de manera consensuada lo que es una comunidad o lo que debe hacer un algoritmo de detección de comunidades. En las subsecciones 2.1 y 2.2 se abordan las definiciones de comunidades más trabajadas en la literatura y los principales mecanismos de evaluación de los resultados obtenidos, respectivamente. En las subsecciones 2.3 2.4 se describen las características principales de algunas líneas de interés relacionadas con la detección de comunidades.

Existen distintas clasificaciones de redes complejas de acuerdo a las características de los datos y las relaciones que modelan. En una red puede ser representado el sentido de las relaciones entre las entidades (dirigidas o no dirigidas), los vértices y/o aristas de la red pueden ser ponderados de acuerdo a su relevancia (ponderadas o no ponderadas), la evolución de la red en el tiempo puede ser modelada (dinámicas o estáticas), entre otras.

En este reporte técnico se abarcarán la mayoría de estos tipos de redes complejas. En vista de esto, se propone representar una red compleja a partir de un *grafo etiquetado*.

Definición 1 (Grafo etiquetado). *Un grafo etiquetado es una tétrada* $\langle V, E, L, l \rangle$ *donde:*

- V *es un conjunto cuyos elementos son llamados vértices o nodos y representan las entidades de la red,*
- $E \subset \{\{u,v\} \,|\, u,v \in V, u \neq v\}$ *es un conjunto de pares de vértices, cuyos elementos son llamados aristas o enlaces. Se suele decir que la arista* $\{u,v\}$ conecta *a los vértices u y v y que por lo tanto, u y v son adyacentes o están conectados. El conjunto de vértices adyacentes o vecinos de un nodo v se denota como* $\Gamma(v)$*. El grado de un vértice v se denota como* d_v *e indica la cardinalidad del conjunto de vecinos de v.*
- L *es un conjunto de* etiquetas
- $l : V \cup E \rightarrow L$ *es una función que asigna etiquetas a los vértices y aristas del grafo.*

Cada etiqueta puede ser un conjunto de valores, permitiendo representar el peso para redes ponderadas, el tiempo asociado a la creación de los elementos para redes dinámicas y/o cualquier tipo de valores asociados a la descripción del elemento asociado. En lo adelante, cuando se mencione grafo se hace alusión a este tipo de grafos, a menos que se indique lo contrario.

Si las aristas del grafo son pares ordenados de vértices ($\{u,v\} \neq \{v,u\}$), entonces se dice que la red es *dirigida*, en caso contrario se dice que la red es *no dirigida*. Si cada una de las aristas tienen asociadas un peso se dice que la red es *ponderada*, en caso contrario se dice que es *no ponderada*. Si algunos elementos del grafo no se etiquetan, se dice que es un grafo *parcialmente etiquetado*, y si ningún elemento del grafo se etiqueta, se dice que el grafo es *no etiquetado*.

Definición 2 (Subgrafo y Supergrafo). *Sean $G_1 = \langle V_1, E_1, L, l \rangle$ y $G_2 = \langle V_2, E_2, L, l \rangle$ dos grafos que comparten el mismo conjunto de etiquetas L y la misma función etiquetadora l, se dice que G_1 es subgrafo de G_2 si $V_1 \subseteq V_2$ y $E_1 \subseteq E_2$. En este caso se utiliza la notación $G_1 \subseteq G_2$ y se dice que G_2 es supergrafo de G_1.*

Definición 3 (Subgrafo Inducido). *Sea V_S un conjunto de vértices y $G = \langle V, E, L, l \rangle$ un grafo. El subgrafo S inducido de G a partir de V_S, es el subgrafo de G formado por los vértices de V_S y todas las aristas de E que conectan los vértices de V_S entre sí, es decir: $S = \langle V_S, E_S, L, l \rangle$, donde $E_S = \{\{u,v\} | \{u,v\} \in E \wedge u,v \in V_S\}$.*

Un tipo de subgrafo inducido de gran importancia en la detección de comunidades es la *egonet* asociada a un nodo, la cual intuitivamente es la parte de la red que conoce el nodo a partir de sus vértices adyacentes.

Definición 4 (Egonet o Red Egocéntrica). *La egonet, ego-red o red egocéntrica de un vértice v en una red G, se define como el subgrafo inducido de G a partir de v y sus vértices adyacentes. Se dice que v es el ego o centro de la egonet.*

Sea $G = \langle V, E, L, l \rangle$ una red compleja, una manera de representar la estructura topológica de G es a partir de matrices de adyacencias.

Definición 5 (Matriz de adyacencia). *La matriz de adyacencia de G se denota como A y es la matriz cuadrada de orden $|V|$ donde cada elemento A_{ij} tiene valor 1 si existe una arista entre los vértices v_i y v_j, y en caso contrario el valor de A_{ij} es 0.*

Para el caso de grafos no dirigidos, esta matriz es simétrica. La suma de los elementos de la fila o columna i de la matriz indica el grado del vértice v_i. Los elementos de la diagonal de la matriz tienen valor 0, dado que ningún vértice puede estar conectados consigo mismo.

Definición 6 (Camino o ciclo). *Se dice que $P = (v_1, v_2, \ldots, v_n)$ es un camino en el grafo $G = \langle V, E, L, l \rangle$ si todo vértice de P está en V y para cada par de vértices v_i y v_{i+1} que sean consecutivos en P se cumple que $\{v_i, v_{i+1}\} \in E$. En tal caso se dice que v_1 y v_n están conectados por P. Si $v_1 = v_n$ se dice que P es un ciclo.*

Varias técnicas de detección de comunidades usan el concepto de *camino aleatorio*, el cual es una formalización matemática de la trayectoria que resulta de hacer sucesivos pasos aleatorios.

Definición 7 (Camino aleatorio). *Un camino aleatorio entre un par de vértices consiste en la creación de un camino que los una, donde en cada paso del camino, la probabilidad de moverse de un nodo a uno de sus vecinos es la misma, para todos los vértices con los que está conectado.*

A partir del concepto de camino, se pueden definir las nociones de *distancia* y *conectividad*.

Definición 8 (Distancia entre dos vértices). *Sean u y v vértices de un grafo G, la distancia entre u y v se define como la cantidad de aristas de un camino de longitud mínima que una a u y v. Si u y v no están conectados, entonces su distancia es ∞.*

Definición 9 (Diámetro de un grafo). *El diámetro de un grafo se define como la mayor distancia existente entre los vértices del grafo.*

Definición 10 (Grafo conexo). *Se dice que un grafo es conexo, si para cualquier par de vértices del mismo se puede encontrar un camino que los conecta. Si no existe un camino que una algún par de vértices del grafo, este se puede dividir en al menos dos subgrafos conexos. Los subgrafos conexos maximales de un grafo se denominan componentes conexas. En caso de que el grafo sea conexo, él mismo es su única componente conexa.*

Estrechamente vinculada a la noción de comunidad, aparece el concepto de *densidad*.

Definición 11 (Densidad). *La densidad de un grafo o red indica qué tan relacionados están los nodos de la red y se define como la razón entre la cantidad de aristas de la red y la máxima cantidad posible de aristas que puede existir en la red. Se denota como* $\delta(G)$ *y se calcula de la siguiente manera:*
$\delta(G) = \frac{|E|}{\binom{V}{2}}$.

Cuando $|E| = O(|V|)$ se cumple que $\delta(G) << 1$ y se considera que G es *poco denso*. Es importante señalar que complejidad de detectar comunidades aumenta a medida que se incrementa la densidad de la red. Esto se debe a que las redes densas ($|E| = O(|V|^2)$) tienen una distribución de aristas muy homogénea y el tamaño de las comunidades aumenta, haciendo que estas interaccionen cada vez más con el resto de la red.

2.1. Comunidades

Una de las principales dificultades de la detección de comunidades en redes complejas, es que no se ha logrado un consenso acerca de qué es una comunidad. Esto se debe a que muchos de los algoritmos propuestos, solucionan problemas específicos y definen las comunidades de acuerdo a las características propias del problema que abordan.

En la mayoría de los trabajos donde se intenta definir el concepto de comunidad, solo se proponen condiciones necesarias o suficientes que debe cumplir un grafo para considerarse una comunidad. En algunos trabajos ni siquiera se utilizan definiciones explícitas, sino que las comunidades se definen como los grupos obtenidos al aplicar el algoritmo. Un ejemplo de este tipo de trabajos es el algoritmo de Girvan y Newman [15].

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 qué 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.

2.1.1. Definiciones locales

Las definiciones de *comunidad* que siguen una perspectiva local, utilizan únicamente la información estructural de la propia comunidad. Estas definiciones se centran en el cumplimiento de dos propiedades inherentes de las comunidades, la *densidad* y el *aislamiento*.

Para el análisis de la densidad de una comunidad, solo es necesario considerar las aristas internas de la red. El mayor valor de densidad posible en un grafo, se alcanza cuando para todo par de vértices existe una arista que los conecta. En ese caso, se dice que el grafo es un *clique*. Las comunidades pueden ser vistas

como cliques maximales. Lamentablemente, los cliques de gran tamaño son poco frecuentes en las redes reales. Por tal motivo, requerir el valor máximo de densidad en un subgrafo puede incidir negativamente en el número de comunidades detectadas. Además, la relación entre los vértices de los cliques es muy homogénea y no se puede extraer información acerca de los roles de cada vértice en la comunidad.

Se han propuestos múltiples criterios con la intención de flexibilizar la definición de comunidad basada en cliques. Uno de estos criterios, consiste en definir las comunidades como subgrafos maximales donde para cualquier par de vértices del subgrafo, en la red se puede encontrar un camino de longitud menor que n que los conecta; siendo n un umbral previamente definido. Este tipo de comunidades se define como *n-cliques*[16] y aunque relaja un poco la restricción de los cliques, introduce nuevas dificultades. Una de estas dificultades es que el diámetro de un n-clique puede ser mayor que n e incluso puede tener distintas componentes conexas. Esto se debe a que los vértices del n-clique pueden estar conectados por caminos donde participan vértices de la red que no pertenecen al n-clique. Para evitar estas dificultades, en [17] se proponen dos nuevos tipos de comunidades, *n-clan* y *n-club*. Un n-clan es un n-clique cuyo diámetro es menor que n. Un n-club es un subgrafo maximal de diámetro n.

Otra manera de definir comunidades basadas en cliques, es especificando el nivel de adyacencia entre los vértices de la comunidad. Para esto se proponen los k-plex [18] y los k-core [19]. Un k-plex es un subgrafo donde cada uno de sus vértices, es adyacente con todos los restantes, exceptuando a lo sumo con k, mientras que un k-core es un subgrafo donde cada uno de sus vértices es adyacente con al menos k de los vértices restantes.

Las definiciones locales de comunidad que solo tienen en cuenta la densidad, pueden identificar como comunidad a un subgrafo con alta concentración de aristas entre sus vértices, pero que también esté fuertemente conectado con el resto de la red. En vista de esto, se han propuesto otros criterios locales para definir comunidades, los cuales tienen en cuenta el *aislamiento* de la comunidad. Para esto, además de tener en cuenta las aristas internas de la comunidad, se considera también la información de las aristas que conectan a dicha comunidad con el resto de la red.

La primera idea de este tipo es proveniente de la sociología y propone el uso de *LS-Sets* [20], los cuales son subgrafos que requieren que cada uno de sus nodos tenga más aristas con los demás vértices del subgrafo que con el resto de la red. Esta idea es equivalente a la establecida en [21] con el concepto de *comunidad fuerte*. En [21] también se propone usar el concepto de *comunidad débil*. Se dice que un subgrafo es una comunidad débil, si los vértices del mismo cumplen que la suma de sus grados internos es mayor que la suma de sus grados externos.

Los criterios cuantitativos para definir comunidades desde una perspectiva local, son fácilmente adaptables al caso de detección de comunidades con traslape, dado que solamente utilizan la información interna de las comunidades y no su relación con las demás comunidades del agrupamiento realizado.

2.1.2. Definiciones globales

La principal desventaja de las definiciones locales de comunidad, es que no son capaces de reflejar el papel de las comunidades en la estructura global de la red. Esto limita su uso en aplicaciones donde se realiza un análisis funcional de los sistemas complejos modelados por este tipo de redes. En vista de esto, se han propuesto varias definiciones de comunidad, que consideran las comunidades desde una perspectiva global; es decir, como componentes de la estructura de la red.

La mayoría de las definiciones globales de comunidad se basan en la realización de agrupamientos en la red. Para esto, se definen medidas de calidad de agrupamiento, que indiquen el cumplimiento de alguna propiedad inherente a las redes con comunidades bien definidas. Luego, las comunidades se definen como los grupos de la partición de la red que maximiza la medida de calidad utilizada en la evaluación del agrupamiento, esta definición puede aplicarse también para el caso de las comunidades con traslape o difusas.

Entre las medidas de calidad propuestas para este fin, la más popular de todas es la *modularidad* [22]. La modularidad de una red, indica la diferencia existente entre esta y un grafo aleatorio con la misma distribución de grados de los vértices. Esta medida será analizada detalladamente en la sección 2.2.2.

2.2. Evaluación de algoritmos de detección de comunidades

Seleccionar un algoritmo de detección de comunidades que sea efectivo en una determinada aplicación, es un problema igual de complejo que el propio diseño del algoritmo en sí. Esto se debe a tres factores principales. En primer lugar está dado por el significativo número de técnicas que han sido propuestas para la detección de comunidades. El segundo factor es la presencia de una gran diversidad de redes complejas en el mundo real. Entre los distintos tipos de redes complejas pueden existir marcadas diferencias en cuanto a naturaleza de los datos y/o estructura topológica. Esto incrementa la complejidad de seleccionar una medida de calidad de partición que sea adecuada para todos los tipos de redes. El tercer factor es el hecho de que no existe ningún concepto unificado de comunidad, que cubra todos los dominios de aplicación.

Evaluar la efectividad de un algoritmo de detección de comunidades consta de dos partes fundamentales. En la primera se valida la calidad de los resultados obtenidos. En la segunda se evalúa la eficiencia del algoritmo en función de su complejidad temporal y su consumo de memoria. En esta sección, solo se aborda la primera parte y se realiza un breve análisis de las principales medidas de calidad de partición.

La comparación entre distintas particiones de la red es muy común en la detección de comunidades. Por ejemplo, existen muchos algoritmos que dependen de parámetros para su funcionamiento y usualmente requieren de una fase de ajuste, donde se ejecuta el algoritmo con distintas selecciones de valores de parámetros, obteniéndose particiones diferentes para cada selección. Otro ejemplo donde se necesitan comparar un grupo de particiones es en los algoritmos que siguen un enfoque jerárquico y generan un gran número de particiones diferentes para la formación del dendograma. En ambas situaciones, es necesario contar con funciones de calidad que permitan determinar cuáles son los mejores resultados obtenidos. En vista de esto, se han llevado a cabo numerosos trabajos donde se presentan distintos enfoques para evaluar la eficacia de los algoritmos de detección de comunidades. No obstante, a partir de estudios empíricos realizados en [23] sobre las distintas medidas de calidad existentes, se ha mostrado que ninguna de estas medidas es efectiva en todos los escenarios de aplicación posibles, por lo que deben seleccionarse adecuadamente de acuerdo al contexto en que se utilizan.

Una manera de validar la calidad de la partición obtenida por un algoritmo que detecte comunidades, es mediante la aplicación de dicho algoritmo sobre redes en las que se conozca su estructuración en comunidades. Luego, solo es necesario comparar la partición obtenida por el algoritmo con la que ya se conoce previamente. Esto se realiza mediante el uso de *medidas de similitud* entre particiones, también conocidas en el contexto de evaluación de calidad como *medidas extrínsicas*. Las medidas de evaluación extrínsicas requieren de una partición de referencia que represente el resultado deseado para poder compararlo con la partición obtenida. Usualmente, las particiones de referencia son definidas por especialistas. En la literatura se han propuesto distintas medidas extrínsicas para la evaluación de algoritmos de DCD [24,25,26]. Sin embargo, existen muy pocas medidas de similitud que tengan en cuenta el traslape entre las comunidades. Las dos medidas más utilizadas para la evaluación de algoritmos de OCD son una generalización de la NMI (Normal Mutual Information) presentada en [11] y la Omega Index [27]. Por otro lado, dada la similitud entre ambas tareas, las medidas extrínsicas propuestas para evaluar la calidad de los algoritmos que realizan agrupamiento con traslape [28,29,30,31] pueden ser utilizadas en el contexto de los algoritmos de OCD. De manera general, la principal limitación que presenta el uso de medidas extrínsicas para la

evaluación de calidad, está dada por la escasez de redes reales de gran tamaño de las cuales se conozca una partición de referencia.

2.2.1. Funciones de calidad de comunidades

Intuitivamente, la calidad de una partición puede ser medida a partir de la calidad de todas sus comunidades. En la literatura se han propuesto distintas *funciones de calidad de comunidad* (del inglés *community fitness function*) y estas tienen una gran utilidad en la detección de comunidades. Estas funciones, dado un subgrafo de la red, indican en qué medida el subgrafo cumple alguna propiedad inherente a las comunidades.

Una idea intuitiva, para indicar la calidad de una comunidad, puede ser formulada en función de los grados de los vértices que la componen, de manera que se mide la calidad de la comunidad como la diferencia entre su *grado interno* y su *grado externo*.

Definición 12 (Grado interno). *Sea C una comunidad y v un vértice de C. El grado interno de v con respecto a C se define como la cantidad de aristas que unen a v con el resto de los vértices de C. Se denota como* d_v^{int}. *La suma de los grados internos de todos los vértices de C se denota como* $d^{int}(C)$ *y se conoce como grado interno de C.*

Definición 13 (Grado externo). *Sea C una comunidad y v un vértice de C. El grado externo* k_v^{ext} *de v con respecto a C se definen como la cantidad de aristas que unen a v con los vértices que no pertenecen a la comunidad. Se denota como* d_v^{ext}. *La suma de los grados externos de todos los vértices de C se denota como* $d^{ext}(C)$ *y se conoce como grado externo de C.*

La suma del grado interno y el grado externo de una comunidad C se conoce como el *grado total* de la comunidad y se denota como $d_{total}(C)$.

Una idea más refinada consiste en utilizar distintas medidas de densidad asociadas a las comunidades, tales como *densidad interna*, *densidad externa* y *densidad relativa*

Definición 14 (Densidad interna). *Sea C una comunidad. La densidad interna de C, denotada como* $\delta_{int}(C)$, *se define como la razón entre el número de aristas entre los vértices de C y la mayor cantidad de aristas posibles en C, es decir:*

$$\delta_{int}(C) = \delta(C) = \frac{|E(C)|}{(|V(C)| * |V(C)-1|)/2}.$$

Definición 15 (Densidad externa). *Sea C una comunidad de una red G, la densidad externa de C respecto a G, denotada como* $\delta_{ext}(C)$, *se define como la razón entre el número de aristas que conectan a C con el resto de G y el máximo número de aristas posibles que conecten la comunidad con el resto de G, es decir:*

$$\delta_{ext}(C) = \frac{|E(G) - E(C)|}{|V(C)| * (|V(G)| - |V(C)|)}.$$

Definición 16 (Densidad relativa). *Sea C una comunidad, la densidad relativa de C, denotada como* $\rho(C)$, *se define como la razón entre el grado interno y el grado total de C, es decir:*

$$\rho(C) = \frac{d_{int}(C)}{d_{total}(C)}.$$

En trabajos recientes [32,33] se explica la necesidad de evaluar la calidad de las comunidades teniendo en cuenta tanto la estructura topológica como la información semántica asociada a los elementos que contienen. En vista de esto, se han propuesto distintas medidas de calidad de comunidades en las que se

analiza el grado de homogeneidad entre las aristas y/o la similitud entre los vértices, de manera que se pueda fundamentar la lógica en la formación de estos grupos. Por ejemplo en [33] se propone la medida *calidad lógica* que evalúa la calidad de una comunidad en función de qué tan diferentes son las aristas internas de dicha comunidad.

A lo largo de este trabajo se mencionan y describen brevemente algunas funciones de calidad de comunidades más complejas, que combinan las nociones de densidad, aislamiento y/o conectividad.

2.2.2. Modularidad

La medida modularidad fue propuesta en [22] como una función de calidad para seleccionar la mejor partición del dendograma generado por el algoritmo GN. Desde entonces, ha sido ampliamente utilizada en distintas tareas relacionadas con el estudio de las comunidades [3]. Por ejemplo, la optimización de esta medida constituye una metodología muy usada, sobre todo en el contexto de los algoritmos de DCD. Además, ha sido aplicada en la visualización de grafos [34] y para reducir el tamaño de grandes redes sin que estas pierdan su organización en grupos [35].

Como se ha explicado con anterioridad, no existe un criterio unificado de qué debe cumplir un subgrafo de una red para considerarse una comunidad de la misma. No obstante, si existe un consenso sobre qué tipo de grafos no tienen una estructuración interna basada en comunidades. La existencia de dichos módulos en una red, indica que existen grupos de vértices que tienden a interactuar más entre ellos que con el resto de los nodos. Por tal motivo, en una red donde la probabilidad de que un par de vértices estén conectados es la misma para todos, se considera que no existen comunidades. Los grafos aleatorios son un ejemplo de este tipo de redes, y han sido ampliamente usados en la descripción de sistemas complejos reales [3]. En el contexto de detección de comunidades este tipo de grafos se utiliza mayormente para definir redes que no se organizan en grupos y se conocen como modelos de grafo aleatorio nulo (MGAN). Un MGAN asociado a una red compleja G, debe compartir alguna propiedad estructural con respecto a G. El tipo de MGAN usado para definir la medida modularidad y el único que será abordado en este reporte, es un grafo aleatorio con la misma distribución de grados de los vértices que G.

En la figura 2 se muestra la diferencia estructural entre una red y un MGAN asociado a ella (grafo de la derecha). Evidentemente, cualquier partición del MGAN tendrá más aristas entre vértices pertenecientes a grupos diferentes que la partición intuitiva de la red. Esto ilustra la idea de que una red cuya estructura está basada en comunidades difiere notablemente de sus versiones aleatorias. En vista de ello, la idea básica de la modularidad consiste en medir qué tan diferente es la distribución de las aristas en la red original con respecto a una distribución aleatoria de la misma cantidad de enlaces en un MGAN asociado a la red.

De acuerdo al criterio de modularidad, un subgrafo s se define como comunidad de una red $G = \langle V, E, L, l \rangle$, si la cantidad de aristas entre los nodos de s es mayor que la esperada entre dicho grupo de vértices en un MGAN asociado a G. En vista de esto, la modularidad de una partición P de G se define mediante la expresión:

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

$m = |E|$, A_{ij} son los elementos de la matriz de adyacencia de G, d_i denota el grado del vértice i, mientras que $\delta(C_i, C_j)$ es una función binaria que indica si los vértices i y j pertenecen a la misma comunidad devolviendo el valor 1 en caso afirmativo y 0 en caso contrario.

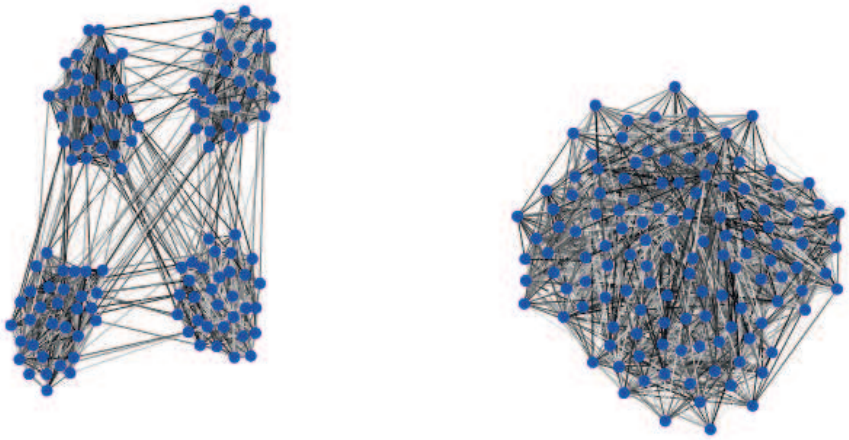

Fig. 2. Ejemplo de una red (izquierda) y un MGAN asociado a ella (derecha).

El sumando $\frac{1}{2m}\sum_{ij} A_{ij}\sigma(C_i,C_j)$ indica las aristas existentes entre los vértices pertenecientes a una misma comunidad. Mientras que el sumando $\frac{1}{2m}\sum_{ij} \frac{d_i d_j}{2m}\sigma(C_i,C_j)$ indica la cantidad de aristas esperada en una misma comunidad en un MGAN asociado a la red original.

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):

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

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.

Aunque la medida modularidad fue inicialmente propuesta para el caso de redes simples, no dirigidas y no ponderadas, se han presentado distintas generalizaciones de la misma para su aplicación en tipos de redes más complejas. En [36] se propone una extensión de esta medida que permite tener en cuenta el peso de las aristas en redes ponderadas. También se ha modificado la modularidad para tener en cuenta la dirección de las aristas en redes dirigidas [35,37]. Por otro lado, en la definición inicial de modularidad se requiere que cada vértice pertenezca a una única comunidad, por lo que no se puede evaluar la calidad de cubrimientos. En la sección 3.3 se abordan las principales generalizaciones de esta medida que tienen en cuenta el traslape entre las comunidades.

