Modulo4_comunidades

Jorge Fábrega

Version 2023

Introducción a la Detección de Comunidades:

La detección de comunidades en gráficos es un problema complejo y no resuelto, importante en varios campos como la sociología, la biología y la informática. La intuición es simple cuando se conoce el resultado al que se desea llegar, pero no es trivial llegar a métricas que conduzcan a él aún cuando se sepa lo que se busca. Por ejemplo, un puzzle en el que tenemos una imagen del resultado final nos dice con claridad el resultado al que debemos llegar, pero si hay piezas parecidas en dos o tres partes de la figura, agrupar adecuadamente las piezasque deberían ir juntas y ubicarlas donde deben ir puede ser muy complicado. Aquí pasa lo mismo, **una comunidad es esencialmente la organización de vértices en grupos, con más aristas conectándolos entre sí (grupos más densos) que con otros vértices fuera del grupo**. El problema de implementación es que el método usado podría estar agrupando vértices que tienen algo en común, pero que no son parte del mismo grupo (del mismo modo que podríamos agrupar piezas que, por ejemplo, tienen el mismo color pero que en realidad corresponden a distintas partes del puzzle).

Veamos algunos ejemplos de algoritmos para detección de comunidades:

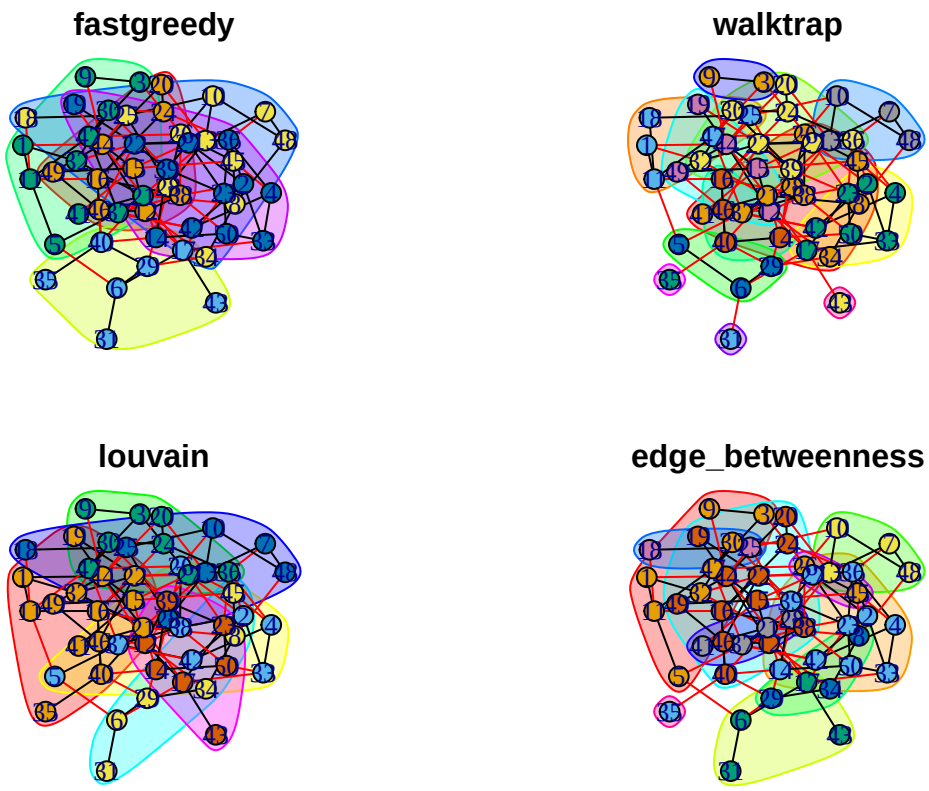
Fastgreedy: Este es un algoritmo especialmente útil en redes grandes. Parte con cada nodo siendo una comunidad y va agrupando.

Walktrap: Si uno camina por un condominio cerrado, tarde o temprano vuelve a caminar por las mismas calles. Eso inspira este algoritmo.