Límites de modularidad

Una de las principales dificultades de la modularidad es que no siempre se cumple que las particiones con mayores valores de esta medida, son las particiones de mejor calidad. Este problema es una consecuencia directa de una propiedad inherente al MGAN usado en la definición de modularidad. Dicha propiedad se conoce como límite de resolución [1]. Para calcular la cantidad de aristas que se espera existan en cada grupo de vértices del MGAN, se sigue un enfoque global y se asume que todos los pares de vértices pueden estar conectados con la misma probabilidad. Este es un modelo poco realista ya que en redes de gran tamaño los vértices limitan su interacción a solo una parte de la red.

El límite de resolución, se evidencia en mayor medida a la hora de determinar si se deben mezclar dos grupos de vértices para mejorar la calidad de la partición. El número esperado de aristas en MGAN entre dos grupos de vértices C_1 y C_2 es igual a $\frac{d_{total}(C_1)d_{total}(C_2)}{2m}$. Teniendo en cuenta esto, la variación de modularidad resultante de mezclar a C_1 y C_2, con respecto a la partición donde estos grupos constituyen comunidades independientes, se puede determinar mediante la expresión: $\Delta Q_{C_1C_2} = \frac{E_{C_1C_2}}{m} - \frac{d_{total}(C_1)d_{total}(C_2)}{2m^2}$, siendo $E_{C_1C_2}$ la cantidad de aristas que conectan a C_1 y C_2. Teniendo en cuenta esto, si los grupos de vértices son pequeños en cuanto a grado total, al menos con respecto a la cantidad de aristas de la red, entonces la cantidad de aristas esperadas entre ambos grupos va a ser muy pequeña. En particular, si se cumple que $d_{total}(C_1)d_{total}(C_2) \leq 2m$ entonces basta con que los grupos estén conectados por una arista, para que la mezcla entre los dos grupos incremente la modularidad de la partición.

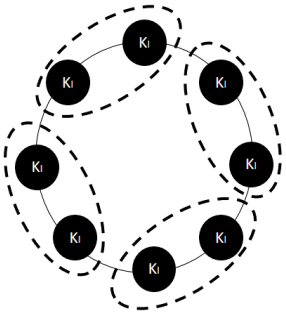

Fig. 3. Ejemplo de repercusión del límite de resolución presente en la modularidad [1].

Debido al límite de resolución, las técnicas basadas en optimización de modularidad pueden fallar en la detección de comunidades cuyo tamaño sea menor que cierto umbral de resolución. Lo que puede incidir negativamente en la efectividad de estos algoritmos sobre redes reales, ya que estas redes se caracterizan por contener comunidades de distintos tamaños. De igual modo, esta medida no debe usarse para comparar la calidad de la estructuración en comunidades de redes de tamaños diferentes.

En vista de lo explicado anteriormente, la modularidad cuenta con un umbral de resolución implícito que restringe el número y el tamaño de cada comunidad. En función de esto, durante la formación de las comunidades que maximizan la modularidad, pueden mezclarse cliques bien delimitados entre sí o dividirse cliques de gran tamaño, lo que intuitivamente son errores que afectan la calidad del agrupamiento.

En la figura 3 se muestra un ejemplo de cómo puede incidir el límite de resolución en las técnicas basadas en la optimización global de modularidad. Cada circulo K_l representa un clique de l aristas, y los cliques se relacionan entre sí mediante una única arista tal como se indica en la figura. Intuitivamente, la partición de mayor calidad en esta red, se obtiene considerando cada clique como una comunidad independiente. No obstante, si el límite de resolución determinado por el tamaño de la red es mayor que l, entonces la partición de máxima modularidad de la red, está conformada por comunidades que incluyen dos o mas cliques, como los grupos indicados por las líneas discontinuas.

2.3. Relaciones jerárquicas entre comunidades

En muchas redes reales las comunidades pueden ser organizadas de forma jerárquica [38], ya que las comunidades pequeñas suelen combinarse para formar comunidades más complejas. No obstante, la mayoría de los algoritmos que detectan comunidades, identifican solamente la partición de la red que optimiza determinado criterio. Idealmente, los algoritmos de detección de comunidades deben saber determinar si existen o no relaciones jerárquicas existentes entre las comunidades y en caso afirmativo, deben ser capaces de detectarlas [3].

2.3.1. Algoritmos jerárquicos

La manera natural de detectar relaciones jerárquicas, es a partir de algoritmos que siguen un enfoque similar al usado en los algoritmos de agrupamiento jerárquicos tradicionales. De forma similar a estos algoritmos tradicionales, los relaciones existentes entre las distintas particiones detectadas suelen ser representadas mediante dendogramas.

Los algoritmos jerárquicos tienen la ventaja de que no dependen de parámetros relacionados al tamaño y a la cantidad de comunidades que se desean detectar. No obstante, los algoritmos de este tipo presentan una serie de dificultades que dificultan su uso. Por un lado, la mayoría de estos algoritmos pueden generar dendogramas con gran número de particiones, por lo que requieren de alguna medida de calidad para identificar las particiones relevantes. Estas medidas de calidad no tienen la misma efectividad en todos los tipos de redes y escenarios en donde se aplican estos algoritmos, por lo que la selección adecuada de estas medidas requiere de conocimiento previo del entorno de aplicación. Por otro lado, en algunos casos los algoritmos realiza una incorrecta asignación de los vértices a las comunidades. Por ejemplo, los vértices que sólo tienen un vecino son frecuentemente clasificados como comunidades unitarias. Además, estos algoritmos construyen siempre una jerarquía aún cuando no existan relaciones jerárquicas entre las comunidades de la red, lo que puede traer consigo que algunos vértices sean clasificados incorrectamente.

2.3.2. Algoritmos multiresolución

La presencia del límite de resolución en la modularidad (ver sección 2.2.2) es un ejemplo de como los algoritmos basados en la optimización de medidas de calidad pueden dejar de detectar comunidades cuyo tamaño esté fuera de determinado rango. Los métodos que siguen este enfoque deben ser capaces de explorar todos los posibles niveles de resolución para cubrir todas las comunidades de la red, independientemente del tamaño de las mismas. Se considera que a mayor resolución menor es el tamaño de la comunidad. Usualmente esto se logra a partir de un parámetro de resolución variable que afecte la medida de calidad en función de controlar el tamaño característico de las comunidades que se detectan. Se deben realizar distintas ejecuciones del algoritmo con distintos valores del parámetro de resolución para cubrir todas las escalas de resolución posible. Los algoritmos que siguen esta metodología se conocen como algoritmos multiresolución.

Los algoritmos multiresolución también son capaces de detectar las relaciones jerárquicas de la red. Esto se logra formando el dendograma con las particiones obtenidas a distintas escalas de resolución, de manera que los niveles inferiores del dendograma se asocian a escalas menores y los niveles superiores a escalas mayores. Las relaciones jerárquicas se establecen a partir de las correspondencias existentes entre las comunidades de niveles consecutivos del dendograma.

El principal problema con este tipo de algoritmo es que son muy sensibles respecto al parámetro de resolución. Si el parámetro es muy grande pueden dividirse comunidades o cliques de gran tamaño, mientras que con valores muy pequeños de resolución las comunidades de menor tamaño suelen mezclarse, incluso cuando no existe gran interacción entre dichos grupos. El correcto ajuste de este parámetro requiere de un conocimiento previo de las características de la red por parte del usuario y esto no siempre se garantiza en problemas reales. Además, para poder cubrir todas las escalas de resolución posible, es necesario ejecutar el algoritmo en reiteradas ocasiones, lo que puede limitar la aplicación del algoritmo en redes de gran tamaño.

2.4. Detección de comunidades en redes dinámicas

Existen dos tipos de enfoques para el análisis estructural de las redes complejas. El enfoque estático se encarga de analizar los vértices y aristas de la red en un momento dado del tiempo. Este enfoque está dirigido a descubrir patrones estructurales en una configuración específica de los componentes de la red en un momento dado. Estas configuraciones se conocen como una *snapshot* o *instantánea* de la red. Por otro lado, el enfoque dinámico se centra en el estudio de *redes dinámicas*, las cuales son redes que evolucionan en el tiempo. En este tipo de enfoque se analiza la red observada en múltiples instantes de tiempo y se centra en revelar los mecanismos evolutivos y los fenómenos dinámicos de la red que inciden en su estructura.

La mayoría de los trabajos existentes para detectar comunidades siguen un enfoque estático. Esto se debe principalmente a dos factores. El primero es el hecho de que la detección de comunidades de redes complejas desde una perspectiva estática, sigue siendo sumamente compleja y todavía se dedican muchos esfuerzos al perfeccionamiento de los algoritmos dedicados a esta tarea. El segundo es que solo recientemente, con el perfeccionamiento de las tecnologías de la información, se ha incrementado la capacidad de generación y almacenamiento de redes *etiquetadas temporalmente*.

Debido a que la mayoría de las redes reales están constantemente sometidas a cambios en su estructura, se ha evidenciado un creciente interés en el análisis estructural de las redes dinámicas. Muchos de los estudios enfocados en este tipo de redes se dedican a monitorear los principales fenómenos que involucran a las comunidades. Estos fenómenos se pueden clasificar de acuerdo al tipo de transformación que sufre la comunidad en: *nacimiento*, *crecimiento*, *fusión*, *contracción*, *división* y *muerte*, los cuales se ilustran en la figura 4. Asociado al nacimiento y la muerte de una comunidad surge la noción de *edad* de una comunidad. Se dice que la *edad de una comunidad* es el tiempo transcurrido desde que se forma la comunidad hasta que deja de serlo. Usualmente, la edad de una comunidad es proporcional a su tamaño [2].

El modelo más utilizado para representar redes dinámicas consiste en definir una red dinámica G como una secuencia de instantáneas $\{G_1, G_2, \ldots, G_n\}$, siendo G_t con $1 \leq t \leq n$, la instantánea que muestra la configuración de G en el instante indicado por la etiqueta de tiempo t. Para una mejor descripción de la red, cada instantánea G_t puede registrar en forma de etiquetas de tiempo la información asociada al tiempo de creación de cada una de sus aristas, incluso en algunas redes se registra también el instante en que desaparece la arista. Este último caso puede ser de gran utilidad en redes donde exista un gran flujo de interacción entre sus entidades, como por ejemplo las redes de comunicaciones o redes metabólicas, en las cuales las aristas entre un mismo par de vértices se pueden establecer en distintos intervalos de tiempo.

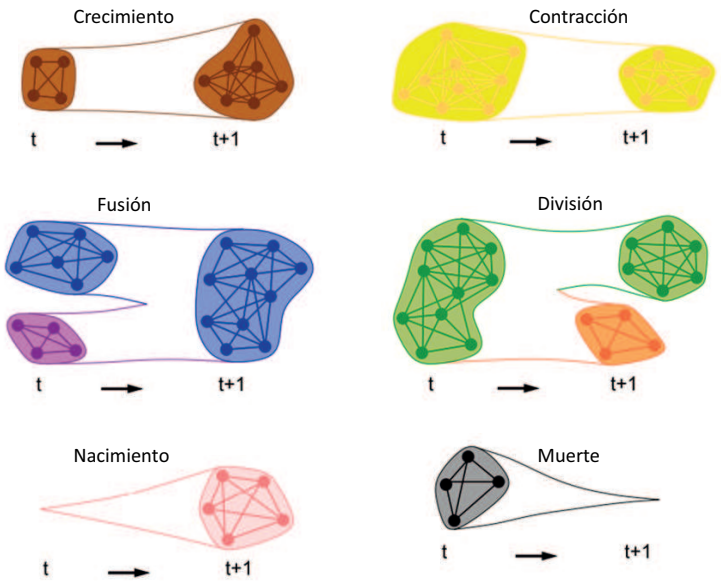

Fig. 4. Ejemplo de transformaciones que pueden sufrir las comunidades. Imagen tomada de [2].

No obstante, en múltiples trabajos [39,40,41,42] se ha mostrado que en la mayoría de las redes reales las relaciones entre los elementos de la red tienden a incrementarse y a mantenerse a lo largo del tiempo. Por tal motivo, el análisis de los cambios asociados a la eliminación de aristas es omitido en dichos trabajos, disminuyendo notablemente la complejidad del problema.

Una de las principales dificultades de representar una red dinámica como una secuencia de instantáneas, radica en la correcta selección de los instantes de tiempo en que se obtendrán las instantáneas. Si se obtienen instantáneas de la red cada pequeños intervalos de tiempo, entonces las instantáneas consecutivas pueden ser muy similares, por lo que la información asociada a estas puede resultar redundante y se realizarán múltiples operaciones innecesarias. Por otro lado, si en el intervalo de tiempo entre dos instantáneas consecutivas ocurren muchas transformaciones en la red, entonces puede interpretarse de manera incorrecta la evolución de las comunidades desde un instante a otro.

La idea utilizada como base en muchos de los trabajos que detectan comunidades en redes dinámicas, consiste en dos fases fundamentales. En la primera fase se identifican las comunidades de cada una de las instantáneas de la red. Esta fase resulta sumamente costosa, ya que se analiza toda la estructura de la red en múltiples ocasiones, al menos una vez por cada instantánea, pero este número se puede incrementar dependiendo del tipo de algoritmo utilizado, un ejemplo de esto es cuando se usan algoritmos multiresolución. En este sentido, se han propuesto distintas heurísticas que utilizan los resultados obtenidos en instantáneas anteriores, para detectar las comunidades en las siguientes instantáneas, sin tener que procesar nuevamente la red en su totalidad. En la segunda fase se obtienen las relaciones existentes entre las particiones detectadas en la primera fase. Sean C_t y C_{t+1} comunidades pertenecientes al par de instantáneas consecutivas G_t y G_{t+1}, respectivamente. Entre C_t y C_{t+1} se establece una relación si C_{t+1} puede interpretarse como la evolución de C_t en el intervalo de tiempo $[t,t+1]$. Un razonamiento intuitivo para llevar a cabo este tipo de asociación es mediante el uso del traslape relativo (ver sección 2.2), asociando a C_{t+1} la comunidad C_t con la cual tenga un mayor traslape relativo. No obstante, esta estrategia puede fallar al asociar comunidades que no reflejan correctamente la transformación ocurrida. Por ejemplo, sea χ_t una comunidad de G_t disjunta con respecto a C_t y sea χ_{t+1} su comunidad asociada en G_{t+1}, si el grado

de traslape entre C_{t+1} y χ_{t+1} es muy alto, entonces puede pasar que C_{t+1} tenga un mayor traslape relativo con χ_t que con C_t. Para tener en cuenta este aspecto y muchos otros, se han propuesto distintas estrategias para obtener estas asociaciones. En [3] se describen las principales estrategias propuestas en la literatura.

Otra manera de representar las redes dinámicas es mediante una configuración inicial de la red y una secuencia de cambios sobre la red, los cuales pueden ser tanto de eliminación como de inserción de vértices y/o aristas. Este modelo de representación se conoce como *modelo incremental de cambio*. Usando este modelo de representación se puede llevar a cabo un análisis más detallado de las trasformaciones que sufre la red y de la manera en que estas influyen en la evolución de las comunidades. Las redes reales se caracterizan por sufrir constantes cambios estructurales, por lo que el número de transformaciones en la secuencia de cambios suele ser muy grande. La mayoría de los algoritmos que siguen este enfoque presentan la dificultad de que actualizan las comunidades de la red, por cada una de las transformaciones de la secuencia de cambios. No obstante, estos algoritmos son capaces de obtener las comunidades de la red modificada por los cambios, a partir de las comunidades detectadas en configuraciones anteriores de la red.

3. Algoritmos para la detección de comunidades con traslape

La detección de comunidades con traslape en redes complejas es una línea de investigación de gran interés en la comunidad científica y cuenta con numerosas propuestas de solución. Esto dificulta la selección de los algoritmos idóneos para los distintos escenarios donde esta tarea resulta de interés. Por tal motivo, ha surgido la necesidad de establecer distintas formas de clasificar metodológicamente los algoritmos existentes.

En este trabajo, los algoritmos son clasificados de manera similar a la propuesta por Fortunato [3], que consiste en agrupar los algoritmos de acuerdo a la metodología que siguen para detectar las comunidades. Aunque de esta forma algunos métodos pueden ser ubicados en múltiples categorías, se propicia un mejor entendimiento de los distintos enfoques y el proceso de refinamiento en cada una de sus técnicas. Otros tipos de clasificación pueden verse en [14,43].

Por cada uno de los enfoques analizados, se describe el tipo de redes que es capaz de procesar, la forma en que definen las comunidades que detectan y se realiza un análisis crítico de las limitaciones que presentan.

3.1. Algoritmos basados en la conexión entre comunidades

Como se ha explicado con anterioridad, los vértices pertenecientes a una comunidad mantienen una gran interacción entre ellos y pueden estar relacionados, en menor medida, con vértices de otras comunidades, estableciendo vínculos entre los distintos grupos de la red. Intuitivamente, una forma de detectar las comunidades consiste en detectar los vértices y aristas que las mantienen conectadas. Esa es la filosofía de los algoritmos descritos en esta sección.

3.1.1. Familia de algoritmos GN.

El algoritmo de Girvan y Newman (GN) [15] es uno de los métodos de detección de comunidades más conocidos. Desde su presentación en 2002, se han desarrollado varios algoritmos que buscan extenderlo con el objetivo de optimizar su eficiencia y adaptarlo a requerimientos más complejos. En esta sección se

explica la filosofía del algoritmo GN y de aquellas extensiones del mismo, que se han propuesto para la detección de comunidades con traslape.

Esta familia de algoritmos sigue una metodología que consiste básicamente en un proceso iterativo donde en cada paso, se detecta y elimina de la red un elemento de la misma (vértice o arista), que participe en la conexión entre comunidades diferentes. El paso iterativo se realiza hasta que se hayan eliminado todos los elementos o se haya alcanzado una partición de buena calidad (ver sección 2.2). Este enfoque es similar al agrupamiento jerárquico divisivo y el resultado final del proceso se representa con un dendograma como usualmente se realiza en este tipo de algoritmos.

La principal distinción entre los algoritmos de esta familia, es el criterio utilizado para seleccionar el elemento de la red que se elimina en cada paso. Generalmente, esta selección se basa en los valores que tiene cada elemento respecto a alguna medida de *centralidad*. La *centralidad* de un elemento es una propiedad que indica de forma aproximada, la influencia del elemento en la conexión de la red.

La idea general del algoritmo GN consiste básicamente en la eliminación iterativa de la arista de la red con mayor valor de centralidad. La estructura de control del mismo se puede resumir en los siguientes pasos.

1. Estimar la centralidad de todas las aristas de la red.
2. Seleccionar una arista con el mayor valor de centralidad y eliminarla de la red.
3. Repetir el procedimiento para la red modificada.

Para estimar la centralidad de una arista, en el algoritmo GN se proponen tres medidas: la *intermediación de aristas*, la *intermediación de recorrido aleatorio de aristas* y la *intermediación de flujo de corriente de aristas*. Estas medidas fueron expuestas desde perspectivas diferentes pero están basadas en la misma idea. Esta idea plantea que todos los caminos, entre los vértices de dos comunidades diferentes, deben pasar por las aristas que conectan tales comunidades. Estas aristas son conocidas como aristas puentes o inter-grupos. Como generalmente la cantidad de enlaces entre comunidades es muy pequeña, es lógico asumir que las aristas puentes están contenidas en una gran cantidad de caminos entre los vértices de la red.

La *intermediación de aristas* (del inglés *edge betweenness*) es la más simple de las medidas propuestas por Girvan y Newman. Esta es una extensión para el caso de las aristas, del concepto de intermediación de vértices (del ingles vertex betweenness) presentado por Freeman[44]. La intermediación de una arista indica la cantidad de caminos de longitud mínima que existen entre todos los pares de vértices de la red, que contienen a dicha arista. Las aristas puentes tienen valores elevados de intermediación. En la figura 5 se muestra como todos los caminos de longitud mínima entre los vértices de la comunidad A y los de la B contienen la arista puente entre estas comunidades. En una red con m aristas y n vértices, la intermediación de todas las aristas se puede calcular con un costo temporal $O(mn)$ [22].

La *intermediación de recorrido aleatorio (IRA) de aristas* (del inglés *random walk betweenness*) puede ser vista en el contexto de procesos tales como el envío de señales en la red. Si se desea enviar una señal desde un nodo a otro, esta no tiene que viajar necesariamente por caminos de longitud mínima. Este proceso puede ser modelado a partir de recorridos aleatorios. Para calcular la IRA de una arista, se halla la probabilidad de que esté contenida en un recorrido aleatorio para cada uno de los posibles pares de vértices y se promedian esos valores. Esta medida puede ser calculada para todas las aristas de la red con un costo temporal de $O[(m+n)n^2]$ [22].

La *intermediación de flujo de corriente (IFC) de aristas* (del inglés *current flow edge betweenness*) se basa en el funcionamiento elemental de un circuito eléctrico. En tal sentido la red se modela como un circuito en el cual las unidades de resistencia son representadas por las aristas de la red. De esta manera,

si se aplica un cambio de voltaje entre un par de vértices, por cada arista corre cierta cantidad de corriente. Para calcular este tipo de intermediación, se realiza un cambio de voltaje entre todos los posibles pares de vértices y se promedian la cantidad de corriente que pasó por cada arista. Se puede comprobar que esta medida es equivalente a la IRA [22] y por tanto, su cálculo tiene el mismo costo temporal que la anterior.

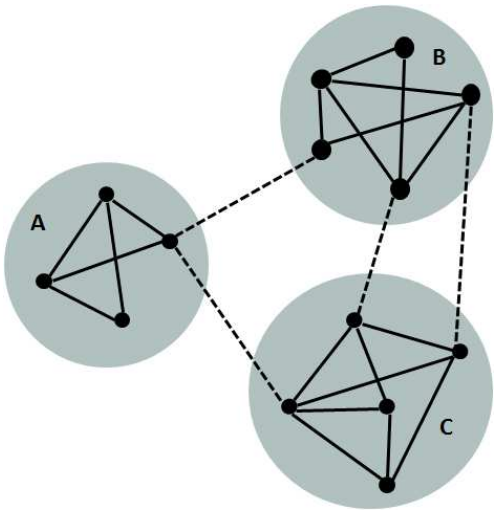

Fig. 5. Las aristas puentes de la figura 1 se muestran en líneas discontinuas. Al eliminarlas, las tres comunidades se separan y se obtiene la partición deseada.

Los caminos de longitud mínima entre los vértices de dos comunidades diferentes pasan por las distintas aristas puentes que unen estas comunidades. La distribución de la cantidad de caminos de longitud mínima por cada una de esas aristas puentes no tiene porque ser equitativa y algunas pueden tener valor de centralidad muy pequeños. Sin embargo, si la arista de mayor centralidad es una arista puente, al eliminarla de la red, los vértices que estaban conectados por los caminos que la contenían, ahora se conectarán usando alguna de las aristas puentes restantes, aumentando la centralidad de las mismas. Debido a esto, una vez que la arista de mayor centralidad es eliminada de la red, la centralidad de las aristas restantes necesita ser actualizada.

El paso de recalcular la centralidad solo resulta necesario para las aristas de la componente conexa que contenía a la última arista eliminada, ya que las aristas solo pueden participar en caminos entre vértices de su componente conexa. Por tal motivo, en redes que contengan numerosas comunidades y el algoritmo divida rápido la red en múltiples componentes conexas, el costo temporal se alivia mucho. No obstante, el algoritmo sigue siendo costoso y poco útil en problemas reales que trabajen con redes de gran tamaño.

De esta forma, si GN usa intermediación de aristas, su complejidad computacional es de $O(m^2n)$ o $O(n^3)$ si la red es dispersa. Los mejores resultados experimentales del algoritmo GN han sido obtenidos usando esta medida de centralidad [22]. Además, esta puede ser extendida al trabajo con redes ponderadas. Newman define en un trabajo posterior [45] la intermediación de una arista ponderada como la razón entre la intermediación de la arista en la red no ponderada asociada y el peso de la arista.

En [22], Girvan y Newman proponen una medida llamada *modularidad* (ver sección 2.2.2)) que permite seleccionar del dendograma obtenido por el algoritmo GN, el nivel o conjunto de comunidades de mejor calidad, es decir, que refleja de una mejor manera las comunidades existentes en la red.

La manera en que se estima la centralidad de los elementos de la red resulta una componente crítica en la eficiencia y el correcto funcionamiento de los algoritmos que se basan en la conexión entre comunidades.

Girvan y Newman proponen calcular la intermediación de las aristas mediante la realización de recorridos BFS a partir de cada vértice de la red. De esta manera, se obtienen los caminos de longitud mínima que se originan en cada uno de los vértices. Durante los recorridos, se registran por cada arista la cantidad de veces que aparece en un camino de longitud mínima. Realizar este proceso en cada paso iterativo del algoritmo constituye una sobrecarga temporal que hace impracticable la ejecución del algoritmo en muchas aplicaciones reales.

En [46], se plantea la idea de que no es necesario llevar a cabo este proceso para todos los vértices de la red. En este trabajo, se propone realizar ese proceso solo a un subconjunto aleatorio de vértices, y estimar la intermediación de las aristas usando una simulación Monte Carlo. Los valores de intermediación obtenidos pueden contener algunos errores, dependiendo de la cantidad de vértices seleccionados para iniciar los recorridos. De esta manera, se establece una relación entre eficiencia y eficacia del algoritmo. No obstante, los resultados experimentales mostrados en [46], muestran que seleccionando por cada componente conexa de la red, una cantidad de vértices proporcional al logaritmo de la cantidad de vértices de la componente, se puede obtener un balance entre rapidez del algoritmo y calidad en las particiones obtenidas. Este algoritmo es no determinista, ya que para diferentes selecciones de vértices se pueden obtener particiones diferentes. Debido a esto, los vértices pueden estar asociados a comunidades diferentes para distintas ejecuciones del algoritmo. No obstante, en la experimentación realizada solamente ocurre este caso con los vértices que participan en la conexión entre comunidades. Esto es realmente una buena característica del método, ya que permite identificar el traslape existente entre las comunidades.

Existen varios algoritmos que han extendido el algoritmo GN, con el objetivo de poder detectar comunidades entre las cuales pueda existir el traslape [47,48]. Estos algoritmos siguen la misma estructura de control que GN pero a diferencia de este último, en el paso 2 también se tienen en cuenta la centralidad de los vértices. Los vértices de mayor centralidad son divididos en diferentes copias de manera que cada una esté asociada a una comunidad diferente. En este contexto, la división de estos vértices se interpreta como que las comunidades resultantes se mantienen conectadas mediante el vértice dividido.

En [47], se propone el algoritmo CONGA (Cluster-Overlap Newman Girvan Algorithm). En este algoritmo la centralidad de los vértices se calcula mediante la *intermediación* de vértices propuesta en [44]. La intermediación de un vértice indica la cantidad de caminos de longitud mínima que contienen dicho vértice. En cada iteración del algoritmo se decide qué operación realizar a partir de la centralidad de los elementos. Posteriormente, si la centralidad de un vértice es mayor que la máxima centralidad de las aristas, este es dividido en dos y se reparten sus aristas entre las dos copias. Para determinar que aristas se quedan en cada copia del vértice, se utiliza una medida llamada *intermediación de separado* o *split betweenness*. Dado un vértice v y una bipartición $\{n_1, n_2\}$ de los vértices adyacentes de v, la intermediación de separado (IS) de v se define como la cantidad de caminos de longitud mínima que unen vértices de n_1 con vértices de n_2 y que además contienen a v. Por otro lado, si el elemento de mayor centralidad es una arista entonces se procede de manera similar al algoritmo GN. La mayor limitación de CONGA de forma similar al algoritmo GN es su alta complejidad computacional lo cual lo hace poco útil para problemas reales. Esta complejidad es de $O(m^3)$ o $O(n^3)$ para redes dispersas.

Con el objetivo de optimizar los cálculos del algoritmo CONGA, en [48] se propone el algoritmo CONGO (CONGA Optimized). Este algoritmo usa un enfoque local para el cálculo de la intermediación de los vértices y aristas. Bajo este enfoque, no se analizan todos los caminos de longitud mínima, sino que se procesan solamente los caminos de longitud mínima cuya longitud sea menor que un parámetro h, previamente definido. Aunque CONGO resulta considerablemente más eficiente que el algoritmo CONGA, incorpora el uso de un parámetro que depende de la colección a procesar. Asignar valores adecuados a este parámetro puede resultar difícil en problemas reales, en los cuales, por lo general, el usuario desconoce

las características de los datos. La complejidad temporal de CONGO es $O(m \log m + m^{2h+2}/n^{2h+1})$ pero puede llegar a ser $O(n \log n)$ para redes dispersas.

En cuanto a la efectividad de estos algoritmos, tanto CONGO como CONGA tienden a ubicar vértices en más comunidades de lo que realmente les corresponde. Esto se debe a la operación de división que se le realiza a los vértices de alta centralidad. El número de vértices mal ubicados puede llegar a ser elevado, lo que puede complejizar la interpretación del traslape existente entre las comunidades detectadas.

3.1.2. Otros enfoques basados en centralidad.

Una de las mayores desventajas de los algoritmos de la familia GN, es que estos tienen que actualizar la centralidad de todos los elementos, después de cada cambio realizado en la red. Los algoritmos que se describen a continuación analizan la centralidad de los vértices una única vez, a partir de la información de sus vecinos. Este análisis de la centralidad se realiza con el objetivo de identificar el rol que desempeña cada vértice en la estructura de las comunidades que lo contienen.

En [49] se propone un algoritmo de OCD que define las comunidades mediante la combinación de los grupos obtenidos por cada nodo, desde una perspectiva local. En este algoritmo la centralidad de un nodo se determina por la cantidad de componentes conexas que se obtienen al eliminar al nodo de su egonet. Los grupos resultantes de incorporar el nodo en cada una de esas componentes conexas, se conocen como los *grupos de amigos* de dicho nodo. La formación de las comunidades de este algoritmo se basa en la idea de que si un vértice pertenece a una comunidad entonces uno de sus grupos de amigos también pertenecerá a dicha comunidad. En vista de esto, después de determinar los grupos de amigos de cada uno de los vértices, se realiza un proceso iterativo donde estos se combinan para formar las comunidades. Este proceso culmina cuando ya no se pueda mezclar ningún par de grupos. Dos grupos de amigos pueden ser mezclados si todos los vértices del grupo de menor tamaño, con excepción de a lo sumo uno, pertenecen al grupo de mayor tamaño.

La complejidad computacional de este algoritmo es $O(n^3)$, aunque puede reducirse a $O(n^2 \log n)$ para redes dispersas. No obstante, el algoritmo puede ser fácilmente paralelizado gracias a la independencia entre las operaciones que realiza. Por otro lado, no se presentaron resultados comparativos con ningún otro algoritmo del estado del arte, por lo que resulta difícil comprobar la efectividad del algoritmo.

Otro ejemplo de algoritmo de OCD basado en centralidad es FRINGE [50] (Friendship Networks with General Elements). Este algoritmo está basado en la idea de amistad entre los miembros de las redes sociales y la noción de liderazgo de algunos de estos miembros. La estructura general de FRINGE de puede resumir de la siguiente manera. Inicialmente, se seleccionan los vértices líderes de la red. Luego, por cada uno de los vértices líderes obtener su comunidad. Finalmente, si alguna de las comunidades detectadas en la fase anterior está contenida en otra se mezclan ambas comunidades.

En FRINGE, el liderazgo se determina a partir de una medida de centralidad denominada *grado extendido*. Normalmente, en redes sociales los vértices con mayor grado tienen mayor influencia en la red. El *grado extendido* de un nodo se basa en la idea de extender su influencia en la red, a partir de la influencia de sus vértices adyacentes. En vista de esto, el grado extendido de un nodo se determina como la suma de su grado y los grados de sus nodos vecinos. Luego, la selección de los vértices líderes de la red consiste en obtener iterativamente el vértice de mayor grado extendido que no haya sido seleccionado como líder en iteraciones anteriores. Este proceso iterativo finaliza cuando la diferencia entre los grados extendidos del próximo líder a seleccionar y del último seleccionado, sea mucho mayor que la diferencia entre los dos últimos vértices seleccionados. Esta restricción permite establecer cierto grado de homogeneidad entre los líderes.

Por cada uno de los líderes determinados en la primera fase, se crea una comunidad a partir de la relación que mantienen estos con el resto de los vértices de la red. Si un vértice es adyacente a un líder,

entonces pertenece a la comunidad asociada a ese líder. Posteriormente, se realiza un proceso iterativo que consiste en asignar en la i-ésima iteración, a vértices que no hayan sido asignados anteriormente a ninguna comunidad y que estén a una distancia i de algún líder. Este proceso finaliza cuando todos los vértices han sido asignados a alguna comunidad. Con el objetivo de asegurar que las comunidades sean densas, se establece un criterio cuantitativo para restringir la asignación de los vértices a las comunidades. Esta restricción consiste en que un vértice v es asignado a una comunidad C solamente si satisface que la suma entre la cantidad de vecinos de v que ya han sido asignados a alguna comunidad y la cantidad de vecinos de v que pertenecen a C, sea mayor o igual que el doble de la mayor cantidad de vecinos de v existentes en una misma comunidad. Las comunidades resultantes de este proceso satisfacen la condición de n-cliques (ver sección 2.1), siendo n el doble de la mayor distancia entre un líder y un vértice del resto de la red. La complejidad computacional de FRINGE es $O(n^2)$.

Una de las principales limitaciones de FRINGE es que depende mucho de la selección de los líderes, por ejemplo cuando se seleccionan líderes pertenecientes a una misma zona densa de la red, entre las comunidades obtenidas por FRINGE para cada uno de estos líderes puede existir un alto nivel de traslape. Además, si los líderes de las comunidades no son verdaderamente representativos de estas, entonces los vértices que conectan comunidades diferentes suelen ser asignados a comunidades incorrectas.

3.2. Algoritmos basados en el desplazamiento de cliques

Como se ha explicado anteriormente, las comunidades son grupos de gran densidad interna que tienen muy poca interacción con el resto de la red. En vista de esto, una comunidad puede ser vista estructuralmente como la unión de subgrafos densos que deben compartir muchos vértices en común, ya que si fueran disjuntos entre sí, sería mejor dividir la comunidad en varias comunidades más pequeñas. Intuitivamente, si un clique pudiera ser *desplazado* sobre la red de alguna manera, este quedaría atrapado en la comunidad que lo contiene, debido a la poca densidad de aristas que existen entre comunidades. En esta sección se describen los algoritmos que se basan en esta idea. Primeramente, se realiza un análisis del tipo de comunidades que definen estos algoritmos y luego, se describe el algoritmo CPM y todas sus extensiones.

3.2.1. Comunidades k-clique

En este tipo de algoritmos se sigue un enfoque puramente local y topológico para definir las comunidades, utilizando el concepto de *k-clique*. Un k-clique es un subgrafo completo de k vértices. La noción de desplazamiento de un k-clique se define a partir de la adyacencia entre k-cliques. Dos k-cliques son adyacentes si tienen en común $k-1$ vértices. La unión de k-cliques adyacentes se conoce como *camino de k-cliques*. Si dos k-cliques pertenecen al mismo camino de k-cliques entonces se dice que están conectados. Un k-clique se desplaza sobre la red cuando rota hacia uno de sus k-cliques adyacentes sobre los $k-1$ vértices que tienen en común.

En la figura 6 se muestra un ejemplo de cómo se desplaza un 4-clique mediante sus 4-cliques adyacentes. El subgrafo de la red en el cual se encuentra el 4-clique que se va a desplazar, se resalta con el color azul. De igual modo, se indican en azul el resto de los vértices de la red que han sido alcanzados por el desplazamiento. La flecha representa el desplazamiento del 4-clique hacia uno de sus 4-cliques adyacentes, de manera que, el nuevo vértice que será alcanzado en el desplazamiento se indica en rojo.

En función de los conceptos introducidos en lo anterior, una *comunidad k-clique* se puede definir como el subgrafo maximal formado por la unión de un k-clique y todos los k-cliques conectados a él. Una manera equivalente de ver este tipo de comunidades es usando la noción de desplazamiento de cliques, de manera que una comunidad k-clique se determina por todos los vértices que pueden ser alcanzados en el

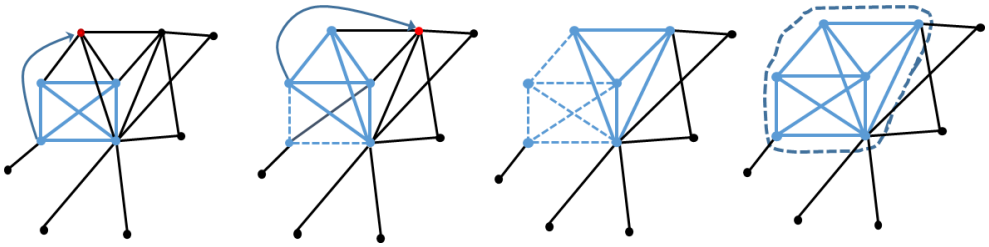

Fig. 6. Ejemplo de desplazamiento de un 4-clique.

desplazamiento de un k-clique.

El grado de restricción impuesto por la definición de k-clique, puede incidir notablemente en el número de comunidades detectadas. La figura 7 muestra ejemplos de subgrafos densos (7.a y 7.d), que no son considerados comunidades usando la definición basada en k-cliques. Para k = 5, el subgrafo 7.a no se considera una comunidad por la ausencia de una arista. Los 5-cliques 7.b y 7.c, no se consideran adyacentes y no pueden formar la comunidad 7.d), porque solo tienen tres (k-2) vértices en común. No obstante, regulando el valor de *k* se puede flexibilizar esta restricción. Por otro lado, el uso de esta definición provoca que exista una gran homogeneidad estructural entre las comunidades detectadas. Esto dificulta la interpretación de las comunidades en tareas donde es necesario definir la función de cada elemento en la comunidad.

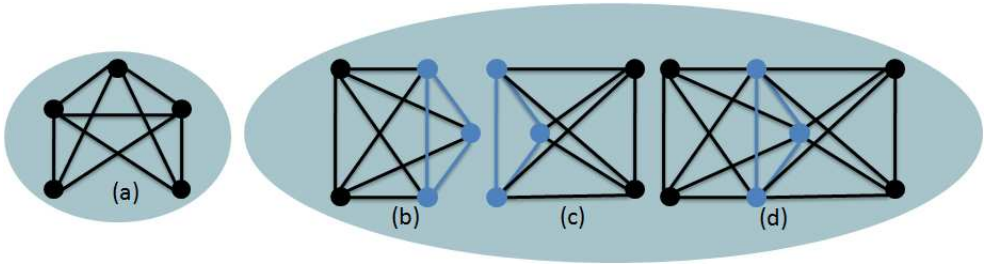

Fig. 7. Ejemplo del grado de restricción de la definición de comunidad basada en k-clique. Para k = 5, los subgrafos densos (a) y (d) no se consideran comunidades.

En la figura 8 se muestra una pequeña red y las comunidades k-clique que esta contiene. En esta imagen se pueden observar características relevantes de los cubrimientos detectados por los algoritmos basados en el desplazamiento de cliques. En primer lugar, no todos los vértices son asignados a una comunidad. Cualquier vértice que no participe en ningún k-clique no puede ser alcanzado por el desplazamiento de un k-clique por lo que no pertenecerá a ninguna comunidad. En la figura estos vértices se resaltan en rojo. Una manera de descartar a priori algunos de estos vértices, sin necesidad de calcular los k-cliques, es mediante el uso de la distribución de los grados de los vértices. Aquellos vértices cuyo grado sea menor que *k* nunca pueden ser alcanzado por el desplazamiento de un k-clique. Este hecho puede utilizarse en una fase de preprocesamiento de la red para descartar los vértices que nunca formarán parte de ninguna comunidad y de esta manera disminuir el tamaño de la red. Otra de las características del algoritmo es su capacidad de detectar comunidades traslapadas, ya que un vértice puede pertenecer a distintos k-cliques y estos no tienen que estar necesariamente conectados. En la figura se muestra el traslape resaltado en naranja. Por otro lado, una desventaja de este tipo de definición, es que las comunidades k-clique no tienen en cuenta la noción de aislamiento con el resto de la red y solo se centran en maximizar su densidad interna. En la

figura se puede ver, como las comunidades azul y amarilla están estrechamente relacionadas y se separan porque de esa manera se obtienen comunidades más densas.

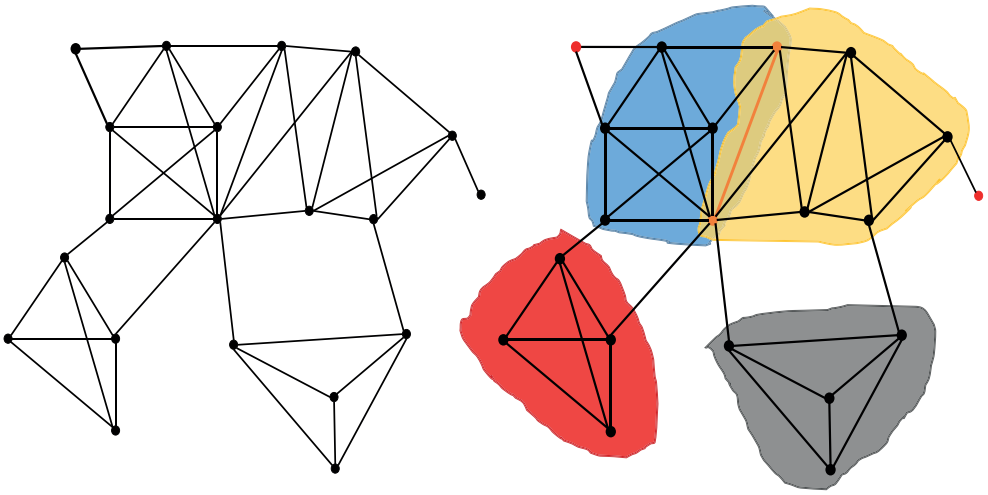

Fig. 8. Ejemplo de una pequeña red y sus comunidades k-clique, para k = 4.

3.2.2. Familia de algoritmos CPM

La primera técnica de detección de comunidades basada en el desplazamiento de cliques fue el algoritmo CPM [51] (Clique Percolation Method). Se han propuesto distintos algoritmos que buscan extender al CPM con el objetivo de mejorar su rendimiento [52] o adaptarlo a tipos de redes más complejas, tales como redes ponderadas [53] y redes dirigidas [54]. De igual modo, CPM también se ha adaptado para procesar redes dinámicas y monitorear la evolución de sus comunidades [2,55]. En cada uno de estos casos las comunidades se definen usando el concepto de comunidad k-clique analizado en la sección anterior. En esta sección se explica la manera en que este concepto de comunidad se adapta a otros tipos de redes y la metodología usada para obtener las comunidades.

Para determinar las comunidades k-clique, el algoritmo CPM inicialmente obtiene todos los N_k cliques maximales de la red. Una vez detectados los cliques maximales, se crea una matriz donde se registra el traslape existente entre estos. Dicha matriz de traslape es de N_k filas y N_k columnas, de manera que en la posición i, j se registra la cantidad de vértices en común que tienen el i-ésimo y el j-ésimo clique. Para seleccionar solamente los k-cliques, basta con cambiar a cero aquellos valores de la matriz asociados a los cliques con tamaño menor que k. Mientras que para la selección de los k-cliques adyacentes basta con cambiar a cero aquellos valores fuera de la diagonal que sean menores que $k-1$. La matriz resultante de las modificaciones anteriores se puede interpretar como la matriz de adyacencia entre los k-cliques de la red. En vista de esto, las comunidades k-clique pueden detectarse, obteniéndo las componentes conexas de esta matriz.

El algoritmo CPM depende notablemente del parámetro k ya que a partir de este valor se definen las comunidades detectadas. No obstante, como la matriz de traslape entre cliques resume la información asociada a todos los cliques maximales de la red, y esta solamente se calcula una vez al inicio del procedimiento, el ajuste correcto de los valores de k puede hacerse eficientemente. Además, estudios empíricos realizados en [51,11,56] han mostrado que valores de k entre 3 y 6 arrojan frecuentemente los mejores resultados.

La complejidad computacional de CPM queda determinada por el cálculo de los cliques maximales de la red. Aunque el costo de esta tarea crece de manera exponencial con respecto al tamaño de la red, los resultados experimentales mostrados en [51] confirman que para redes poco densas esta tarea se puede realizar eficientemente debido a la ausencia de cliques en la red. Expresar cuantitativamente y de forma exacta la complejidad computacional de CPM resulta muy complicado ya que este depende de numerosos factores.

Una de las mayores bondades de las redes ponderadas es que permiten modelar la intensidad de las interacciones entre las entidades de la red. Esto permite eliminar el ruido o perturbaciones en los datos y procesar solo la información verdaderamente significativa. En vista de esto, se han propuesto distintos enfoques para que durante la detección de comunidades se involucren solo las aristas que tengan significación en la red. Una manera intuitiva de aplicar este tipo de filtro, es eliminando de la red aquellas aristas cuyo peso sea menor que un umbral especificado por el usuario, y luego procesar la red resultante como una red no ponderada. Lamentablemente, de esta manera no se utiliza toda la información descrita en los pesos de las aristas. Por otro lado, se pueden dejar de detectar muchas comunidades en la red, ya que tales módulos pueden contener múltiples aristas de peso pequeño.

En [53] se propone el algoritmo CPMw como una extensión del algoritmo CPM para procesar redes ponderadas. En CPMw se introduce el concepto de *intensidad* de un k-clique para llevar a cabo el filtro de información significativa de la red. La intensidad o peso de un k-clique se define como la media geométrica de los pesos de las aristas internas del clique. Las comunidades k-cliques obtenidas por CPMw se forman de manera similar a la propuesta en CPM, con la única diferencia de que solo se utilizan k-cliques cuya intensidad sea mayor que un umbral especificado por el usuario. Este umbral es la mayor dificultad que introduce CPMw con respecto a CPM, ya que el algoritmo es sensible al valor de este parámetro.

Para tener en cuenta la dirección de las aristas en redes dirigidas se propone el algoritmo CPMd [54], donde se sustituye la noción de k-clique por el concepto de *k-clique dirigido*. Un k-clique dirigido es un subgrafo completo de k vértices donde se puede establecer un orden entre los vértices, de manera que para todo par de vértices existe una arista desde el vértice de mayor orden hacia el vértice de menor orden. El ordenamiento entre los vértices se define a partir de sus *grados externos restringidos*. El grado externo restringido de un vértice v se define como la razón entre el número de aristas originadas desde v que terminan en otro vértice del subgrafo y el grado externo total de v. Por otro lado, la adyacencia entre k-cliques dirigidos y las comunidades detectadas por este algoritmo, se definen de manera similar al algoritmo CPM. La complejidad computacional de los algoritmos CPMw y CPMd es similar a la de CPM, ya que estos métodos solo se diferencian entre sí, por la manera en que se definen los k-cliques, y el costo de los procedimientos para obtener los k-cliques maximales son similares.

El algoritmo SCP [52] (Sequential Clique Percolation) constituye una variación del algoritmo CPM, desarrollada para las comunidades k-clique de la red dado un valor específico de k. En SCP no se obtienen todos los cliques maximales de la red ya que no se tiene que registrar la información asociada a todos los posibles valores de k. En este algoritmo, se realiza el análisis sobre una red auxiliar inicialmente vacía, la cual crece de manera secuencial por la inserción de las aristas de la red original. El funcionamiento de SCP consiste en dos fases principales que se llevan a cabo simultáneamente. Por cada nueva arista $\{u,v\}$ que se inserta, en la primera fase se detectan los nuevos k-cliques que se forman producto de dicha adicion; para esto se buscan los (k-2)-cliques existentes entre los vecinos comunes de u y v en la red auxiliar. En la segunda fase, se maneja un grafo bipartito que inicialmente está vacío y en el cual iterativamente se van insertando los cliques que se detectan en la primera fase. Los vértices de dicho grafo se dividen en dos conjuntos asociados al tamaño de los cliques que representan. En un conjunto se tiene a los cliques de

tamaño $k-1$ y en el otro a los cliques de tamaño k. Por cada clique de tamaño k contenido en este grafo, existe una arista del mismo a cada uno de a los k cliques de tamaño $k-1$ que este contiene. Al procesar todas las aristas de la red original, se construyen las comunidades k-clique a partir de las componentes conexas del grafo bipartito. SCP escala linealmente con respecto al número de k-cliques. Aunque en muchas redes reales el número de k-cliques puede ser muy grande, los resultados experimentales han mostrado que SCP se comporta mucho más eficiente que CPM.

Aunque en SCP el ajuste del parámetro k es mucho más complejo que en el algoritmo CPM, este algoritmo tiene la ventaja de que para redes ponderadas puede obtener en una sola ejecución, la estructura en comunidades para todos los umbrales de significación posible. Para lograr esto es necesario insertar las aristas en un orden descendente con respecto a su peso y registrar el cubrimiento detectado por cada nueva arista procesada.

El algoritmo CPM también ha servido como base para el diseño de otros métodos que se encargan de detectar comunidades con traslape en redes dinámicas. En [2] fue llevado a cabo un estudio pionero en el análisis de este tipo de redes para monitorear la evolución de las comunidades en el tiempo. Esta propuesta sigue un enfoque de dos fases, similar al explicado en la sección 2.4. Primeramente, se detectan la partición en comunidades de cada instantánea y posteriormente, se detectan las relaciones existentes entre dichas particiones. Para realizar la asociación entre las comunidades de las instantáneas consecutivas G_t y G_{t+1}, se construye una red auxiliar $G_\cup$ a partir de la unión de G_t y G_{t+1}. Dado que para obtener $G_\cup$ solo se deben realizar inserciones de vértices y aristas sobre cualquiera de las instantáneas, cada una de las comunidades de las instantáneas tiene que aparecer en $G_\cup$, ya sea sin cambios o transformadas a partir de crecimientos o de mezclas con otras comunidades, es decir cada una de las comunidades de las instantáneas aparecerán en exactamente una comunidad en $G_\cup$. Sea $C_\cup$ una comunidad de $G_\cup$. A cada comunidad C_{t+1} perteneciente a G_{t+1} que esté contenida en $C_\cup$ se le asocia en G_t aquella comunidad C_t que esté contenida en $C_\cup$ y que tenga el mayor traslape relativo respecto a C_{t+1}.

La metodología propuesta en este trabajo es muy costosa computacionalmente debido a dos aspectos fundamentales. En primer lugar, no se utilizan los resultados obtenidos en las instantáneas analizadas anteriormente para detectar las comunidades de las instantáneas restantes durante la primera fase. El segundo aspecto está dado por la manera en que se establece la asociación entre las comunidades de las diferentes instantáneas. Para esto, se aplica el algoritmo CPM sobre la unión de cada par de instantáneas consecutivas, lo que consume un tiempo aproximadamente igual que el de la primera fase.

En [55] se propone otro algoritmo basado en desplazamiento de cliques, capaz de procesar redes dinámicas Este algoritmo se llama *Incremental K-clique Clustering* y en lo adelante se denotará como IncrCPM. En IncrCPM, se modela el funcionamiento del algoritmo CPM mediante un grafo de cliques. Los vértices de dicho grafo son los cliques maximales de la red de tamaño mayor que k y existe una arista para cada par de cliques que tienen al menos $k-1$ vértices en común. En este contexto, las comunidades k-clique se definen como las componentes conexas del grafo de cliques.

En IncrCPM se representa la red mediante un modelo incremental de cambio (ver sección 2.4) donde se aceptan como transformaciones válidas la inserción y la eliminación de aristas. Sea C un conjunto donde se registran los cliques maximales de la red que se desea procesar y H un grafo de cliques donde se registra la adyacencia entre los $k-cliques$ de C. Las transformaciones del modelo incremental de cambios se procesan secuencialmente. Por cada nueva transformación procesada se actualiza el conjunto C y el grafo H se modifica en función de los cambios ocurridos en C. En cada momento, los vértices de H son los cliques de C cuyo tamaño sea mayor que k y que solamente existan aristas entre cliques que compartan al menos $k-1$ vértices en común. Cuando se procesa la inserción de una arista se pueden formar nuevos

cliques, los cuales se obtienen de manera similar a como se hace en el algoritmo SCP. Estos cliques se agregan en C después de eliminar de dicho conjunto, aquellos cliques que estén contenidos en alguno de los cliques que serán agregados. Por otro lado, cuando se procesa la eliminación de una arista $\{u,v\}$ se procesan solamente los cliques de H que contienen dicha arista. Cada clique $C_i \in C$ donde aparecen u y v se divide en dos obteniéndose los cliques $C_i - \{u\}$ y $C_i - \{v\}$. Luego, se elimina de C el clique C_i y en su lugar se agregan los dos cliques obtenidos en la división. En el caso de que ningún clique de C contenga la arista $\{u,v\}$ entonces esta transformación no afecta la estructura en comunidades de la red y C se mantiene intacto.

La complejidad computacional de este algoritmo depende tanto de la configuración inicial de la red como de la secuencia de cambios. La mayoría de las operaciones que realiza son sumamente eficientes ya que solo involucran la actualización de los cliques que participan en la transformación. Los experimentos descritos en [55] mostraron que este algoritmo es más eficiente que el CPM estático.

De manera general, todos los métodos descritos en esta sección son muy sensibles a la densidad de la red. La técnica de desplazamiento de cliques asume que en la red existe un gran número de cliques y por tanto, cuando la red es poco densa puede dejar de detectar comunidades relevantes. Por otro lado, si la red tiene componentes muy densas y un gran número de cliques, se pueden detectar comunidades triviales de gran tamaño o en el peor de los casos devolver la red completa como una comunidad. Además, si la densidad de la red es heterogénea, o sea tiene partes densas y otras partes dispersas, la selección de un valor adecuado para el parámetro k puede resultar muy compleja.

3.3. Algoritmos basados en la optimización de medidas de calidad

Generalmente, las medidas de calidad cuantifican las propiedades deseadas en las comunidades o particiones que se identifican. En vista de esto, su uso no ha sido restringido a la comparación de particiones o en la validación de los resultados obtenidos. La optimización de este tipo de medidas ha sido la base de muchos algoritmos propuestos para la detección de comunidades.

En la subsección 3.3.1 se abordan los principales algoritmos de OCD que siguen un enfoque de optimización global de medidas de calidad. En particular se profundiza en el análisis de los algoritmos basados en la medida modularidad descrita en la sección 2.2.2. La idea central de este tipo de algoritmos consiste en realizar cambios estructurales que involucren distintas comunidades para maximizar la calidad de las particiones. Para esto, llevan a cabo un análisis de la red desde una perspectiva global y requieren que esté disponible la información estructural de la red completa. Dado que en redes de gran tamaño es muy difícil contar con esta información, se han propuesto algoritmos que siguen un enfoque local para la evaluación de calidad. Los algoritmos de optimización local se centran en incrementar la calidad de cada comunidad de manera independiente sin tener en cuenta las relaciones entre los distintos grupos. En la subsección 3.3.2 se describen los algoritmos de OCD que siguen una estrategia de expansión local para optimizar las medidas de calidad de las comunidades.

3.3.1. Algoritmos de expansión global basados en modularidad

Como se ha explicado en secciones anteriores, una de las mayores dificultades en el problema de detección de comunidades es que no existe un consenso de qué es una *buena* partición. Entre las medidas de calidad de partición más utilizadas se encuentra la función de modularidad (ver sección 2.2.2). En esta sección, se explica brevemente la idea de los algoritmos basados en optimización de modularidad, se analizan los principales métodos de OCD que basan su funcionamiento en esta medida y se describen las generalizaciones del concepto de modularidad que permiten el traslape entre comunidades.

Los algoritmos que se basan en la optimización de modularidad asumen que mayores valores de modularidad indican particiones de mejor calidad y por ello, seleccionan como mejor partición aquella asociada al valor más alto de modularidad. Llevar a cabo una búsqueda exhaustiva de la partición con valor óptimo de modularidad, resulta impracticable dada la complejidad del problema y el enorme tamaño que caracteriza a las redes reales. De hecho, ha sido demostrado que este problema pertenece al conjunto de problemas NP [57]. Por tal motivo, se han propuesto distintas heurísticas que permiten detectar particiones que se aproximan bastante a la óptima, consumiendo un tiempo mucho menor. Tal el es caso de las estrategias *greedy*, o de las técnicas de optimización basadas en *recocido simulado* y algoritmos genéticos, entre otras. Este tipo de enfoque ha sido ampliamente abordado en el contexto de DCD. La mayoría de los algoritmos de DCD basados en optimización de modularidad se describen minuciosamente en [3].

El algoritmo de DCD llamado Louvain [58] 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 un grupo unitario. 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. La partición final detectada en esta primera fase constituye el primer nivel del dendograma generado por Louvain. Posteriormente, en la segunda fase se crea una red auxiliar donde cada nodo representa una comunidad de la partición obtenida en la primera fase. Si existen aristas puentes entre dos comunidades C_1 y C_2, entonces en la red auxiliar existe una arista entre los nodos asociados a dichos grupos. El peso de dicha arista, es la sumatoria de los pesos de todas las aristas puentes entre C_1 y C_2. Luego, el siguiente nivel del dendograma se obtiene aplicando la primera fase sobre la red auxiliar. Repitiendo este procedimiento se obtienen los distintos niveles jerárquicos. La complejidad computacional de Louvain es de $O(m)$, siendo m el número de aristas de la red, por lo que permite procesar redes de gran tamaño.

La actualización de las comunidades de los vértices durante la primera fase, depende del orden en que se recorren los vértices. Por tal motivo Louvain puede comportarse de un modo no determinista, lo que es un aspecto negativo para la experimentación y aplicación de un algoritmo. No obstante, dada la gran eficiencia computacional de Louvain, este ha sido utilizado como componente de numerosas técnicas de detección de comunidades más complejas. Tal es el caso del algoritmo FD (Fuzzy Detection) [59] el cual aprovecha el comportamiento no determinista de Louvain para identificar los nodos que pertenecen a varias comunidades.

Al aplicar varias veces el algoritmo Louvain sobre una misma red, se pueden obtener distintas particiones. El funcionamiento de FD se basa en registrar por cada par de vértices de la red, la probabilidad de co-aparición. La probabilidad de co-aparición entre dos nodos es la probabilidad de que ambos pertenezcan a la misma comunidad. Para esto, se lleva a cabo una fase iterativa, donde en cada momento se ejecuta el algoritmo Louvain sobre la red y se actualiza la probabilidad de co-aparición entre cada par de vértices. Además, también se registra la partición P de mayor valor de modularidad entre todas las obtenidas. La condición de parada que se utiliza para esta fase, es que la diferencia entre las probabilidades de co-aparición obtenidas en iteraciones sucesivas sea menor que un umbral ε especificado por el usuario.

Una vez culminada la primera fase, el cubrimiento final determinado por FD se crea a partir de P. En este algoritmo, las comunidades se definen por la unión de módulos estructurales denominados *grupos robustos*. Dichos módulos no son más que grupos cuyos vértices presentan altos valores de probabilidad de co-aparición. Para la detección de los grupos robustos de la red, se eliminan de P todas las aristas que conecten nodos con probabilidad de co-aparición menor que un parámetro α. Como resultado de este filtro, cada comunidad de P puede quedar dividida en varios grupos robustos. Se selecciona como núcleo representativo de cada comunidad a su grupo robusto de mayor tamaño.

Siguiendo un enfoque similar al usado en Louvain, en FD se crea una red auxiliar donde cada nodo representa un grupo robusto de los detectados previamente. Las aristas de esta red se definen a partir de las relaciones existentes entre los grupos representativos, tal como se hace en Louvain. A esta red auxiliar se le aplica la primera fase de FD para obtener la probabilidad de co-aparición entre cada par de grupos robustos. De esta manera se puede obtener el grado de pertenencia de cada grupo robusto con respecto a las comunidades. Luego, por cada núcleo c_i representativo de una comunidad de P, se obtiene una comunidad, mezclando a c_i con todos los grupos robustos con que tenga una probabilidad de co-aparición mayor que un parámetro β. De esta manera el traslape entre las comunidades queda establecido por los grupos robustos que pueden pertenecer a distintas comunidades. Esto puede incidir notablemente en la realización de otras tareas relacionadas tales como establecer las relaciones jerárquicas entre las comunidades o interpretar el papel del traslape en la evolución de los grupos en las redes dinámicas.

La complejidad computacional de FD depende notablemente del cálculo de las probabilidades de co-aparición. Sobre todo en cuanto al consumo de memoria ya que se deben registrar dicha probabilidad entre cada par de nodos de la red. En cuanto a eficiencia, el costo temporal de FD es de $O(Km)$, siendo K el número de veces que se ejecuta el algoritmo Louvain durante la primera fase.

La presencia de los parámetros $\varepsilon, \alpha, \beta$ que deben ser ajustados por el usuario constituye una gran dificultad para la aplicación práctica de FD. Esto se debe a que el comportamiento del algoritmo puede ser muy sensible con respecto a los valores de dichos parámetros. El número de particiones que se utilizan para obtener la probabilidad de co-aparición de cada par de vértices y que se regula mediante el parámetro ε, debe ser el mínimo necesario para obtener un buen balance entre eficiencia y estabilidad de los resultados. Además, los filtros realizados con los umbrales α, β son determinantes a la hora de formar las comunidades. Por ejemplo, en caso de que una comunidad esté formada por distintos grupos robustos de tamaños similares, se debe ajustar el umbral α para una mayor seguridad en la identificación del núcleo de la comunidad.

Otra métodología de expansión global es propuesta en el algoritmo EAGLE [60] (Agglomerative Hierarchical Clustering based on Maximal Clique). En este método se sigue un enfoque jerárquico aglomerativo con el fin de detectar las relaciones jerárquicas existentes entre las comunidades de la red. Al igual que en varios algoritmos descritos anteriormente, en EAGLE se utilizan los cliques maximales como núcleos estructurales de las comunidades. Para evitar la formación de grupos redundantes solamente se consideran como comunidades iniciales aquellos cliques maximales de tamaño mayor o igual que k, siendo k un parámetro especificado por el usuario. Aquellos vértices que no pertenezcan a ninguno de los cliques seleccionados por el criterio anterior, se tratan como grupos unitarios y se agregan a la partición inicial. Esta partición constituye el nivel inferior del dendograma generado por EAGLE.

Luego, basándose en el enfoque aglomerativo, se lleva a cabo un proceso iterativo de manera que la partición obtenida en sucesivas iteraciones constituyen niveles superiores del dendograma. En cada paso iterativo se seleccionan las dos comunidades C_1 y C_2, de la partición anterior, que presenten la mayor similitud estructural. La nueva partición se forma a partir de las comunidades de la partición anterior, pero

sustituyendo a C_1 y C_2 por la comunidad resultante de su mezcla. Este proceso finaliza cuando la partición actual está formada por una única comunidad.

La manera en que se determina la similitud entre dos comunidades C_1 y C_2 brinda una descripción implícita del valor de modularidad asociado a la unión de C_1 y C_2. En función de esto, se define a la función S de la siguiente forma:

$S = \frac{1}{2m}\Sigma_{v\in C_1, w\in C_2, v\neq w}\left[A_{vw} - \frac{d_v d_w}{2m}\right]$, donde:
A_{vw} es el valor asociado a los vértices v y w, en la matriz de adyacencia.

La principal limitación de EAGLE está en su elevado costo computacional. En primer lugar, determinar los cliques maximales de una red se conoce que es un problema de complejidad exponencial. No obstante, en la práctica se ha comprobado que esta tarea puede ser llevada a cabo en muchas redes reales debido a la baja densidad que caracteriza a dichas redes. Sin tener en cuenta este costo inicial, EAGLE sigue siendo ineficiente ya que presenta una complejidad computacional de $O(n^2+h)s)$, donde s es la cantidad cliques maximales detectados inicialmente en la red y h es la cantidad de pares de cliques maximales que se relacionan entre sí, mediante aristas puentes o a través de vértices en común. Además, el parámetro k necesita ser ajustado dependiendo de las características estructurales de la red, lo que dificulta la aplicación de EAGLE por parte del usuario.

De manera general, los algoritmos que siguen una estrategia de optimización global presentan una serie de dificultades que han provocado un decremento en el número de trabajos que siguen este enfoque. En primer lugar, de acuerdo a las medidas de evaluación de partición existentes, las particiones con altos valores de calidad no tienen que estar necesariamente formadas por comunidades de calidad. Por otro lado, los resultados obtenidos en estos algoritmos se encuentran en función de toda la estructura de la red, por lo que si el análisis se centra en algunas zonas de interés se obtendrían particiones diferentes. Además, la aparición de los límites de resolución inherente a la optimización global, obliga a buscar alternativas de solución tales como métodos multiresolución los cuales pueden llegar a ser sumamente costosos. En muchos de los algoritmos multiresolución se requiere de conocimiento previo sobre la cantidad de comunidades que se desean detectar. En vista de esto, recientemente se han dedicado muchos más esfuerzos orientados a enfoques locales de optimización.

Generalizaciones de modularidad para procesar comunidades con traslape

Como se menciona en la sección 2.2.2, existen distintas generalizaciones de modularidad que permiten la aplicación de esta medida en contextos más complejos. En particular, para la detección de comunidades con traslape no puede usarse la expresión cuantitativa propuesta en [22]. En vista de esto, se han propuesto distintas extensiones de la definición de modularidad, para permitir el traslape entre las comunidades.

En [60] se presenta una de estas extensiones de modularidad, para medir la calidad de las particiones del dendograma generado por el algoritmo EAGLE, descrito anteriormente. La modificación propuesta consiste en tener en cuenta la cantidad de comunidades de la partición a las que pertenece cada vértice, y se define de acuerdo a la siguiente expresión:

$Q(P) = \frac{1}{2m}\Sigma_{ij}\frac{1}{O_i O_j}\left[A_{ij} - \frac{d_i d_j}{2m}\right]\sigma(C_i, C_j)$, donde:
O_i denota la cantidad de comunidades donde pertenece el vértice i.

Otra variante de modularidad que permite el traslape entre las comunidades, se propuso en [61]. En esta variante, se tiene en cuenta las nociones de aislamiento y la densidad de las comunidades. En esta variante, la modularidad de la partición se obtiene promediando el aporte de cada comunidad. Dicho aporte se determina mediante la expresión:

$$Q_C = \left[\frac{1}{V_C}\sum_{i\in C}\frac{\sum_{j\in C}A_{ij}-\sum_{j\notin C}A_{ij}}{d_i\cdot q_i}\right]\frac{E_C}{\binom{V_C}{2}}, \text{ donde:}$$

q_i es la cantidad de comunidades asociadas al vértice i, mientras que V_C y E_C son la cantidad de vértices y aristas de C respectivamente.

Existen distintas propuestas [62,63,64,65] que generalizan la modularidad para evaluar agrupamientos en comunidades difusas. En estas variantes se incorporan al análisis los grados de pertenencia de cada vértice a sus respectivas comunidades. Las variantes presentadas en [62,63] se abordarán con mayor detalle en la sección 3.7.

En la expresión original de modularidad, el factor $\sigma(C_i,C_j)$ indica la pertenencia mutua de los vértices i y j a la misma comunidad. En el caso de agrupamiento difuso, la probabilidad de que el vértice i pertenezca a una comunidad C se representa a partir de su grado de pertenencia α_{iC}, mientras que la probabilidad de que ambos vértices i y j pertenezcan a C se determina mediante el producto de ambos grados de pertenencia. En vista de esto, este tipo de modularidad se determina mediante la expresión:

$$Q(P) = \frac{1}{2m}\sum_{C\in P}\sum_{ij}\left[A_{ij}-\frac{d_i d_j}{2m}\right]\alpha_{iC}\alpha_{jC}.$$

La diferencia principal entre este tipo de medidas, está dada por la manera en que se obtienen los grados de pertenencia de los vértices a sus respectivas comunidades.

En [64], para obtener el grado de pertenencia $\alpha_{i,C}$ de un vértice i a una comunidad C, se tienen en cuenta tanto las conexiones internas de i en C como las conexiones internas de i en el resto de las comunidades donde pertenece. Este cálculo se determina mediante la expresión:

$$\alpha_{iC} = \frac{d_i^{int}(C)}{\sum_{C'\in P} d_i^{int}(C')}.$$

Otra extensión difusa de modularidad se propone en [65]. En esta variante se analizan los cliques maximales presentes en la red. Dado el alto grado de conectividad presente entre los vértices de un clique, los autores asumen que si dos comunidades comparten en común un clique maximal, entonces deberían representar una misma comunidad. En vista de eso, asumen que cada clique maximal es parte de una única comunidad. En esta propuesta se calcula el grado de pertenencia α_{iC} mediante la expresión:

$$\alpha_{iC} = \sum_{j\in V}\frac{|O_{ij}^C|}{|O_{ij}|}A_{ij}, \text{ donde:}$$

O_{ij} representa el conjunto de cliques maximales que contienen a los vértices i y j, mientras que O_{ij}^C son los cliques de O_{ij} que están contenidos en la comunidad C.

3.3.2. Algoritmos de expansión local

En redes complejas de gran tamaño se incrementa la naturaleza local de las comunidades, ya que los vértices interactúan solamente con una pequeña parte de la red [66]. Por ejemplo, las redes sociales se caracterizan por la presencia de pequeñas comunidades bien definidas. A medida que estos pequeños

grupos incrementan su tamaño, tienden a perder su estructura de comunidad y a mezclarse con el resto de la red [67]. En vista de esto, se han propuesto distintos algoritmos de OCD donde las comunidades se definen como subgrafos que optimizan localmente una función de calidad de comunidades. En este contexto, un subgrafo G es óptimo local de una función de calidad f si no se puede obtener un mayor valor de f a partir de otro subgrafo resultante de la inserción o eliminación de un solo vértice en G.

De forma general este tipo de algoritmo primeramente identifica un conjunto de *comunidades iniciales* que sirven de *núcleo* estructural de las comunidades que se desean obtener. Posteriormente, cada comunidad inicial es procesada con el objetivo de maximizar una función de calidad, a través de una expansión local de dicha comunidad.

Una de las propiedades más comunes de las redes complejas reales es la llamada *Ley de transitividad*, la cual plantea que si en una red compleja existen las aristas $\{u,v\}$ y $\{v,w\}$ es muy probable que también exista la arista $\{u,w\}$. En base a esta propiedad, la mayoría de los algoritmos que se expanden localmente, lo hacen a partir de los nodos de la *vecindad* de la comunidad, la cual es el conjunto de vértices que no pertenecen a la comunidad pero que están conectados a algún nodo de la misma.

La estrategia más simple para seleccionar las comunidades iniciales consiste en tratar cada vértice como una comunidad unitaria. Sin embargo, esta estrategia tiene un costo computacional muy elevado y siguiendo este enfoque se pueden generar muchas comunidades duplicadas. En vista de eso, se han propuesto distintas estrategias para optimizar la selección de las comunidades iniciales de manera que se pueda cubrir la mayor parte de la red, expandiendo el menor número posible de comunidades iniciales.

En aras de alcanzar una mayor eficiencia computacional, la mayoría de las funciones de calidad utilizadas, se definen a partir de criterios cuantitativos simples. Estos criterios suelen ser fáciles de satisfacer, lo que provoca que el número de subgrafos de la red que se identifican como comunidades puede ser muy alto y que exista un gran nivel de traslape entre estos.

Con base en lo descrito anteriormente, se puede concluir que las diferencias entre los algoritmos que siguen este enfoque, se establecen respecto al método de selección de las comunidades iniciales o grupos semillas, la función de calidad utilizada para definir las comunidades y la heurística utilizada para llevar a cabo la expansión local.

Los primeros algoritmos en seguir una estrategia de expansión local fueron *RARE-IS* [68] (Rank Removal - Iterative Scan) y *LA-IS*^{2} [69] (Link Aggregate - Iterative Scan). Para la selección de los grupos semillas en *RARE-IS* se utiliza el método RARE, el cual consiste en ir eliminando de la red los vértices de mayor centralidad, hasta que la red se divida en componentes conexas que cumplan con el umbral de tamaño especificado por el usuario. Posteriormente, los vértices eliminados se agregan a las componentes conexas, solo si ese cambio incrementa la calidad de la componente donde se agrega. Los subgrafos resultantes de este proceso se utilizan como comunidades iniciales. Por otro lado, *LA-IS*^{2} realiza la formación de los grupos semillas de una manera diferente. Primeramente, se ordenan los vértices de la red en cuanto a su PageRank [70]. Luego, cada uno de los vértices se asigna en ese orden a los grupos cuya cualidad se incremente producto de dicha adición. En caso de no existir tales grupos, se crea uno nuevo formado por el vértice a asignar. En ambos trabajos se propone el uso de distintas medidas de calidad asociadas a la noción de densidad para definir las comunidades. Las heurísticas usadas en ambos para la expansión de las comunidades iniciales, siguen una estrategia *greedy*, de manera que en cada paso se realiza el cambio que más aporte a la calidad de la comunidad. Los cambios permitidos son eliminar un vértice perteneciente a la comunidad o agregar un nuevo vértice a la misma. En este sentido, los dos algoritmos se diferencian en que *LA-IS*^{2} solo agrega vértices que pertenezcan a la vecindad de la comunidad, mientras que en *RARE-IS* se pueden agregar vértices de cualquier parte del resto de la red. En ninguno de los dos algoritmos se realiza una fase de post-procesamiento y su mayor dificultad consiste en que en ambos se genera una gran

cantidad de comunidades, con un alto nivel de traslape entre estas. La complejidad computacional de estos métodos es $O(n^2)$.

Una ligera modificación de *LA-IS*^{2} se propone con el algoritmo *LA-CIS* [71] (Link Aggregate - Connected Iterative Scan). Una de las principales diferencias con *LA-IS*^{2} es que en los requerimientos de la función de calidad utilizada en *LA-CIS* se incorpora la noción de conectividad. En la creación de los grupos semillas y durante la expansión de los mismos, *LA-CIS* comprueba que la comunidad sea conexa. La otra diferencia con respecto a *LA-IS*^{2}, es que se realiza una fase de post-procesamiento donde se refinan los resultados obtenidos. En esta fase, se realiza un análisis de las comunidades detectadas para mezclar las que tengan mucho traslape o eliminar las que puedan obtenerse a partir de otras. Por otro lado, para reducir el número de comunidades se propone realizar filtrados en dependencia del contexto donde se aplica el algoritmo, mediante el uso de otros criterios conocidos para definir comunidades (ver sección 2.1). No obstante, los problemas de *LA-IS*^{2} no se solucionan completamente y *LA-CIS* puede tener un desempeño pobre en redes con un bajo nivel de traslape entre sus comunidades. El algoritmo *LA-CIS* no realiza ningún tipo de mejoras en cuanto a eficiencia computacional con respecto a *LA-IS*^{2} y por lo tanto, su complejidad temporal es la misma, $O(n^2)$.

En [66], se propone el algoritmo multiresolución LFK el cual fue el primero en ser capaz de detectar tanto el traslape como las relaciones jerárquicas existentes entre las comunidades. LFK interpreta los vértices de la red como grupos unitarios de semillas, por lo que cada uno de ellos forma una comunidad. Por otro lado, la heurística utilizada para llevar a cabo la expansión local de la función de calidad, es similar a las utilizadas por los algoritmos descritos anteriormente. Siguiendo un enfoque *greedy* se selecciona el vértice de la vecindad de la comunidad que más incremente la calidad del grupo con su incorporación al mismo. La diferencia principal en este sentido es que después de cada inserción de un nodo nuevo, se remueven de la comunidad todos los elementos cuya eliminación incremente la calidad del nuevo grupo.

La función de calidad utilizada por LFK se define como la razón entre el grado interno y el grado total de la comunidad, este último aparece influenciado por un parámetro de resolución α que permite controlar el tamaño de las comunidades que se detectan. Esta función de calidad puede calcularse de la siguiente manera:

$$f(C,\alpha) = \frac{d^{int}(C)}{(d^{int}(C) + d^{ext}(C))^{\alpha}}.$$

Esta función de calidad puede ser extendida para procesar redes ponderadas y redes dirigidas. Para evaluar comunidades en redes ponderadas, solo es necesario remplazar el grado de los vértices por la sumatoria de los pesos de las aristas. En cuanto a la evaluación en redes dirigidas, el grado interno de un vértice se remplaza por el número de aristas que se inician en otro vértice de la comunidad y que terminan en él, mientras que el grado externo se sustituye por la cantidad de aristas que inciden en él y que provienen del resto de la red.

La variación del parámetro de resolución permite la exploración de la red en todas las escalas o resoluciones posibles. A partir de las particiones obtenidas con resoluciones diferentes, se puede detectar las relaciones jerárquicas existentes entre las comunidades.

Entre las principales limitaciones de LFK está la dependencia del algoritmo respecto al parámetro de resolución, ya que ajustar correctamente este parámetro requiere de un conocimiento previo de las características de la red por parte del usuario. Además, las comunidades detectadas por LFK pueden tener alto nivel de traslape y el algoritmo no cuenta con ninguna fase de post-procesamiento donde se validen los resultados obtenidos. Por otro lado, la complejidad computacional de LFK es $O(n^2 \log n)$ lo cual limita su uso en numerosas aplicaciones prácticas. El factor $\log n$ se debe a la cantidad mínima de valores diferentes para el parámetro de resolución α que se necesitan para obtener las comunidades de todos los tamaños en la red.

En LFK se lleva a cabo la formación de una comunidad por cada uno de los nodos de la red, lo que puede ser extremadamente costoso considerando que es necesario experimentar con distintos umbrales de resolución. En vista de esto, en el mismo trabajo se presenta una variante del algoritmo denominada LFM (Local Fitness Maximization) donde solo se modifica la selección de comunidades iniciales. En esta variante también se utilizan los vértices como grupos unitarios de semillas, la diferencia con LFM esta dada en que se reduce la cantidad de comunidades iniciales. Para esto se propone realizar una selección aleatoria de los vértices a expandir, mediante un proceso iterativo donde en cada iteración solo se expande como semilla a un nodo que no haya sido asociado a alguna comunidad en iteraciones anteriores.

En distintos estudios ha sido comprobado que los cliques son estructuras características de las comunidades [51]. Basándose en este hecho, se proponen otros algoritmos multiresolución nombrados GCE [72] (Greedy Clique Expansion) y MONC [73] (Merging of Overlapping Natural Communities) que combinan la detección de cliques con un enfoque de expansión local similar al usado en LFK. En ambos casos se usa la misma función de calidad propuesta en LFK.

En GCE se utiliza como comunidades iniciales a los cliques maximales de la red cuyo tamaño sea mayor que el valor de un parámetro k especificado por el usuario. Una vez detectados los cliques a usar como grupos semillas, estos se expanden a partir de un proceso iterativo donde en cada paso se selecciona el vértice cuya incorporación incremente en mayor medida la calidad de la comunidad. La diferencia principal con respecto a la heurística usada en LFK, es que en GCE no se elimina de la comunidad ningún vértice previamente seleccionado como parte de esta. Una vez finalizado el proceso de expansión, se determina si la comunidad resultante y alguna comunidad detectada anteriormente tienen una similitud mayor que un umbral ε especificado por el usuario. En tal caso, se considera que la comunidad es un duplicado de la anterior y se descarta, de otro modo se registra como comunidad de la red.

La complejidad computacional de GCE es muy difícil de expresar en términos de la cantidad de vértices y la cantidad de aristas, no obstante en los experimentos se comprobó que su costo temporal usualmente es mucho menor que el de LFM. Sin embargo, también se evidenció una dificultad que presenta el algoritmo, y es que GCE es muy sensible al grado promedio de los vértices de la red. Otra dificultad de GCE es que este depende de tres parámetros cuyos valores deben ser ajustados por el usuario, estos son α, ε y k. Aunque los autores proponen durante la experimentación, valores que pueden usarse por defecto para estos parámetros, su efectividad no tiene porque ser igual para todas las redes donde se aplique el algoritmo.

Como se explicó en la sección 2.3.2, los algoritmos multiresolución exploran la red a distintas escalas de resolución para obtener las comunidades de todos los tamaños posibles presentes en la red y detectar las relaciones jerárquicas existentes entre estas. En particular, tanto LFK como GCE ajustan el parámetro de resolución con pequeños cambios y ejecutan la detección de comunidades de la red para cada uno de estos valores de resolución. De esta manera, el algoritmo puede ser ejecutado con umbrales de resolución inadecuados y no aportar nada a la estructura de comunidades que se desea detectar, lo que puede convertirse en una sobrecarga temporal completamente innecesaria. Como propuesta de solución a esta problemática se presenta el algoritmo MONC, el cual es capaz de determinar internamente cuáles son los valores de resolución en que el parámetro necesita ser evaluado, disminuyendo notablemente el costo temporal de la técnica.

Al igual que en GCE, en MONC se seleccionan las comunidades iniciales a partir de cliques maximales de la red, solo que el usuario no necesita especificar ningún umbral de tamaño mínimo. Por cada clique maximal K, se forma un grupo semilla compuesto por el clique resultado de eliminar de K los vértices que menos aporten a la calidad de la comunidad en una escala inferior. La selección de cliques como comunidades iniciales implica que todas las comunidades detectadas por el algoritmo contengan al menos un clique, lo cual puede ser una restricción muy fuerte y pueden dejar de detectarse muchas

comunidades existentes en la red que no cumplan este requisito. En vista de esto, en MONC también pueden seleccionarse como comunidades iniciales a los grupos unitarios de semillas formados por cada uno de los vértices de manera similar a como se hace en LFK. No obstante, en los resultados experimentales mostrados en [73] se evidencia que MONC es más efectivo usando las semillas basadas en cliques. Esto puede ser debido a que durante la expansión de las comunidades, MONC no permite la eliminación de elementos de la comunidad, una vez sean identificados como parte de esta.

La función de calidad usada en MONC es una simple modificación a la definida a partir de la función que se usa en LFK y GCE. El cambio consiste en incrementar una unidad el numerador con el objetivo de permitir que las comunidades puedan estar formadas por un único vértice.

En cuanto a la metodología usada para la expansión local de las comunidades, MONC se basa en el cálculo de los niveles de resolución con los que el algoritmo necesita explorar la red para detectar los cambios ocurridos en las comunidades que se expanden. Esta metodología consiste en incorporar a la comunidad un vértice de su vecindad siguiendo una estrategia greedy al igual que en LFK y GCE, solo que en esta ocasión no se incorpora el vértice que más incremente la función de calidad de la comunidad, sino que se incorpora el vértice que incremente la calidad de la comunidad en el mayor valor de resolución posible.

Una de las principales desventajas de los mecanismos de expansión usados tanto en GCE como en MONC es que los cambios son irreversibles, ya que una vez se asigne un vértice como parte de una comunidad, dicho vértice no va a dejar de pertenecer a la misma aún cuando realmente disminuya su calidad .

Una de las mayores dificultades de MONC es que generan una gran cantidad de comunidades con alto nivel de traslape entre estas. En vista de esto, MONC también realiza una fase de post-procesamiento se lleva a cabo un filtrado sobre las comunidades duplicadas y se mezclan las que estén cercanas a ser duplicadas. El usuario debe especificar un umbral de similitud para definir cuándo dos comunidades son cercanas a ser duplicadas. Dada una comunidad C, MONC interpreta el conjunto de comunidades cercanas a C, denotado como $\eta(C)$, como diferentes puntos de vista de C desde la perspectiva de los grupos semillas que generaron las comunidades de $\eta(C)$. A partir del traslape existente entre las comunidades de $\eta(C)$, se puede establecer una asignación difusa a los vértices de C. El grado de pertenencia en C de un vértice v se calcula como la cantidad de comunidades de $\eta(C)$ que contienen a v.

En la fase de post-procesamiento también se realiza la identificación de los niveles de resolución críticos de la red a partir de los cuales se identificarán las relaciones jerárquicas de la red. Esto se logra a partir de un análisis basado en la cantidad y tamaño de las comunidades detectadas en cada uno de los niveles de resolución calculados en la fase de expansión de todos los grupos semillas.

La complejidad computacional de MONC, no puede ser descrita fácilmente a partir de una expresión cuantitativa. No obstante, los experimentos realizados mostraron que MONC tiene un costo temporal mucho menor que LFK y que GCE. Por otro lado, MONC no requiere ningún parámetro para la detección de las comunidades. Solo necesita el ajuste de algunos parámetros en la fase de post-procesamiento durante la selección de los niveles de resolución adecuados para establecer las relaciones jerárquicas, y para determinar el grado de pertenencia de los vértices a sus respectivas comunidades.

La mayoría de algoritmos que detectan comunidades están diseñados para un contexto específico, por lo que es muy difícil seleccionar un algoritmo que pueda ser utilizado en distintos entornos de aplicación. Para lidiar con esta problemática se propone el algoritmo multifuncional OSLOM [74] (Order Statistics Local Optimization Method). Este algoritmo es capaz de procesar redes teniendo en cuenta la dirección y el peso de las aristas. Además, detecta tanto las relaciones jerárquicas como el traslape existente entre las comunidades. Otra característica muy importante de este algoritmo es que puede ser utilizado como

un procedimiento auxiliar para refinar los resultados obtenidos por otro algoritmo de detección de comunidades.

OSLOM sigue un enfoque de optimización local similar al de los algoritmos descritos anteriormente. La función de calidad que se optimiza durante la fase de expansión describe la significación estadística de cada grupo. Los autores definen la significación estadística de una comunidad en una red, como la probabilidad de que dicha comunidad exista en un MGAN con la misma distribución de grado de los vértices de la red (ver sección 2.2). Intuitivamente, si un vértice v se relaciona con una comunidad C más que lo esperado en un MGAN, entonces la interacción entre v y C se considera significativa por lo que v debe ser parte de los vértices de C.

En vista de esto, el funcionamiento general de OSLOM se describe como sigue. Inicialmente, se conforman los grupos semillas que serán refinados durante la fase de expansión. Estos grupos pueden ser especificados por el usuario, utilizando la salida de cualquier algoritmo de detección de comunidades. En caso de no ser indicados como entrada de OSLOM, cada grupo semilla C_0 se forma a partir de un único vértice seleccionado de manera aleatoria, agregando a C_0 los q vértices de mayor significación entre los miembros de su vecindad. El valor de q puede escogerse arbitrariamente, aunque los autores proponen tomarlo de una distribución que siga alguna ley de potencia. Una vez definidos los grupos semillas, se comienza con la expansión de cada grupo.

El resultado final de expandir un grupo semilla en OSLOM, es una comunidad cuya interacción con los vértices de su vecindad es similar a como se esperaba que se relacionaran en un MGAN H y cuya densidad interna no es compatible con las interacciones entre los elementos de la comunidad en H. Para esto se lleva a cabo el cálculo de distintas variables aleatorias que representan la similitud entre la relación topológica de un vértice v con un grupo C en la red y la relación esperada entre estos en H. Este cálculo implica que el algoritmo tenga un comportamiento no determinista durante la fase de expansión. Para lidiar con esto, cada grupo semilla C se expande L veces, donde L es un parámetro especificado por el usuario. Si en al menos $L/2$ ocasiones el resultado de la expansión es un grupo no vacío, entonces se considera significativo el grupo C_L, formado por los vértices que aparecen en al menos $L/2$ expansiones de C. En tal caso, se considera de C_L es la expansión del grupo C. La heurística utilizada para llevar a cabo la expansión de los grupos se divide en dos fases principales. Inicialmente, se agrega al grupo C cada vértice perteneciente a su vecindad que sea significativo estadísticamente para C. Los autores consideran que un vértice v es significativo para un grupo C con respecto a un MGAN H, si las relaciones existentes entre v y C en la red original y las existentes en H, presentan una similitud menor que un umbral P especificado por el usuario. Evidentemente, algunos vértices pueden dejar de ser significativos para el grupo durante la expansión. En la segunda fase, se eliminan los vértices w del grupo, que no sean significativos para C y que no serían incorporados nuevamente si se expandiera el grupo $C - \{w\}$.

La etapa final del algoritmo, consiste en la obtención de las relaciones jerárquicas existentes entre las comunidades de la red. Estas relaciones jerárquicas se obtienen mediante un proceso iterativo de manera que inicialmente las comunidades son las detectadas en las fases anteriores del algoritmo. En cada paso, se generan las comunidades de un nivel jerárquico superior a partir de las comunidades obtenidas en la iteración anterior. Para esto, se crea una red auxiliar G' donde cada vértice de G' son las comunidades detectadas previamente. Las aristas de G' se forman a partir de las relaciones existentes entre las comunidades. Luego, las comunidades de dicho nivel jerárquico se definen como las comunidades de G'. La cantidad de niveles generados del dendograma se determina por la cantidad de iteraciones que se realizan en este procedimiento. La condición de parada del mismo, puede ser mediante la especificación por parte del usuario de un número máximo de iteraciones o cuando G' esté formada por un único vértice.

OSLOM puede ser adaptado al caso de redes ponderadas y redes dirigidas mediante la generalización del cálculo de la significación estadística de los nodos con respecto a las comunidades. Para el caso de las

redes ponderadas, en vez de tener en cuenta solamente las relaciones topológicas del vértice mediante el análisis de su grado esperado en el MGAN, se debe incorporar la intensidad con que se relaciona, teniendo en cuenta el peso de sus aristas. Por otro lado, para lidiar con la dirección de las aristas se deben analizar de manera independiente las aristas que inciden desde el grupo hacia el vértice y las que se dirigen desde el vértice hasta el grupo.

Otra gran ventaja de OSLOM es que puede usarse para monitorear la evolución de las comunidades en redes dinámicas. Este análisis se lleva a cabo combinando las configuraciones de la red en diferentes instantes de tiempo, lo que resulta un estudio mucho más realista y descriptivo que analizar cada instantánea de manera independiente. Como se ha explicado, una característica de OSLOM es que puede comenzar su ejecución a partir de un cubrimiento inicial de la red que se desea procesar. En vista de esto, se puede utilizar el cubrimiento obtenido por OSLOM en un instante de tiempo t, para iniciar el procesamiento de la instantánea asociada al instante de tiempo $t+1$. De este modo, el cubrimiento asociado al instante $t+1$ puede interpretarse como un refinamiento del cubrimiento obtenido en el instante t. Esto permite un mejor observamiento de la evolución de cada comunidad entre dos instantáneas consecutivas de la red.

De manera general, la complejidad computacional de OSLOM es de $O(n^2)$, aunque puede llegar a tener un costo real mucho menor dependiendo de las características estructurales de la red que se desea procesar. Para procesar redes de gran tamaño, OSLOM debe ser combinado con técnicas de detección de comunidades más eficientes, de manera que inicie su funcionamiento sobre un cubrimiento o partición inicial de la red.

En distintos estudios experimentales [74,75], se muestra que OSLOM obtiene buenos resultados al procesar redes que presentan bajo nivel de traslape entre sus comunidades. Así mismo, los grupos obtenidos en OSLOM son, por la manera en que se crean, poco probables de encontrar en grafos aleatorios. Esto constituye una gran ventaja a la hora de distinguir si un grupo de vértices forma una comunidad o si las relaciones existentes entre estos son sencillamente resultado de una interacción aleatoria. Sin embargo, una gran deficiencia de OSLOM es que puede llegar a clasificar como comunidades unitarias a un gran número de vértices. Además la naturaleza no determinista del algoritmo representa una clara dificultad a la hora de su aplicación ya que no se puede contar con cierto grado de estabilidad en sus resultados. Otra desventaja de su uso está dada por la presencia de varios parámetros que deben ser ajustados por parte del usuario, dependiendo de la naturaleza de los datos representados.

En [76], se presenta el algoritmo iLCD (intrinsic Longitudinal Community Detection) el cual es capaz de detectar comunidades con traslape en redes dinámicas. En iLCD se representa la red dinámica a partir de un modelo incremental de cambio (ver sección 2.4), donde solo se tienen en cuenta cambios de inserción de aristas. Como configuración inicial de la red se considera una red vacía, mientras que la secuencia de cambios se forma a partir de conjuntos de aristas que se ordenan por el tiempo en que son insertadas en la red. Por cada nuevo conjunto de aristas de la secuencia de cambios, se conforman las comunidades de la red resultante de la inserción de dichas aristas, a partir de las comunidades de la configuración anterior de la red. Después de la inserción de cada arista, se verifica si es posible crear una nueva comunidad (o grupo semilla en el contexto de expansión local). Para crear una nueva comunidad, se debe formar producto de la inserción, un grupo inicial que cumpla cierta condición especificada por el usuario. Los autores proponen considerar como un grupo inicial válido a un k-clique, cuyo tamaño k pueda variar dependiendo de la densidad de la red.

Para expandir las comunidades se agregan aquellos vértices de la vecindad de la comunidad que cumplan condiciones asociadas a las nociones de conectividad y robustez. En cuanto a la conectividad, se tiene en cuenta la cantidad de vértices de la comunidad que el candidato puede alcanzar por caminos de longitud menor o igual que dos. En cuanto a robustez, se mide la cantidad de vértices de la comunidad que

el candidato puede alcanzar por más de un camino de longitud menor o igual que dos. Estos valores deben ser al menos como el promedio de los valores correspondientes a los vértices de la comunidad. Después de procesar cada conjunto de aristas de la secuencia de cambios, se mezclan las comunidades detectadas que sean muy similares entre sí. Esta fase depende de un parámetro especificado por el usuario para definir qué tan similares tienen que ser las comunidades que deben mezclarse.

El algoritmo iLCD tiene una complejidad computacional de $O(nC^2)$, siendo C el número de comunidades de la red. La principal limitación de iLCD es que realiza el análisis de cada inserción de aristas de manera independiente cada una. Otra debilidad del algoritmo, es que requiere del ajuste de los parámetros asociados a la formación de grupos semillas y al umbral de similitud entre las comunidades usado en la fase de mezcla.

De manera general, la efectividad de los algoritmos descritos en esta sección dependen notablemente de la selección de grupos semillas. Para obtener resultados de calidad es necesario llevar a cabo el análisis a partir de un conjunto de comunidades iniciales a partir de las cuales se pueda cubrir una gran parte de la red, o al menos las zonas de interés. Por otro lado, es necesario reducir el traslape entre las regiones exploradas durante la expansión de los distintos grupos, en aras de optimizar la eficiencia del algoritmo, evitando realizar operaciones redundantes.

3.4. Algoritmos basados en agrupamiento de enlaces

Usualmente, en las redes que modelan escenarios reales se cumple que los vértices pertenecientes a la misma comunidad interactúan de acuerdo a un tipo específico de relación. Por ejemplo, en redes sociales las comunidades suelen formarse a partir de círculos de amigos, vínculos familiares, intereses profesionales, entre otros tipos de relaciones. En la figura 9 se muestra una pequeña red donde las comunidades pueden definirse a partir del tipo de relación existente entre sus elementos. Como se ilustra en la figura, puede ocurrir que ni siquiera existan aristas puentes entre comunidades, y que las comunidades solo se conecten mediante los vértices que tienen en común. En muchas ocasiones resulta más simple detectar las aristas pertenecientes a la misma comunidad, que determinar como se conectan las comunidades entre sí [3]. En vista de esto, se han propuesto distintas técnicas capaces de detectar comunidades con traslape desde una perspectiva centrada en las aristas, en lugar de centrarse en los vértices. Estos algoritmos se conocen como algoritmos de agrupamiento de enlaces.

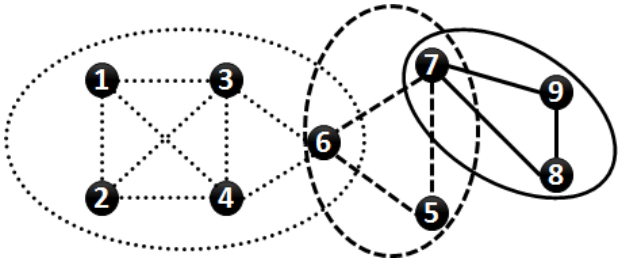

Fig. 9. Ejemplo de agrupamiento de enlaces.

Los algoritmos que siguen este enfoque, asumen que en las redes complejas reales cada arista está asociada a un tipo específico de relación. En función de esto, realizan una partición de las aristas de la red, formando comunidades de enlaces que representan el mismo tipo de relación. Luego, por cada una de las comunidades de enlaces se obtiene una comunidad de vértices a partir de los nodos que pueden ser alcanzados con las aristas de la comunidad de enlace. Si aristas incidentes en un mismo nodo pertenecen

a diferentes comunidades de enlaces, entonces dicho nodo pertenece a distintas comunidades de vértices permitiendo el traslape entre estas.

La idea de detectar comunidades con traslape a partir de agrupamientos sobre las aristas de la red, resulta bastante natural. Sin embargo, estos algoritmos también utilizan definiciones ambiguas de comunidad por lo que no existe una superioridad teórica clara con respecto a las técnicas tradicionales [3]. Por otro lado, estos algoritmos deben ser especialmente eficientes cuando los vértices de la red se conectan mediante diferentes tipos de enlace [3]. Una de las principales ventajas de este tipo de algoritmos es que las comunidades detectadas se caracterizan por presentar un alto grado de homogeneidad en sus aristas internas. Esto puede ser muy útil, a la hora de interpretar los resultados obtenidos.

El agrupamiento sobre las aristas puede realizarse utilizando técnicas tradicionales de agrupamiento de vértices sobre un grafo auxiliar conocido como *grafo de líneas* o *line graph*. El grafo de líneas asociado a una red compleja, no es más que una nueva red donde los vértices son las aristas de la red original. Dos vértices del grafo de líneas aparecen conectados, si sus aristas correspondientes en la red original comparten algún vértice en común. El mayor punto débil de este modelo de representación de la red, es que si un vértice tiene grado k en el grafo original, entonces representará un k-clique en el grafo de líneas, por lo que los vértices de mayor grado en el grafo original jugarán un papel mucho más predominante en los grafos de líneas.

En [77], se propone el primer algoritmo basado en agrupamiento de aristas. En dicho trabajo se propone que cualquier tipo de análisis estructural llevado a cabo sobre redes complejas, puede ser realizado centrándose en las aristas en lugar de los vértices. Para llevar a cabo la detección de comunidades, se propone trabajar sobre grafos de líneas ponderados, de manera que el peso de las aristas contribuya a compensar la desigualdad entre la participación de los vértices de mayor grado y los de menor grado de la red original. Los autores proponen establecer los pesos de las aristas de manera que cada vértice v que conecte a dos aristas de la red original, incremente en $1/d_v$ el peso del enlace entre dichas aristas en el grafo de línea. En este trabajo, se proponen distintas medidas de calidad de partición basadas en propiedades estadísticas de distintos procesos dinámicos que ocurren en la red. Estas medidas extienden el concepto de modularidad (ver sección 2.2.2) mediante la utilización de caminos aleatorios. Con el objetivo de detectar las comunidades del grafo de líneas, se optimizan los valores de estas medidas usando cualquier algoritmo de DCD basado en optimización de medidas de calidad. En vista de esto, la complejidad computacional de esta técnica depende del algoritmo de DCD utilizado. El modelo de representación de grafo de líneas usado en este método fue extendido para el procesamiento de redes ponderadas en [78].

En [79], se propone un algoritmo de agrupamiento de enlaces que no requiere de la formación de un grafo de líneas. En esta propuesta se agrupan las aristas de la red siguiendo un enfoque jerárquico algomerativo. La función de similitud de aristas propuesta en este algoritmo se define solo para aristas que comparten un vértice en común, ya que se supone que estas aristas serán más similares que aquellas que están desconectadas. Luego, la función de similitud entre las aristas $\{v_i, v_j\}$ y $\{v_j, v_k\}$, se denota como $S(\{v_i, v_j\}, \{v_j, v_k\})$ y se determina como la razón entre los vértices que tienen en común la egonet de v_i y la de v_j, sobre la cantidad de vértices diferentes pertenecientes a dichas egonets. Para seleccionar los niveles del dendograma de mayor calidad, se propone una medida de calidad de partición denominada *densidad D* que se basa en la *densidad de aristas* de cada comunidad. La densidad de aristas de cada comunidad se define como el número de aristas de la comunidad, normalizado por la menor y la mayor cantidad de aristas que se requieren para conectar dichos nodos, se denota como D_C y se determina mediante la expresión:

$$D_C = \frac{m_c - (n_c - 1)}{n_c(n_c - 1)/2 - (n_c - 1)}.$$

Luego, la densidad D de una partición se define como la media ponderada de la densidad de aristas de todas las comunidades donde el peso es la cantidad de aristas real de cada comunidad. En ese mismo trabajo, la similitud entre las aristas fue extendida para procesar redes ponderadas y redes dirigidas. La complejidad computacional de este algoritmo es $O(m^3)$, lo que hace prohibitivo su aplicación en redes de gran tamaño.

En [80], se propone el algoritmo GaoCD (Genetic algorithm for overlapping Community Detection). Este algoritmo sigue un enfoque genético para agrupar las aristas de la red. Los algoritmos de este tipo se basan en una estrategia evolutiva para buscar de manera aproximada o exacta las soluciones a problemas de optimización. La idea central de estos algoritmos consiste en ir explorando una población de individuos cuyas características se definen a partir de la evaluación de la función que se desea optimizar. De esta población se seleccionan los individuos que tienen más probabilidades de sobrevivir y se crean nuevos individuos realizando operaciones de *mutación* entre los mejores, variando de esta manera la población de individuos. La función de evaluación utilizada en GaoCD para definir la selección de los individuos, es la densidad D propuesta en [79], siendo esta la medida de calidad que se desea optimizar. Para realizar las operaciones requeridas por GaoCD, las comunidades se modelan mediante una representación genética, que no es más que una secuencia de los identificadores de las aristas de la comunidad. Es decir, cada individuo de la población se representa mediante m genes $(g_1, g_2, \ldots, g_m)$, donde cada gen representa el identificador de una arista. Esta secuencia de identificadores debe cumplir que si g_j es el identificador de la arista e_i entonces las aristas e_i y e_j son adyacentes. Esta representación facilita la realización de las operaciones genéticas tales como mutación y selección natural. La transformación de este código genético a comunidades de vértices puede llevarse a cabo de manera lineal. La complejidad computacional de GaoCD es de $O(g * s(m+n))$, donde g es la cantidad de generaciones de individuos analizadas y s es el tamaño de la población utilizado. A pesar de la flexibilidad y robustez que representa el enfoque genético para la optimización de la medida de calidad, se introduce el alto grado de dificultad y los problemas inherentes a este tipo de enfoque, tales como el ajuste correcto de parámetros.

La principal dificultad de los algoritmos descritos en esta sección, es que suponen que la interacción existente entre cualquier par de vértices, puede ser manejada en función del tipo de relación dominante entre dichos vértices. Es decir, por cada par de vértices conectados en la red, se asume que solo se relacionan mediante un único tipo de vínculo. En consecuencia, cada arista solo puede pertenecer a una única comunidad en el agrupamiento realizado. Esta suposición no siempre se cumple, de hecho en las redes reales es muy común que las entidades se relacionen de múltiples maneras. Volviendo al ejemplo de las redes sociales, si dos personas son familiares y cursaron estudios juntos, estas personas solo estarán relacionadas en una comunidad, cuando deberían estar relacionadas en las comunidades de estudio y de familia. Aunque esto no implica necesariamente que dichas personas pertenezcan a comunidades diferentes, puede incidir negativamente en la identificación de los roles de cada individuo en su comunidad ya que pueden dejar de tenerse en cuenta muchas aristas internas de las comunidades.

3.5. Algoritmos basados en procesos dinámicos de la red

En esta sección se describen los algoritmos que modelan el problema de detección de comunidades a partir de procesos que pueden ocurrir en las redes complejas. Este tipo de procesos se caracterizan por ser dinámicos y dependen de la manera en que interactúan los elementos de la red. En la subsección 3.5.1 se describen los algoritmos que definen las comunidades a partir de fenómenos de propagación entre los elementos. Posteriormente, en la subsección 3.5.2 se analiza un interesante enfoque basado en teoría de juegos.

3.5.1. Algoritmos basados en fenómenos de difusión

Algunos trabajos de detección de comunidades [81,82,83] proponen definir las comunidades usando distintos fenómenos de propagación, que ocurren frecuentemente en las redes reales. Un ejemplo de estos fenómenos es la difusión de información entre los nodos de la red. Siguiendo este enfoque, se puede decir que las comunidades son aquellos conjuntos de nodos que pueden ser agrupados mediante la propagación de alguna propiedad o acción en la red [14].

La mayoría de las técnicas basadas en fenómenos de difusión solo son capaces de detectar comunidades disjuntas. No obstante, dada la simplicidad y el bajo costo computacional de estos algoritmos, algunos han sido extendidos para la detección de comunidades con traslape. Tal es el caso de los *algoritmos basados en propagación de etiquetas*. En este tipo de algoritmos la información se representa mediante etiquetas y se intenta simular la propagación de dichas etiquetas en la red. Intuitivamente, se asume que la difusión de información es más intensa entre los vértices pertenecientes a una misma comunidad.

Existen dos componentes principales entre las distintas técnicas que siguen este enfoque. Estas componentes son: los mecanismos usados para la propagación de las etiquetas entre los nodos de la red y las reglas que definen cómo procesar la información recibida desde otros nodos.

El algoritmo LPA [81] (Label Propagation Algorithm) fue el primer método capaz de detectar las comunidades de una red a partir de la propagación de las etiquetas de sus vértices. Este algoritmo agrupa los vértices de la red en comunidades disjuntas, es libre de parámetros y tiene una complejidad lineal con respecto al número de vértices y aristas de la red, lo que permite su aplicación en redes de gran tamaño. La idea en que se basa LPA es que cada vértice pertenece a la comunidad que contiene el mayor número de sus vecinos. La estructura general de LPA se puede describir como sigue. Inicialmente a cada vértice se le asigna una etiqueta cuyo valor esté asociado a la semántica del modelado de la red. Posteriormente, se lleva a cabo un proceso iterativo de propagación de etiquetas donde, en cada paso, se actualizan las etiquetas de los vértices. La etiqueta de cada vértice se reemplaza por la etiqueta predominante entre sus vértices adyacentes. El proceso concluye si después de una iteración las etiquetas de los vértices no varían o si se excede un número máximo de iteraciones, especificado por el usuario. Usualmente, después de un pequeño número de iteraciones los grupos de gran densidad interna llegan a un consenso de su arista representativa. Los resultados experimentales publicados en [81] muestran que con solo cinco iteraciones se pueden alcanzar muy buenos resultados. Finalmente, los vértices asociados a la misma etiqueta, se asocian a la misma comunidad. En la figura 10 se muestran un ejemplo del funcionamiento de LPA.

Existen dos enfoques fundamentales para llevar a cabo la actualización de las etiquetas de los vértices. Por un lado, se encuentra la llamada *actualización asincrónica*, donde se involucran etiquetas de la iteración anterior y etiquetas de la iteración actual. Para realizar esta actualización, los vértices se ordenan de manera aleatoria, de modo que durante la actualización de la etiqueta de un vértice v, se consulten las etiquetas actuales de los vecinos menores que v, y las etiquetas de la iteración anterior para los vecinos mayores que v. Por otro lado, está la conocida como *actualización sincrónica*, donde se procesan únicamente las etiquetas que tenían los vértices vecinos en la iteración anterior. Estudios experimentales [81] han comprobado que el uso de la actualización asincrónica conlleva a una convergencia más rápida del algoritmo, mientras que al utilizar la actualización sincrónica se obtienen soluciones más estables, además de que facilita la implementación en paralelo del algoritmo.

Durante la actualización de las etiquetas en LPA, puede ocurrir que existan varias etiquetas presentes en igual número máximo de vecinos. En ese caso, se selecciona cualquiera de ellas de manera aleatoria. Intuitivamente, una manera de identificar el traslape entre las comunidades detectadas, es permitiendo que los vértices se asocien a todas las etiquetas predominantes en sus vecinos. Basándose en esta idea, se han propuesto distintos algoritmos de OCD que permiten asignar múltiples etiquetas a cada vértice.

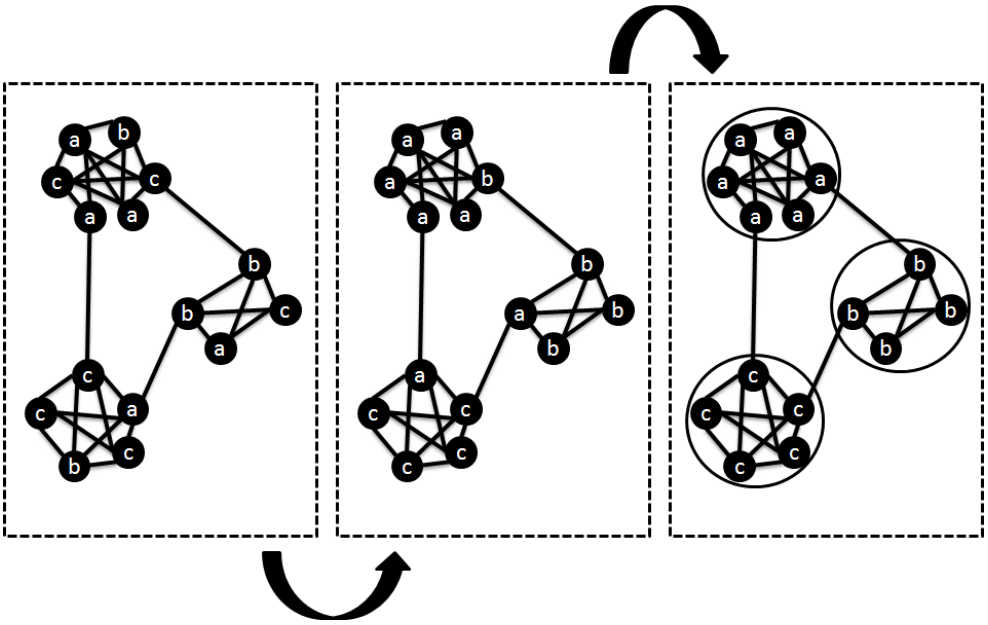

Fig. 10. Ejemplo del funcionamiento del algoritmo LPA.

En [56], se propone el algoritmo COPRA (Community Overlap Propagation Algorithm). En este algoritmo, la etiqueta de cada vértice se representa a partir de un conjunto de pares (c,b), donde c es el identificador de una comunidad y b es el grado de pertenencia del vértice en la comunidad c. De esta manera, se permite que cada vértice pueda ser asignado a múltiples comunidades. Inicialmente, cada vértice se asocia a una sola comunidad con coeficiente de pertenencia igual a 1. Luego, tal como se procede en LPA, la propagación de las etiquetas ocurre entre los vértices vecinos, siguiendo un enfoque iterativo. En cada iteración, cada vértice v actualiza de manera síncrona su etiqueta a partir de las etiquetas de los demás vértices de su egonet. La nueva etiqueta se forma con los pares (c,b) tal que existe al menos un vecino de v que pertenece a la comunidad c y b es el promedio normalizado de los grados de pertenencia, con respecto a c, de todos los vecinos de v que estén asociados a dicha comunidad. Los grados de pertenencias se normalizan en un rango de $[0,1]$, de modo que la sumatoria de dichos valores de como resultado 1. Al obtener las etiquetas de esta manera, los vértices pueden quedar asociados a comunidades a las que no pertenecen realmente; e.g., a una comunidad que esté asociada a solo uno de sus vértices adyacentes. Por tal motivo, los autores proponen registrar en las etiquetas únicamente los pares (c,b) que cumplan que $b \geq 1/k$, siendo k un parámetro especificado por el usuario. De esta manera se limita el número máximo de comunidades que pueden ser asociadas a un vértice. En caso de que todos los grados de pertenencia de un vértice sean menor que $1/k$, entonces se selecciona el par con mayor coeficiente de pertenencia. Si existe más de un par con el mismo coeficiente de pertenencia máximo, entonces se selecciona uno de manera aleatoria. Luego, se eliminan de la etiqueta el resto de los pares que no fueron seleccionados. El proceso de selección propuesto hace que COPRA tenga un comportamiento no determinista, lo que afecta la estabilidad del algoritmo.

El proceso iterativo llevado a cabo en COPRA necesita condiciones de parada más complejas que las propuestas para LPA. Tras cada iteración, el número de identificadores de comunidades se mantiene igual o se reduce. El proceso se detiene cuando la información de las comunidades, relacionadas con las etiquetas y el número mínimo de vértices asociado a cada etiqueta desde la ultima reducción, permanece invariable.

En COPRA se pueden generar comunidades desconectadas. Por tal motivo, se realiza una fase de post-procesamiento en la que se sustituyen las comunidades de este tipo por sus componentes conexas, como comunidades independientes. En esta fase final, también se eliminan aquellas comunidades que se encuentran completamente contenidas en otras.

La complejidad temporal de COPRA sobre redes poco densas es $O(k^3n + L(kn\log k))$, siendo L la cantidad de iteraciones y n la cantidad de vértices de la red. En vista de esto, COPRA escala de manera lineal para valores pequeños de k. Por otro lado, COPRA es fácilmente generalizable para el procesamiento de redes ponderadas. Los autores proponen incorporar el peso de las aristas en el análisis del coeficiente de pertenencia de cada vértice v. Esto se logra obteniendo en lugar del promedio, la media ponderada de los grados de pertenencia de los vértices vecinos de v, utilizando como pesos en la fórmula, los pesos de sus respectivas aristas incidentes en v.

Una de las mayores dificultades de COPRA es el ajuste del parámetro k para controlar el máximo número de comunidades que pueden ser asignadas a un vértice. Con este parámetro se impone una restricción global para todos los vértices de la red, sin tener en cuenta las características particulares de cada uno de ellos. Si en la red que se desea procesar, existen vértices que pertenecen a muchas comunidades y además existe otro grupo de vértices asociados a un número mucho menor de comunidades, entonces puede resultar muy difícil una selección adecuada de valores para este parámetro. Para valores grandes de k, los vértices que están asociados a menos comunidades pueden ser asignados a comunidades donde realmente no pertenecen. Por otro lado, para valores pequeños de k, los vértices que pertenecen a un mayor número de comunidades pueden ser asignados a solo una parte de estas. Con el objetivo de solucionar este problema, se propuso en [84] el algoritmo BMLP (Balanced Multi-Label Propagation Algorithm).

El algoritmo BMLP presenta una estructura general casi idéntica a la de COPRA, ya que sigue un enfoque sincrónico para la actualización de las etiquetas y además, utiliza el mismo criterio de parada para el proceso iterativo de propagación. BMLP también realiza la fase de post-procesamiento propuesta en COPRA para el refinamiento de los resultados obtenidos. La principal diferencia entre estos algoritmos es que BMLP actualiza las etiquetas de cada vértice de modo que no limita el número de comunidades asociadas a dicho vértice y asigna sus respectivos grados de pertenencia de manera balanceada. En cada iteración, se actualiza la etiqueta del vértice v a partir de la información de los vecinos tal como se hace en COPRA. No obstante, la selección de los pares que se mantendrán finalmente en la etiqueta, se realiza de manera diferente. Por cada vértice, se obtiene de su etiqueta el mayor coeficiente de pertenencia b_{max}. Luego, los pares seleccionados para mantenerse en la etiqueta son los pares (c,b) tal que $b/b_{max} \geq p$, donde $p \in [0,1]$ es un parámetro especificado por el usuario para controlar el balance entre los coeficientes. Otra característica propia de BMLP es el modo en que se inicializan las etiquetas de los vértices. Los autores proponen un método llamado RC (Rough Cores) para asignar las etiquetas iniciales de cada vértice, de manera que cada una pueda estar asociada a varias comunidades. Este algoritmo obtiene un conjuntos de cliques maximales del menor tamaño posible, formados a partir de dos vértices semilla diferentes para cada uno de los grupos. Luego, cada uno de estos grupos se asocia a una etiqueta, que se le asigna a cada uno de los vértices que contienen. Este método puede ser aplicado como fase de inicialización en cualquier algoritmo de propagación de etiquetas. El algoritmo BMLP no es tan eficiente como COPRA ya que presenta una complejidad computacional para redes dispersas de $O(L(n\log n))$. No obstante, es mucho más estable y detecta cubrimientos de mayor calidad.

En [85], se propone el algoritmo SLPA (Speaker-listener Label Propagation Algorithm) como otra extensión del algoritmo LPA para detectar comunidades con traslape. El funcionamiento de este algoritmo se basa en la manera en que las personas se comunican. Cada vértice de la red puede asumir indistintamente dos roles durante la propagación de las etiquetas. Cuando el nodo se desempeña como proveedor de información asume el rol de *hablante*, mientras que cuando se desempeña como receptor de información,

asume el rol de *oyente*. En los algoritmos de propagación de etiquetas descritos anteriormente, los vértices ignoran la información procesada en iteraciones anteriores. En cambio, SLPA provee a cada nodo de una memoria en donde se registra el historial de etiquetas que le han sido asignadas al nodo durante la fase de propagación. De esta manera, se acumula la información asociada a la frecuencia con que se le asigna cada etiqueta. Otras componentes fundamentales en SLPA son la *regla del habla* y la *regla del escucha*, las cuales definen cómo un nodo hablante difunde la información a sus vecinos y cómo un nodo oyente procesa la información proveniente de sus vecinos, respectivamente. Las reglas propuestas por los autores se basan en la preferencia que tienen los humanos de comunicar las opiniones que se discuten con más frecuencia. La regla del habla indica que el vértice hablante le comunique a sus vecinos la etiqueta que se le ha asignado en mayor cantidad de ocasiones. Por otro lado, la regla del escucha indica que el vértice oyente registre en su memoria, la etiqueta que más vecinos le envían. Estas reglas pueden ser modificadas sin afectar el resto de la dinámica del algoritmo.

La estructura general de SLPA tiene ciertas similitudes con la de los algoritmos descritos anteriormente. En la fase de inicialización de etiquetas, a cada uno de los vértices se le asigna una única etiqueta. Luego, comienza el proceso iterativo de propagación. En cada iteración se recorren de manera aleatoria cada vértice de la red, de manera que dicho vértice asume el rol de oyente y sus vecinos asumen el rol de hablantes. En SLPA se lleva a cabo una actualización asincrónica, es decir, si un vértice hablante ya ha sido oyente en la misma iteración entonces puede usar la información recibida para comunicarla a sus vecinos. Debido a esto y a la memoria de cada nodo, en SLPA se combinan la efectividad del enfoque asíncrono con la estabilidad que trae el análisis de la información pasada. La propagación finaliza cuando se realizan un número máximo de iteraciones L, especificado por el usuario. Los autores indican que para valores de $L \geq 20$ se obtienen resultados de buena calidad. Una vez terminada la fase de propagación, se ejecuta una fase de post-procesamiento donde se conforman las comunidades a partir de las memorias de cada nodo. En esta última fase, cada etiqueta l es asociada a una comunidad y la probabilidad de seleccionar dicha etiqueta en la memoria de un nodo v, se interpreta como el grado de pertenencia de v en la comunidad asociada a l.

La complejidad computacional de SLPA es de $O(Ln)$ para redes dispersas y de manera general escala lineal con respecto a la cantidad de aristas de la red. Debido a la independencia entre las componentes de SLPA, el algoritmo puede ser fácilmente extendido para procesar redes ponderadas generalizando la reglas de interacción entre los nodos durante la propagación. Otra ventaja del algoritmo es que no requiere de ningún conocimiento sobre el número de comunidades, ni el grado de pertenencia de los vértices en dichos grupos.

La principal ventaja de los algoritmos descritos en esta sección está dada por su eficiencia computacional, ya que escalan de manera casi lineal con respecto al tamaño de la red. Otra bondad en este sentido es que estas técnicas pueden ser fácilmente paralelizadas, ya que el proceso de propagación de etiquetas se puede llevar a cabo de manera independiente por cada vértice. Además, estos algoritmos son capaces de procesar redes dirigidas de manera natural, ya que el flujo de información o propagación de estados entre los vértices de la red, depende de la dirección de las aristas que relacionan dichos elementos.

La mayoría de los algoritmos basados en propagación de etiquetas presentan la dificultad de ser no deterministas. Es decir, para distintas ejecuciones del algoritmo con igual ajuste de parámetros, se pueden obtener cubrimientos diferentes. No obstante, en [86] se indica que en las redes reales pueden existir distintos agrupamientos con altos valores de modularidad. En vista de esto, obtener todas las soluciones posibles puede ser de gran importancia práctica, ya que cada una puede incidir de manera diferente en la aplicación donde se utiliza.

3.5.2. Otro enfoque basado en la dinámica de la red

En [87] se propone el algoritmo CFG (Community Formation Game), que sigue un enfoque basado en teoría de juegos. En este algoritmo se aborda el problema de detección de comunidades mediante un juego de estrategia donde los vértices de la red representan agentes independientes. Cada agente v debe seleccionar un conjunto L_v de comunidades a las que desea pertenecer, lo cual representará la estrategia de juego seguida por dicho agente. Dado que el conjunto L_v puede tener más de una comunidad, el algoritmo permite que exista traslape entre comunidades. Además, L_v puede estar vacío, lo que puede resultar muy útil para identificar posibles anomalías.

El objetivo del juego para los agentes es maximizar un valor de utilidad asociado a cada uno de ellos. Dicha utilidad depende del conjunto de comunidades seleccionado por el agente y se determina por una función de ganancia y una función de pérdida. La combinación de ganancia y pérdida para definir la utilidad asociada a la pertenencia de un agente en una comunidad se ajusta a muchos escenarios reales. Esto puede verse como una manera de interpretar las ventajas y desventajas de pertenecer a los posibles grupos. En distintos tipos de redes sociales, la membrecía de una persona en una determinada afiliación puede tener varios costos asociados a la iniciación y a la permanencia en el grupo. En redes de comunicación, las personas vinculadas a una compañía local pueden tener preferencias en determinados servicios de comunicación interna entre los asociados de la compañía y por otro lado presentar dificultades para comunicarse con las personas ajenas a este grupo.

Los autores proponen utilizar una variación de la medida modularidad para definir la función de ganancia. En vista de esto, la función de ganancia para un agente i en una red G, se define como:

$$Q_i = \frac{1}{2m}\sum_{j\in V(G)}\left[A_{ij}\hat{\delta}(i,j) - \frac{d_i d_j}{2m}\cdot|L_i \cap L_j|\right], \text{ donde:}$$

$\hat{\delta}(i,j)$ es una función binaria que indica si los vértices i y j pertenecen a alguna comunidad en común.

Por otro lado, la función de pérdida propuesta es muy simple y consiste en multiplicar el número de comunidades que han sido seleccionadas por el agente por una constante mayor que cero, que puede ser especificada por el usuario. Dicha constante se puede interpretar como el costo asociado a la pertenencia del agente en cada comunidad. Finalmente, la utilidad de cada agente se define como la diferencia entre su ganancia y su pérdida

El cubrimiento que se obtiene mediante la aplicación de CFG se forma a partir del conjunto de comunidades seleccionado por cada agente, cuando el juego llega a un estado de equilibrio de Nash. El juego alcanza dicho estado de equilibrio cuando ninguno de los agentes puede incrementar su utilidad a partir de una *pequeña* variación en su estrategia. En el contexto de CFG, los agentes solo pueden variar las comunidades de su estrategia a partir de tres operaciones básicas que son: unirse a una nueva comunidad, abandonar una de sus comunidades, o cambiar una de sus comunidades por una nueva.

En vista de lo anterior, los autores proponen un procedimiento sencillo para llegar a un estado de equilibrio. Inicialmente, cada agente está asociado a una comunidad unitaria. Luego, se lleva a cabo un proceso iterativo, donde en cada iteración se selecciona aleatoriamente un agente y se le aplica la operación básica que más incremente su utilidad. Este proceso finaliza cuando ningún vértice puede aumentar su utilidad.

Los autores demostraron que dada la función de utilidad propuesta, y las operaciones que pueden realizar los agentes, el estado de equilibrio se alcanza siguiendo el procedimiento anterior en a lo sumo $O(m^2)$ pasos, por lo que la complejidad de CFG es a lo sumo cuadrática con respecto al número de aristas. Esta complejidad puede limitar en gran medida la aplicación de CFG en muchas situaciones prácticas. No obstante, dada la simplicidad del algoritmo y el enfoque local que sigue para optimizar la utilidad de los agentes, el algoritmo puede ser fácilmente paralelizado. Por otro lado, las funciones de ganancia

y de pérdida propuestas en CFG son relativamente simples y puede que en ocasiones no se ajusten a las características del sistema complejo que se modela.

3.6. Algoritmos basados en técnicas para detectar comunidades disjuntas

En las redes complejas que modelan escenarios reales, puede existir gran variedad en cuanto al nivel de traslape entre sus comunidades. Para obtener una correcta estructura de comunidades resulta necesario seleccionar el tipo de algoritmo indicado para cada red. Los algoritmos de DCD no son capaces de determinar el traslape entre las comunidades. Por otro lado, si la red se caracteriza por tener bajo nivel de traslape entre sus comunidades, entonces este tipo de algoritmos arrojan los mejores resultados en cuanto a eficacia [88]. En esta sección se discuten los principales enfoques propuestos para extender estos algoritmos, de manera que puedan aplicarse en la detección de comunidades con traslape.

En [88], se propone la metodología PEACOCK que consiste en aplicar algoritmos de DCD sobre una nueva red con comunidades disjuntas que representen implícitamente el traslape entre las comunidades de la red original. Para la creación de la nueva red, utiliza una de las ideas fundamentales de CONGA (ver sección 3.1), la cual plantea que los vértices con alto grado de centralidad suelen pertenecer a varias comunidades. PEACOCK consta de tres fases principales, en la primera, se crea una red que representará la red original. Para construir esta otra red, se parte de la red original y se realiza un proceso iterativo donde, en cada paso se dividen los vértices cuya centralidad sobrepase un umbral previamente definido por el usuario. Este proceso finaliza cuando todos los vértices de la red tienen una centralidad inferior al umbral. Para la división de los vértices se utiliza el criterio IS propuesto en CONGA. En la segunda fase, se detectan las comunidades existentes en la nueva red, usando un algoritmo de DCD. La tercera fase, se encarga de obtener las comunidades de la red original a partir de la partición obtenida en la fase anterior. Para agrupar en comunidades la red original, se procede de manera que por cada comunidad C detectada en la nueva red, se obtiene una comunidad formada por los vértices de que dieron origen a los vértices de C.

Otra metodología propuesta en [89], consiste en aplicar directamente un algoritmo de DCD sobre la red original y realizar un post-procesamiento de la partición devuelta. Esta metodología se basa en la idea de que un buen algoritmo de DCD debe dar una aproximación inicial aceptable de las comunidades de la red. Luego, solo resulta necesario realizar un refinamiento de las comunidades devueltas para obtener el traslape entre estas. Para esto se sigue una estrategia de expansión local (ver sección 3.3.2) sobre las comunidades obtenidas inicialmente. La métrica utilizada en esta metodología para determinar la calidad de la comunidad se llama *fuerza* e indica la relación existente entre las aristas internas de la comunidad y todas las aristas que inciden en los vértices de la comunidad. El traslape se controla a partir de un parámetro definido por el usuario, que indica el máximo nivel de traslape que el usuario desea que exista entre las comunidades detectadas. Esto conlleva a un mayor grado de dificultad del uso de esta técnica.

La complejidad temporal de estas técnicas queda determinada por la eficiencia del algoritmo utilizado en la detección de comunidades disjuntas. Esto representa una ventaja, ya que los algoritmos más eficientes de este tipo son menos costosos que los algoritmos de OCD. Por otro lado, en la mayoría de los experimentos publicados en [88,89], la calidad de los resultados es comparable con la obtenida por muchos de los algoritmos de OCD existentes.

En [90] se propone el algoritmo DEMON (Democratic Estimate of the Modular Organization of a Network), donde se tiene en cuenta la perspectiva local de cada nodo para identificar la estructuración en comunidades de la red. En muchos contextos resulta natural la idea de utilizar la egonet de un nodo v para analizar la organización de la red desde la perspectiva de v. Por ejemplo, en redes sociales un individuo

solo conoce bien cómo se relacionan los elementos involucrados en su vecindad, y muy poco o nada del resto de la estructura de la red.

Inicialmente, la perspectiva que tiene cada nodo sobre los grupos de la red se obtiene analizando la estructura en comunidades de la egonet de dicho nodo, ignorando al propio vértice y las aristas incidentes en él. Para determinar estas comunidades locales, los autores proponen utilizar el algoritmo LPA (ver sección 3.5.1) por su eficiencia computacional. No obstante, para esta tarea puede ser aplicado cualquier algoritmo de DCD. En el análisis estructural de la egonet de cada vértice v, se ignora la influencia de v en esta red por dos razones principales. En primer lugar, la participación de v en las comunidades de la red puede ser juzgada en el análisis de sus vértices adyacentes. En segundo lugar, dado que v conecta a todos los vértices de su egonet, dos vértices que pertenezcan a comunidades diferentes pueden estar estrechamente relacionados en la egonet.

Posteriormente, se determina la estructuración global de la red a partir de la información que conoce cada nodo sobre los grupos en que participan sus vecinos. Para esto, se lleva a cabo un proceso de unificación de información, donde se mezclan las micro comunidades obtenidas previamente desde la perspectiva local de cada nodo. En esta fase de unificación se mezclan las comunidades cuyo grado de traslape sea mayor que un umbral $epsilon$ especificado por el usuario.

Al igual que en los algoritmos descritos anteriormente, la complejidad temporal de DEMON depende del algoritmo DCD utilizado en su funcionamiento interno. Independientemente a esto, dado que para cada nodo de la red DEMON realiza un número limitado de operaciones, donde se involucra solamente la egonet de dicho nodo, se puede afirmar que DEMON depende notablemente de la distribución de los grados de los vértices de la red. En vista de esto, DEMON puede ser especialmente efectivo en redes con pocos vértices de grado alto que interactúen con una gran parte del resto de la red. Además, como el aporte de cada nodo se puede obtener de manera independiente, esta fase del algoritmo puede ser fácilmente paralelizada.

Una de las mayores ventajas de DEMON es que permite procesar adiciones de vértices y/o aristas, actualizando las comunidades de la red a partir de las comunidades obtenidas previamente. Para esto no se requiere procesar nuevamente todos los vértices, ya que los únicos nodos que varían su aporte al proceso de unificación son los involucrados en los cambios sufridos por la red. Por lo tanto, solamente es necesario obtener la perspectiva local de estos nodos y mezclar esta información con las comunidades previamente detectadas. Otra gran ventaja, es que DEMON puede ser aplicado siguiendo un enfoque distribuido, dividiendo la red en múltiples partes de menor tamaño y llevando a cabo la ejecución del algoritmo sobre cada una de estas partes de manera independiente. La división realizada sobre la red debe ser una partición de los vértices de la misma, de manera que la egonet de cada nodo esté completamente contenida en la parte donde el nodo pertenece. La combinación de los resultados obtenidos por cada una de las partes, se lleva a cabo usando la propia fase de unificación de DEMON.

La dependencia del parámetro ε es una dificultad en la aplicación práctica de DEMON. El correcto ajuste de este parámetro requiere de un conocimiento previo por parte del usuario sobre la estructura de la red. Además, este parámetro incide notablemente en la efectividad del algoritmo, ya que implícitamente regula la densidad de las comunidades detectadas.

Utilizar métodos de solución de problemas de menor complejidad también ha sido aplicado en la detección de comunidades difusas. En [91] se propone la técnica MakeFuzzy para dotar a los algoritmos de OCD, de un procedimiento que permita determinar los grados de pertenencia de los vértices a sus respectivas comunidades. El criterio propuesto determina el grado de pertenencia de un vértice v a una comunidad C como la razón entre el número de vértices adyacentes a v que pertenecen a C y la cantidad de vértices de C. En la experimentación documentada en [91], se comprobó que complementar algoritmos de OCD con MakeFuzzy, puede incrementar la efectividad de estos al procesar redes con comunidades

difusas. En particular, para algoritmos que tienden a asignar muchos vértices a comunidades incorrectas, el incremento es sustancial.

3.7. Otros enfoques basados en asignación difusa

Como se mencionó en la sección 2 pueden existir dos tipos de traslape entre las comunidades. A lo largo de este trabajo, se han descrito varios algoritmos capaces de detectar el grado de pertenencia de los vértices a sus respectivas comunidades. En esta sección se abordan dos algoritmos de detección de comunidades difusas que no se corresponden con ninguna de las metodologías descritas en secciones anteriores. Estos algoritmos utilizan en su funcionamiento una extensión al caso difuso de la función de modularidad vista en la sección 2.2.2. Dado que las metodologías propuestas en estos algoritmos no son una simple optimización de la medida de modularidad, estos trabajos no fueron abordados en la sección 3.3

En [62] se propone un algoritmo de FCD en el cual se combinan técnicas de representación espectral, agrupamiento difuso y una extensión de modularidad como medida de calidad. Para representar la red se sigue un enfoque de agrupamiento espectral de grafos [92,93] y en tal sentido, la red se representa como la matriz espectral E_K formada por los primeros K vectores propios de la generalización del sistema vectorial $Ax = tDx$, donde: A es la matriz de adyacencia de la red, D es la matriz diagonal asociada a A de manera que $D_{ii} = \sum_j A_{ij}$ y K es un parámetro especificado por el usuario para acotar superiormente la cantidad de comunidades que se desean detectar.

Una vez representada la red en un espacio vectorial euclideano, se lleva a cabo un agrupamiento difuso, explorando cada uno de los posibles números de comunidades c que se desean detectar, $2 \leq k \leq K$. Para esto se seleccionan de E_K los primeros c vectores y se normalizan usando la norma euclideana. Dichos vectores se agrupan usando el algoritmo *C-means difuso* [94,95], aunque puede usarse cualquier otro algoritmo de agrupamiento que realice una asignación difusa de los elementos. Finalmente, se escoge como resultado final, el cubrimiento asociado al valor de c que alcance el maximo valor de calidad, de acuerdo al a funcion utilizada. Los autores determinan la calidad de un cubrimiento mediante una extensión difusa de modularidad definida sobre una matriz U_C de asignación difusa donde el valor asociado al elemento$U_C[v,c]$, representa el grado de pertenencia del vértice v con respecto a la comunidad c. Esta variante de modularidad se determina mediante la expresión:

$$Q(U_C) = \sum_{1 \leq c \leq C} \left[\frac{A(\bar{V}_c, \bar{V}_c)}{A(V,V)} - \left(\frac{A(\bar{V}_c, V)}{A(V,V)} \right)^2 \right],$$

donde $\bar{V}_c$ son los vértices de la red cuyo grado de pertenencia con respecto a la comunidad c es mayor que un umbral λ, especificado por el usuario; $A(\bar{V}_c, \bar{V}_c) = \sum_{i \in \bar{V}_c, j \in \bar{V}_c} (U_{ic} + U_{jc})/2$, $A(V,V) = \sum_{i \in V, j \in V} A_{ij}$ y $A(\bar{V}_c, V) = A(\bar{V}_c, \bar{V}_c) + \sum_{i \in \bar{V}_c, j \in V \bar{V}_c} (U_{ic} + (1 - U_{jc}))/2$.

La principal debilidad de este algoritmo está dada por su alta complejidad computacional, ya que se debe solucionar un sistema de ecuaciones lineales cuya matriz debe ser de gran tamaño (como usualmente pasa en las redes reales), y posteriormente, se llevan a cabo distintos agrupamientos difusos sobre los véctores representativos de la red, lo cual puede ser un proceso lento. La dependencia de los parámetros K y λ constituye otra debilidad del algoritmo, ya que en muchas ocasiones no se puede determinar a priori, el número exacto de comunidades que forman la estructura interna de la red, ni el grado de pertenencia de los vértices a los grupos. Además el algoritmo es muy sensible al ajuste del parámetro K, tanto en eficiencia como en eficacia.

En [63] se presenta otro algoritmo de FCD, el cual interpreta la detección de comunidades difusas como un problema de optimización no lineal. Este algoritmo requiere de una matriz $\bar{S}$ que indica la similitud entre los vértices de la red, la cual debe ser especificada por el usuario. Esto supone un alto grado de complejidad en la aplicación del algoritmo, ya que la red debe ser analizada previamente por especialistas que obtengan dicha información.

La idea central del algoritmo consiste obtener la matriz de asignación difusa U que minimice la función objetivo descrita por:

$$F = \sum_{v_i \in V} \sum_{v_j \in V} (\bar{S_{ij}} - S_{ij})^2,$$

donde $\bar{S_{ij}}$ indica la similitud esperada por el usuario entre los vértices v_i y v_j, mientras que S_{ij} indica la similitud real existente entre dichos vértices. Esta función de similitud se calcula como:
$S_{ij} = \sum_{1 \leq c \leq k} U_{ci} U_{cj}$.

Este problema de optimización puede ser resuelto usando diferentes heurísticas, pero la más utilizada ha sido *recocido simulado*. Para identificar la cantidad k de comunidades en que se debe agrupar la red, se incrementa el valor de k hasta que no se mejore la calidad del cubrimiento. Para medir la calidad del agrupamiento, los autores proponen una extensión difusa de modularidad definida por la expresión:

$$Q(P) = \frac{1}{2m} \sum_{C \in P} \sum_{i,j \in C} \left[A_{ij} - \frac{d_i d_j}{2m} \right] S_{ij}.$$

La complejidad temporal de este algoritmo se ve afectada por el hecho de que debe realizar el proceso de optimización por cada valor de k con los que se experimenta. En cuanto a la complejidad espacial, el algoritmo debe reservar en memoria las dos matrices de similitud, lo cual es sumamente costoso. Estos factores pueden dificultar la aplicación del algoritmo en redes de gran tamaño.

4. Discusión

En esta sección, se realiza una comparación cualitativa de los algoritmos descritos en la sección 3. Inicialmente, en la sección 4.1 se describen las distintas características de los algoritmos de detección de comunidades, abordadas en la comparación. Finalmente, en la sección 4.2, se presenta el análisis comparativo entre los algoritmos y se formulan conclusiones a partir de este.

4.1. Aspectos a tener en cuenta en la comparación

En el contexto de la evaluación y comparación de los algoritmos que detectan comunidades en redes complejas, resulta interesante el análisis de distintas características inherentes al problema. En este acápite, se describen los aspectos utilizados en nuestra comparación, se explica por qué es necesario su análisis y se indican los valores que se muestran en la tabla comparativa 1.

- Complejidad computacional:
  El aumento de las capacidades de generación y almacenamiento de grandes volúmenes de datos, ha provocado que la complejidad computacional sea uno de los principales aspectos evaluativos, para cualquier tipo de algoritmos. Las redes que modelan sistemas complejos reales, suelen estar compuestas por millones de vértices y aristas. La necesidad de lidiar con estos volúmenes de información, ha provocado un gran cambio en cuanto a la manera de manejar este tipo de datos. Algoritmos con complejidad computacional de $O(n^2)$ o superior, resultan poco útiles en múltiples escenarios reales.

Mientras más baja sea la complejidad temporal, mayor es el volúmen de datos que puede procesar el algoritmo. En vista de esto, en la comparación se analizará la complejidad computacional de los algoritmos, en lo siguiente referida como *complejidad*. Esta variable tomará valor *alta* cuando la complejidad del algoritmo sea al menos de $O(n^3)$, *media* cuando la complejidad sea mayor que $O(n\log n)$ y *baja* cuando la complejidad sea menor o igual que $O(n\log n)$.

- Ajuste de parámetros:
  Para evaluar un algoritmo, es necesario tener en cuenta el análisis de los parámetros que requieren para su ejecución. Un buen algoritmo de detección de comunidades debe depender de la menor cantidad de parámetros posible. Esto permite, que los usuarios puedan usar el algoritmo sin ningún tipo de dificultad relacionada con el ajuste de los parámetros. Además, una selección inadecuada de parámetros puede incidir notablemente en la efectividad del algoritmo. En modo de resumen, la columna Parámetros muestra solamente la cantidad de parámetros de los cuales depende el algoritmo, ya que la manera en que estos inciden en el algoritmo se explica en la descripción individual de los mismos.

- Tipo de red:
  Como se ha explicado, la detección de comunidades es un problema orientado a aplicación. Para cada dominio de aplicación, la red debe cumplir ciertos requerimientos y características que permitan modelar de manera efectiva el sistema complejo que se analiza. En muchos casos, tales como las redes que modelan el funcionamiento de la Web y las redes de comunicaciones, se necesita tener en cuenta la dirección de las aristas para representar el flujo de información o de señales. La dirección en las aristas también se tiene en cuenta para redes donde las relaciones entre las entidades modeladas no son simétricas. Por otro lado, también puede ser necesario ponderar las aristas de acuerdo a algún criterio para representar la intensidad de la relación o el grado de afinidad entre los elementos. La mayoría de las redes reales sufren transformaciones constantemente, por lo que en muchos casos se requiere representar la evolución de la red en el tiempo. En la columna Redes se especifica si el algoritmo es capaz de procesar redes dirigidas, ponderadas y/o dinámicas, mostrando las letras D, P y N, respectivamente.

- Definición de comunidades:
  La forma en que se definen las comunidades, es otro factor importante a la hora de seleccionar un algoritmo de detección de comunidades. En redes pequeñas donde resulta necesario hacer una análisis funcional de la red, se prefiere usar los algoritmos que definen las comunidades desde una perspectiva global. Por otro lado, en redes complejas muy grandes, donde la propiedad del *mundo pequeño* o *small world* se acentúa, son mas populares los enfoques locales para definir las comunidades. En este sentido, en la columna Definición se pueden asignar dos valores, *local* y *global*.

- Tipo de traslape:
  Por defecto todos los algoritmos que se abordan en este trabajo, detectan comunidades con traslape. No obstante, en algunas aplicaciones no solo es necesario saber si un vértice está asignado a más de una comunidad, sino que también resulta interesante conocer el grado de pertenencia de cada vértice a sus respectivas comunidades. En la columna Fuzzy, se marcan los algoritmos capaces de detectar comunidades difusas.

- Análisis de relaciones jerárquicas:
  En muchas aplicaciones prácticas, resulta interesante conocer las relaciones jerárquicas existentes en-

tre las comunidades de la red [96,38,97]. La forma más popular de representar estas jerarquías, es mediante el uso de dendogramas. Aunque en la detección de comunidades de manera general, existen muchos algoritmos capaces de generar estos dendogramas, son muy pocos los trabajos que detectan las relaciones jerárquicas y el traslape entre las comunidades, al mismo tiempo. En la columna Jerarquía, se indican los algoritmos estudiados que detectan las relaciones jerárquicas entre las comunidades.

4.2. Comparación cualitativa de los algoritmos

En esta subsección se realiza una comparación cualitativa entre los algoritmos analizados en la sección 3, teniendo en cuenta los aspectos descritos en la subsección anterior. En estudio comparativo se excluyeron algunas técnicas abordadas en el reporte. Tal es el caso de las técnicas propuestas en [88,89,91], las cuales constituyen metodologías usadas para complementar otros algoritmos de detección de comunidades, con el fin de adaptarlos a requerimientos más complejos. Los algoritmos *CPMw* y *CPMd* se abordan en el estudio como extensiones del algoritmo *CPM* para procesar redes ponderadas y redes dirigidas, respectivamente.

En la tabla 1 se muestra la evaluación de los algoritmos seleccionados, de acuerdo a las características de interés. Como se puede apreciar, algunos valores de la tabla aparecen afectados por el supra-índice $*$. A continuación se explica el significado de este supra-índice, para cada una de las columnas de la tabla donde se utiliza:

- Columna Parámetros:
  En esta columna aparecen con supra-índices las clasificaciones de los algoritmos *CPM, Evans & Lambiotte, MONC y GaoCD*. El algoritmo *CPM* es marcado con el supra-índice por el parámetro que usa para procesar redes ponderadas, ya que en este estudio, el algoritmo CPMw se considera como una extensión de CPM. El algoritmo *MONC* aparece marcado porque solamente utiliza los parámetros en la fase de post-procesamiento, para llevar a cabo la asignación difusa e identificar las relaciones jerárquicas entre las comunidades. Por otro lado, el funcionamiento del algoritmo *Evans & Lambiotte* solo requiere de los parámetros que usa el algoritmo de DCD que utiliza para detectar las comunidades en el grafo de líneas. El algoritmo *GaoCD* se marca porque involucra los parámetros del enfoque genético que utiliza para detectar las comunidades.

- Columna Jerarquía:
  Para entender las relaciones jerárquicas existentes entre las comunidades, no basta con generar el dendograma que las representa. En algunos casos, la interpretación de estos dendogramas puede ser muy compleja, debido a la cantidad de cubrimientos representados o a que no existe ninguna manera de determinar cuál es el cubrimiento que mejor describe la estructuración de la misma. Los algoritmos *CONGA, CONGO*, así como los propuestos por *Tyler et al. y Ahn et al.* fueron marcados con un *D* porque generan dendogramas difíciles de interpretar y por consiguiente, poco útiles en situaciones prácticas. Por otro lado, los algoritmos *LFK, GCE y MONC* se marcan con el supra-índice *M* porque detectan las relaciones jerárquicas entre las comunidades siguiendo un enfoque multiresolución. Como se explicó en la sección 2.3, este tipo de enfoque depende del ajuste de un parámetro de resolución por parte del usuario, lo que complejiza la aplicación de dichos algoritmos.

A partir de los datos incluidos en la tabla 1, se pueden hacer algunas observaciones parciales. En cuanto a la eficiencia computacional, se puede observar que la mayoría de los algoritmos (21 de 28) tienen

Tabla 1. Comparación entre los algoritmos atendiendo a los aspectos seleccionados.

Algoritmo	Complejidad	Parámetros	Redes	Definición	Fuzzy	Jerarquía
CONGA [47]	*alta*	0	-	global	-	$\surd^D$
CONGO [48]	*media*	1	-	local	-	$\surd^D$
Tyler et al. [46]	*media*	1	-	global	-	$\surd^D$
Rees et al. [49]	*media*	0	-	local	-	-
FRINGE [50]	*media*	0	-	local	-	-
CPM [51]	*media*	1^*	P,D	local	-	-
SCP [52]	*baja*	2	P	local	-	-
Palla et al. [2]	*alta*	1	N	local	-	-
IncrCPM [55]	*baja*	1	N	local	-	-
FD [59]	*media*	3	-	global	$\surd$	-
EAGLE [60]	*alta*	1	-	global	-	$\surd$
RARE-IS [68]	*media*	0	-	local	-	-
LA-IS^2 [69]	*media*	0	-	local	-	-
LFK [66]	*media*	1	P,D	local	-	$\surd^M$
GCE [72]	*media*	3	-	local	-	$\surd^M$
MONC [73]	*media*	2^*	-	local	$\surd$	$\surd^M$
OSLOM [74]	*media*	3	P,D,N	local	-	$\surd$
iLCD [74]	*media*	2	N	local	-	-
Evans & Lambiotte [77]	*alta*	0^*	P	global	-	-
Ahn et al. [79]	*alta*	0	P,D	global	-	$\surd^D$
GaoCD [80]	*media*	0^*	-	global	-	-
COPRA [56]	*baja*	1	P,D	local	$\surd$	-
BMLP [84]	*baja*	1	P,D	local	$\surd$	-
SLPA [85]	*baja*	1	P,D	local	$\surd$	-
CFG [87]	*baja*	1	-	local	-	-
DEMON [90]	*baja*	1	N	local	-	-
Nepusz et al. [63]	*media*	1	-	global	$\surd$	-
Zhang et al. [62]	*media*	1	-	global	$\surd$	-

una complejidad computacional media o alta. Entre los algoritmos que tienen complejidad baja, SCP es muy sensible al parámetro que controla la densidad de las comunidades, CFG solo es capaz de procesar redes estáticas, no dirigidas y no ponderadas. Ninguno de los algoritmos de complejidad baja es capaz de organizar jerárquicamente las comunidades detectadas, y solamente *IncrCPM y DEMON* son capaces de procesar redes dinámicas.

En cuanto al tipo de traslape que detectan, solamente (7 de 28) algoritmos realizan una asignación difusa de los elementos a sus respectivas comunidades. De estos, los algoritmos *COPRA, BMLP y SLPA* basados en propagación de etiquetas, son los únicos que tienen una complejidad temporal baja. Por otro lado, el algoritmo MONC obtiene el grado de pertentencia de los vértices a las comunidades asociadas, en una fase de post-procesamiento. Para esto, requiere del ajuste de un parámetro por parte del usuario, lo que dificulta notablemente una correcta asignación difusa de los elementos.

En cuanto a los tipos de redes que son capaces de procesar, muy pocos algoritmos (9 de 28) tienen en cuenta el peso en las aristas, mientras que (7 de 28) son capaces de procesar la dirección de los enlaces. Por otro lado, solamente *IncrCPM, OSLOM, iLCD, DEMON* y el propuesto por *Palla et al.* (6 de 28) son capaces de monitorear la evolución de las comunidades en redes dinámicas. El algoritmo de *Palla et al.* no utiliza la estructuración en comunidades detectadas en instantes de tiempos anteriores para actualizar las comunidades de las siguientes instantáneas. Por otro lado, los algoritmos *IncrCPM y OSLOM* son los únicos capaces de procesar la eliminación de vértices y aristas, como cambios en la red de un instante de tiempo a otro. En cuanto a la complejidad computacional, el único algoritmo de complejidad baja que detecta comunidades en redes dinámicas es DEMON. Sin embargo, DEMON no es capaz de procesar redes ponderadas ni redes dirigidas, y tampoco detecta las relaciones jerárquicas existentes entre las comunidades de las redes.

Como se puede apreciar, el único algoritmo multifuncional capaz de procesar los tres tipos de redes, es el algoritmo OSLOM. Este algoritmo tiene una complejidad computacional media y requiere de tres parámetros, lo cual puede dificultar su aplicación en redes de gran tamaño. En este sentido, las técnicas basadas en propagación de etiquetas permiten manejar de manera natural la dirección y el peso en las aristas. Además, utilizando la información relacionada con los grados de pertenencia de las etiquetas se puede realizar una asignación difusa de los vértices a las comunidades. De igual modo, este enfoque ha sido extendido para el procesamiento de redes dinámicas [98], mostrando muy buenos resultados. Por tal motivo, el uso de esta metodología puede ser muy efectivo a la hora de diseñar un algoritmo multifuncional.

5. Conclusiones y recomendaciones sobre trabajo futuro

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 un análisis crítico del estado del arte en el contexto de la detección de comunidades con traslape. 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.

De manera adicional, se presentan algunas líneas de investigación relacionadas con la detección de comunidades que deben ser abordadas como trabajo futuro ya que resultan de gran interés práctico.

- La búsqueda de propiedades que permitan identificar eficientemente las aristas que conectan comunidades diferentes, resulta de gran interés para muchas de las técnicas descritas en este reporte. Los algoritmos basados en centralidad, analizados en la sección 3.1, consisten precisamente en la detección de estas aristas puentes. En cuanto a los algoritmos basados en agrupamiento de enlaces, la detección de este tipo de aristas resulta muy útil para reducir el tamaño del grafo de líneas asociado a la red. Usualmente, el grafo de líneas tiene una mayor cantidad de vértices y aristas que la red original, sobre todo cuando la red es densa. Sin embargo, para formar las comunidades de enlaces solo resulta necesario agrupar aquellas aristas de la red original que conecten vértices pertenecientes a una misma comunidad. Luego, ignorar las aristas puentes de la red original durante la creación del grafo de líneas puede reducir considerablemente el tamaño de dicho grafo. Por otro lado, la detección de las aristas puentes puede ser de gran importancia en aplicaciones donde se analiza el funcionamiento general de la red, la manera en que interactúan las comunidades o las relaciones jerárquicas existentes entre estas. Con base en lo anterior, una línea de investigación que debe ser abordada como trabajo futuro es la detección eficiente de este tipo de aristas. En este sentido, se han propuesto algunas estrategias locales para mejorar la eficiencia de las técnicas propuestas para este fin (ver sección 3.1). Dichas estrategias siguen presentando dificultades e incluso pueden requerir conocimiento previo por parte del usuario para el ajuste de parámetros que regulan qué tan local será el análisis.

- Por otro lado, a lo largo del trabajo se ha evidenciado que las medidas de calidad juegan un papel fundamental en la detección de comunidades. No obstante, todavía resulta necesario proponer nuevas

medidas, más efectivas que puedan ser aplicadas en cualquier tipo de escenario práctico [23]. De hecho, la ausencia de mecanismos confiables para la evaluación y comparación de los resultados, es una de las principales razones del enorme número de trabajos de detección de comunidades propuestos en la literatura [3]. En las secciones 2.1 y 2.2 se analizan las diferencias entre el enfoque local y el enfoque global para definir las comunidades y evaluar la calidad de las particiones realizadas. En vista de esto, puede resultar de gran importancia proponer medidas de calidad donde se combinen ambos enfoques, con lo que se pudiera lograr un mayor entendimiento tanto de la estructura general de la red, como de los principios que rigen la organización interna en las comunidades.

- La mayoría de los esfuerzos en la detección de comunidades han sido dirigidos precisamente a obtener la estructuración en comunidades de la red. Interpretar lo que muestran las comunidades detectadas sobre las características del sistema modelado, resulta de gran importancia ya que brinda una idea de cómo aplicar los resultados obtenidos en las diferentes aplicaciones. No obstante, según nuestro conocimiento no se ha propuesto ninguna técnica capaz de encontrar una descripción intencional de las comunidades que detecta. Esto se debe principalmente a la ambigüedad existente en la noción de comunidad. En vista de esto, establecer una definición más específica del concepto *comunidad* aún sigue siendo sumamente necesario. Otra línea de investigación muy interesante en este sentido, puede ser proponer nuevas técnicas de detección de comunidades capaces de vincular la información topológica de la red, con la información no estructural asociada a los vértices, siempre que se cuente con tales datos. De esta manera se pueden obtener comunidades más realistas y se facilita la interpretación de los resultados obtenidos. Siguiendo esta idea se han propuesto distintos enfoques [99,100,32,101]; no obstante, en ninguno de estos trabajos se detecta el traslape existente entre las comunidades.

  Hasta nuestro conocimiento, los algoritmos de OCD propuestos en la literatura se basan en un análisis puramente topológico de la red. Esto permite que se puedan aplicar en un gran número de escenarios prácticos ya que no requieren de ningún tipo de información adicional, aparte de la relacionada con la interacción entre los elementos. No obstante, se han presentado distintos trabajos donde se refleja la importancia de incorporar la información inherente a los elementos, para lograr una mayor efectividad en la detección de comunidades de la red [102,103]. Por ejemplo, en redes sociales y redes de comunicación es muy común la presencia de *spammers*, los cuales interactúan con muchos elementos de la red sin el consentimiento o aprobación de estos. Este tipo de interacción no tiene aporte significativo en el funcionamiento de la red y puede afectar la interpretación de la estructura de la red. Este tipo de fenómenos está presente en otro tipos de redes complejas y pueden afectar la eficacia de las técnicas que solo tienen en cuenta la topología de la red.

- Como se ha explicado en secciones anteriores, la detección de comunidades se puede llevar a cabo en distintos escenarios de aplicación. No obstante, la mayoría de las técnicas propuestas solo son capaces de procesar un tipo específico de redes. Por otro lado, existen muy pocos métodos capaces de extraer toda la información estructural que se desea obtener. En este sentido, o dejan de detectar el traslape o las relaciones jerárquicas existentes entre las comunidades, o no son capaces de monitorear la evolución de estos grupos en redes dinámicas. Hasta donde llega nuestro conocimiento, existen muy pocos algoritmos de OCD que no presenten estas dificultades. Por tal motivo, el diseño de nuevos algoritmos que sean aplicables en múltiples escenarios sigue siendo de un gran interés práctico.

- A pesar de la importancia del análisis estructural de las redes dinámicas, existen muy pocos algoritmos capaces de monitorear la evolución de las comunidades en este tipo de redes. De hecho, la mayoría no son capaces de utilizar los grupos previamente detectados para identificar las comunidades resultantes

de transformaciones en la red. Por otro lado, el traslape existente entre las comunidades puede influir en la manera en que evolucionan estos grupos [59]. En vista de esto, se deberían desarrollar metodologías que incorporen esta información en los algoritmos de detección de comunidades dinámicas, con la intención de obtener resultados más estables y de validar la significación del traslape.

- Existen marcadas diferencias, en cuanto a eficiencia computacional, entre los primeros algoritmos de detección de comunidades y los que han sido propuestos recientemente. Las redes complejas reales se caracterizan por relacionar enormes volúmenes de información y usualmente sobrepasan los millones de vértices y aristas. Por tal motivo, proponer nuevas técnicas cada vez más escalables que sean capaces de procesar redes de tal magnitud, constituye una línea de gran interés en el contexto de detección de comunidades. Una manera de reducir la complejidad de estos métodos es llevando a cabo el análisis (al menos en las fases más costosas del algoritmo) sobre una red auxiliar de menor tamaño que tenga propiedades similares a las de la red original. Por ejemplo, en la sección 3.1 se analizó la variante propuesta en [46] para optimizar el algoritmo GN. En esta variante, la centralidad se calcula teniendo en cuenta solo la contribución de un limitado número de vértices de la red, seleccionados de manera aleatoria. Otra línea que resulta de interés para el análisis estructural de redes de gran tamaño, es proponer técnicas distribuidas y/o paralelas que permitan la ejecución simultánea de múltiples procesos de detección de comunidades.

Referencias bibliográficas

1. Fortunato, S., Barthélemy, M.: Resolution limit in community detection. Proceedings of the National Acadamy of Sciences of the United States of America (PNAS) **104**(1) (2007) 36–41
2. Palla, G., Barabasi, A.L., Vicsek, T.: Quantifying social group evolution. Nature **446**(7136) (April 2007) 664–667
3. Fortunato, S.: Community detection in graphs. Physics Reports **486** (2010) 75 – 174
4. Erdös, P., Rényi, A.: On random graphs i. Publicationes Mathematicae Debrecen **6** (1959) 290
5. Spirin, V., Mirny, L.: Protein complexes and functional modules in molecular networks. Proc Natl Acad Sci USA **100** (2003) 12123–12128
6. Chen, J., Yuan, B.: Detecting functional modules in the yeast protein–protein interaction network. Bioinformatics **22**(18) (September 2006) 2283–2290
7. Krishnamurthy, B., Wang, J.: On network-aware clustering of web clients. In: SIGCOMM. (2000) 97–110
8. Reddy, P.K., Kitsuregawa, M., Sreekanth, P., 0002, S.S.R.: A graph based approach to extract a neighborhood customer community for collaborative filtering. In Bhalla, S., ed.: DNIS. Volume 2544 of Lecture Notes in Computer Science., Springer (2002) 188–200
9. Nguyen, N.P., Dinh, T.N., Tokala, S., Thai, M.T.: 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, New York, NY, USA, ACM (2011) 85–96
10. Rice, S.A.: The identification of blocs in small political bodies. American Political Science Review **21**(03) (1927) 619–627
11. Lancichinetti, A., Fortunato, S.: Community detection algorithms: a comparative analysis (2009) cite arxiv:0908.1062 Comment: 12 pages, 8 figures. The software to compute the values of our general normalized mutual information will be soon available at http://santo.fortunato.googlepages.com/inthepress2.
12. Liu, G.: Community structure and detection in complex networks: A survey. (2012)
13. Prabavathi, G., Thiagarasu, V.: A review on overlapping community detection algorithms (2013)
14. Coscia, M., Giannotti, F., Pedreschi, D.: A classification for community discovery methods in complex networks. Statistical Analysis and Data Mining **4**(5) (2011) 512–546
15. Girvan, M., Newman, M.E.J.: Community structure in social and biological networks. Proceedings of the National Academy of Sciences **99**(12) (June 2002) 7821–7826
16. Alba, R.D.: A graph-theoretic definition of a sociometric clique. Journal of Mathematical Sociology **3** (1973) 3–113
17. Mokken, R.J.: Cliques, clubs and clans. Quality & Quantity **13**(2) (1979) 161–173
18. Seidman, S.B., Foster, B.L.: A graph-theoretic generalization of the clique concept. Journal of Mathematical Sociology **6** (1978) 139–154

19. Seidman, S.B.: Network structure and minimum degree. Social Networks **5**(3) (1983) 269 – 287
20. Borgatti, S., Everett, M., Shirey, P.: Ls sets, lambda sets, and other cohesive subsets. Social Networks (12) (1990) 337–358
21. Radicchi, F., Castellano, C., Cecconi, F., Loreto, V., Parisi, D.: Defining and identifying communities in networks. Proc. Natl. Acad. Sci. U.S.A. **101**(9) (2004) 2658–2663
22. Newman, M.E.J., Girvan, M.: Finding and evaluating community structure in networks. Phys. Rev. E **69**(2) (February 2004) 026113
23. Almeida, H., Guedes, D., Meira Jr, W., Zaki, M.J.: Is there a best quality metric for graph clusters? In: Machine Learning and Knowledge Discovery in Databases. Springer (2011) 44–59
24. Traud, A.L., Kelsic, E.D., Mucha, P.J., Porter, M.A.: Comparing community structure to characteristics in online collegiate social networks. SIAM review **53**(3) (2011) 526–543
25. Danon, L., Diaz-Guilera, A., Duch, J., Arenas, A.: Comparing community structure identification. Journal of Statistical Mechanics: Theory and Experiment **2005**(09) (2005) P09008
26. Leskovec, J., Lang, K.J., Mahoney, M.: Empirical comparison of algorithms for network community detection. In: Proceedings of the 19th international conference on World wide web, ACM (2010) 631–640
27. Collins, L.M., Dent, C.W.: Omega: A general formulation of the rand index of cluster recovery suitable for non-disjoint solutions. Multivariate Behavioral Research **23**(2) (1988) 231–242
28. Amigo, E., Gonzalo, J., Artiles, J., Verdejo, F.: A comparison of extrinsic clustering evaluation metrics based on formal constraints. Inf. Retr. **12**(5) (2009) 613
29. Nuovo, A.G.D., Catania, V.: On external measures for validation of fuzzy partitions. In Melin, P., Castillo, O., Aguilar, L.T., Kacprzyk, J., Pedrycz, W., eds.: IFSA (1). Volume 4529 of Lecture Notes in Computer Science., Springer (2007) 491–501
30. Campello, R.J.G.B.: Generalized external indexes for comparing data partitions with overlapping categories. Pattern Recognition Letters **31**(9) (2010) 966–975
31. Ramírez, E.H., Brena, R.F., Magatti, D., Stella, F.: Topic model validation. Neurocomputing **76**(1) (2012) 125–133
32. Ding, Y.: Community detection: Topological vs. topical. Journal of Informetrics **5**(4) (2011) 498 – 514
33. Moradi, F., Olovsson, T., Tsigas, P.: An evaluation of community detection algorithms on large-scale email traffic. In Klasing, R., ed.: SEA. Volume 7276 of Lecture Notes in Computer Science., Springer (2012) 283–294
34. Noack, A.: Modularity clustering is force-directed layout. Physical Review E **79**(2) (2009) 26102
35. Arenas, A., Duch, J., Fernández, A., Gómez, S.: Size reduction of complex networks preserving modularity. New Journal of Physics **9** (2007) 176
36. Newman, M.E.J.: Analysis of weighted networks. Phys. Rev. E **70**(5) (2004) 056131
37. Leicht, E.A., Newman, M.E.J.: Community structure in directed networks. Phys. Rev. Lett. **100**(11) (March 2008) 118703
38. Sales-Pardo, M., Guimera, R., Moreira, A.A., Amaral, L.A.N.: Extracting the hierarchical organization of complex systems. Proceedings of the National Academy of Sciences **104**(39) (2007) 15224–15229
39. Bringmann, Björn y Berlingerio, M.y.B.F.y.G.A.: Learning and predicting the evolution of social networks. Intelligent Systems, IEEE **25**(4) (2010) 26–35
40. Berlingerio, M., Bonchi, F., Bringmann, B., Gionis, A.: Mining graph evolution rules. In: Machine learning and knowledge discovery in databases. Springer (2009) 115–130
41. Leung Cane Wing-ki, Lim Ee-Peng, D.L.y.W.J.: Mining interesting link formation rules in social networks. In: Proceedings of the 19th ACM international conference on Information and knowledge management, ACM (2010) 209–218
42. Sun, Y., Han, J., Aggarwal, C.C., Chawla, N.V.: When will it happen?: relationship prediction in heterogeneous information networks. In: Proceedings of the fifth ACM international conference on Web search and data mining, ACM (2012) 663–672
43. Papadopoulos, S., Kompatsiaris, Y., Vakali, A., Spyridonos, P.: Community detection in social media. Data Mining and Knowledge Discovery **24**(3) (2012) 515–554
44. Freeman, L.: A Set of Measures of Centrality Based on Betweenness. Sociometry **40** (1977) 35–41
45. Newman, M.E.: Analysis of weighted networks. Physical Review E **70**(5) (2004) 056131
46. Tyler, J.R., Wilkinson, D.M., Huberman, B.A.: Email as spectroscopy: automated discovery of community structure within organizations. (2003) 81–96
47. Gregory, S.: An algorithm to find overlapping community structure in networks. In: Proceedings of the 11th European Conference on Principles and Practice of Knowledge Discovery in Databases. PKDD 2007, Berlin, Heidelberg, Springer-Verlag (2007) 91–102
48. Gregory, S.: A fast algorithm to find overlapping communities in networks. In: Proceedings of the 2008 European Conference on Machine Learning and Knowledge Discovery in Databases - Part I. ECML PKDD '08, Berlin, Heidelberg, Springer-Verlag (2008) 408–423
49. Rees, B.S., Gallagher, K.B.: Overlapping community detection by collective friendship group inference. In Memon, N., Alhajj, R., eds.: ASONAM, IEEE Computer Society (2010) 375–379

50. Palazuelos, C., Zorrilla, M.E.: Fringe: A new approach to the detection of overlapping communities in graphs. In Murgante, B., Gervasi, O., Iglesias, A., Taniar, D., Apduhan, B.O., eds.: ICCSA (3). Volume 6784 of Lecture Notes in Computer Science., Springer (2011) 638–653
51. Palla, G., Derenyi, I., Farkas, I., Vicsek, T.: Uncovering the overlapping community structure of complex networks in nature and society. Nature **435**(7043) (2005) 814–818
52. Kumpula, J.M., Kivelä, M., Kaski, K., Saramäki, J.: Sequential algorithm for fast clique percolation. Physical Review E **78**(2) (2008) 026109
53. Farkas, I., Ábel, D., Palla, G., Vicsek, T.: Weighted network modules. New Journal of Physics **9**(6) (2007) 180
54. Palla, G., Farkas, I.J., Pollner, P., Derenyi, I., Vicsek, T.: Directed network modules. New Journal of Physics **9**(6) (2007) 186
55. Duan, D., Li, Y., Li, R., Lu, Z.: Incremental k-clique clustering in dynamic social networks. Artif. Intell. Rev. **38**(2) (2012) 129–147
56. Gregory, S.: Finding overlapping communities in networks by label propagation. New Journal of Physics **12**(10) (2010) 103018
57. Brandes, U., Delling, D., Gaertler, M., Gorke, R., Hoefer, M., Nikoloski, Z., Wagner, D.: On modularity clustering. Knowledge and Data Engineering, IEEE Transactions on **20**(2) (2008) 172–188
58. Blondel, V.D., Guillaume, J.L., Lambiotte, R., Lefebvre, E.: Fast unfolding of communities in large networks. Journal of Statistical Mechanics: Theory and Experiment **2008**(10) (2008) P10008
59. Wang, Q., Fleury, E.: Fuzziness and overlapping communities in large-scale networks. j-jucs **18**(4) (feb 2012) 457–486
60. Shen, H., Cheng, X., Cai, K., Hu, M.B.: Detect overlapping and hierarchical community structure in networks. Physica A: Statistical Mechanics and its Applications **388**(8) (2009) 1706–1712
61. Lázár, A., Ábel, D., Vicsek, T.: Modularity measure of networks with overlapping modules (2009)
62. Zhang, S., Wang, R.S., Zhang, X.S.: Identification of overlapping community structure in complex networks using fuzzy c-means clustering. Physica A: Statistical Mechanics and its Applications **374**(1) (2007) 483–490
63. Nepusz, T., Petróczi, A., Négyessy, L., Bazsó, F.: Fuzzy communities and the concept of bridgeness in complex networks. Physical Review E **77**(1) (2008) 016107
64. Chen, D., Shang, M., Lv, Z., Fu, Y.: Detecting overlapping communities of weighted networks via a local algorithm. Physica A: Statistical Mechanics and its Applications **389**(19) (2010) 4177–4187
65. Shen, H.W., Cheng, X.Q., Guo, J.F.: Quantifying and identifying the overlapping community structure in networks. Journal of Statistical Mechanics: Theory and Experiment **2009**(07) (2009) P07042
66. Lancichinetti, A., Fortunato, S., Kertész, J.: Detecting the overlapping and hierarchical community structure in complex networks. New Journal of Physics **11**(3) (2009) 033015
67. Leskovec, J., Lang, K.J., Dasgupta, A., Mahoney, M.W.: Statistical properties of community structure in large social and information networks. In: WWW '08: Proceeding of the 17th international conference on World Wide Web, New York, NY, USA, ACM (2008) 695–704
68. Baumes, J., Goldberg, M.K., Krishnamoorthy, M.S., Magdon-Ismail, M., Preston, N.: Finding communities by clustering a graph into overlapping subgraphs. IADIS AC **5** (2005) 97–104
69. Baumes, J., Goldberg, M., Magdon-Ismail, M.: Efficient identification of overlapping communities. In: Intelligence and Security Informatics. Springer (2005) 27–36
70. Page, L., Brin, S., Motwani, R., Winograd, T.: The pagerank citation ranking: Bringing order to the web. In: Proceedings of the 7th International World Wide Web Conference, Brisbane, Australia (1998) 161–172
71. Goldberg, M., Kelley, S., Magdon-Ismail, M., Mertsalov, K., Wallace, A.: Finding overlapping communities in social networks. In: Social Computing (SocialCom), 2010 IEEE Second International Conference on, IEEE (2010) 104–113
72. Lee, C., Reid, F., McDaid, A., Hurley, N.: Detecting highly overlapping community structure by greedy clique expansion. arXiv preprint arXiv:1002.1827 (2010)
73. Havemann, F., Heinz, M., Struck, A., Gläser, J.: Identification of overlapping communities and their hierarchy by locally calculating community-changing resolution levels. Journal of Statistical Mechanics: Theory and Experiment **2011**(01) (2011) P01023
74. Lancichinetti, A., Radicchi, F., Ramasco, J.J., Fortunato, S.: Finding statistically significant communities in networks. PloS one **6**(4) (2011) e18961
75. Xie, J., Kelley, S., Szymanski, B.: Overlapping community detection in networks: the state of the art and comparative study. ArXiv e-prints (October 2011)
76. Cazabet, R., Amblard, F., Hanachi, C.: Detection of overlapping communities in dynamical social networks. In: SocialCom/PASSAT'10. (2010) 309–314
77. Evans, T., Lambiotte, R.: Line graphs, link partitions, and overlapping communities. Physical Review E **80**(1) (2009) 016105
78. Evans, T.S., Lambiotte, R.: Line graphs of weighted networks for overlapping communities. The European Physical Journal B - Condensed Matter and Complex Systems **77**(2) (2010) 265–272

79. Ahn, Y.Y., Bagrow, J.P., Lehmann, S.: Link communities reveal multiscale complexity in networks. Nature **466**(7307) (2010) 761–764
80. Shi, C., Cai, Y., Fu, D., Dong, Y., Wu, B.: A link clustering based overlapping community detection algorithm. Data Knowl. Eng. **87** (2013) 394–404
81. Raghavan, U.N., Albert, R., Kumara, S.: Near linear time algorithm to detect community structures in large-scale networks (September 2007)
82. Tantipathananandh, C., Berger-Wolf, T., Kempe, D.: A framework for community identification in dynamic social networks. Proceedings of the 13th ACM SIGKDD international conference (2007) Dynamic approach Computing & algorithms.
83. Chen, W., Wang, Y., Yang, S.: Efficient influence maximization in social networks. In: KDD. (2009) 199–208
84. Wu, Z., Lin, Y.F., Gregory, S., Wan, H., Tian, S.F.: Balanced multi-label propagation for overlapping community detection in social networks. J. Comput. Sci. Technol. **27**(3) (2012) 468–479
85. Xie, J., Szymanski, B.K., Liu, X.: Slpa: Uncovering overlapping communities in social networks via a speaker-listener interaction dynamic process. In: Data Mining Workshops (ICDMW), 2011 IEEE 11th International Conference on, IEEE (2011) 344–349
86. Good, B., Montjoye, Y.D., Clauset, A.: Performance of modularity maximization in practical contexts. Physical Review E **81**(4) (2010) 046106
87. Chen, W., Liu, Z., Sun, X., Wang, Y.: A game-theoretic framework to identify overlapping communities in social networks. Data Mining and Knowledge Discovery **21**(2) (2010) 224–240
88. Gregory, S.: Finding overlapping communities using disjoint community detection algorithms. In: Complex Networks. Springer (2009) 47–61
89. Wang, X., Jiao, L., Wu, J.: Adjusting from disjoint to overlapping community detection of complex networks. Physica A: Statistical Mechanics and its Applications **388**(24) (2009) 5045–5056
90. Coscia, M., Rossetti, G., Giannotti, F., Pedreschi, D.: Demon: a local-first discovery method for overlapping communities. In 0001, Q.Y., Agarwal, D., Pei, J., eds.: KDD, ACM (2012) 615–623
91. Gregory, S.: Fuzzy overlapping communities in networks. Journal of Statistical Mechanics: Theory and Experiment **2011**(02) (2011) P02017
92. Verma, D., Meila, M.: A comparison of spectral clustering algorithms. (2003)
93. Ng, A.Y., Jordan, M.I., Weiss, Y.: On spectral clustering: Analysis and an algorithm. In: Advances in Neural Information Processing Systems 14, MIT Press (2001) 849–856
94. Dunn, J.C.: A Fuzzy Relative of the ISODATA Process and Its Use in Detecting Compact Well-Separated Clusters. Journal of Cybernetics **3**(3) (1973) 32–57
95. Bezdek, J.C.: Pattern recognition with fuzzy objective function algorithms. Kluwer Academic Publishers (1981)
96. Pons, P., Latapy, M.: Post-processing hierarchical community structures: Quality improvements and multi-scale view. Theor. Comput. Sci. **412**(8-10) (2011) 892–900
97. Ravasz, E., Somera, A., Mongru, D., Oltvai, Z., Barabási, A.: Hierarchical organization of modularity in metabolic networks. Science **297**(5586) (2002) 1551
98. Xie, J., Chen, M., Szymanski, B.K.: Labelrankt: Incremental community detection in dynamic networks via label propagation. In: Proceedings of the Workshop on Dynamic Networks Management and Mining, ACM (2013) 25–32
99. Labatut, V., Balasque, J.M.: Detection and interpretation of communities in complex networks: practical methods and application. In: Computational Social Networks. Springer (2012) 81–113
100. Li, D., Ding, Y., Shuai, X., Bollen, J., Tang, J., Chen, S., Zhu, J., Rocha, G.: Adding community and dynamic to topic models. Journal of Informetrics **6**(2) (2012) 237 – 253
101. Zhao, Z., Feng, S., Wang, Q., Huang, J.Z., Williams, G.J., Fan, J.: Topic oriented community detection through social objects and link analysis in social networks. Knowledge-Based Systems **26**(0) (2012) 164 – 173
102. Zhou, D., Manavoglu, E., Li, J., Giles, C.L., Zha, H.: Probabilistic models for discovering e-communities. In: WWW '06: Proceedings of the 15th international conference on World Wide Web, New York, NY, USA, ACM (2006) 173–182
103. Li, D., He, B., Ding, Y., Tang, J., Sugimoto, C.R., Qin, Z., Yan, E., Li, J., Dong, T.: Community-based topic modeling for social tagging. In Huang, J., Koudas, N., Jones, G.J.F., Wu, X., Collins-Thompson, K., An, A., eds.: CIKM, ACM (2010) 1565–1568

RT_028, enero 2015
Aprobado por el Consejo Científico CENATAV

Editor: Lic. Lucía González Bayona
Diseño de Portada: Di. Alejandro Pérez Abraham
RNPS No. 2143
ISSN 2072-6260
Indicaciones para los Autores:
Seguir la plantilla que aparece en www.cenatav.co.cu
C E N A T A V
7ma. A No. 21406 e/214 y 216, Rpto. Siboney, Playa;
La Habana. Cuba. C.P. 12200
Impreso en Cuba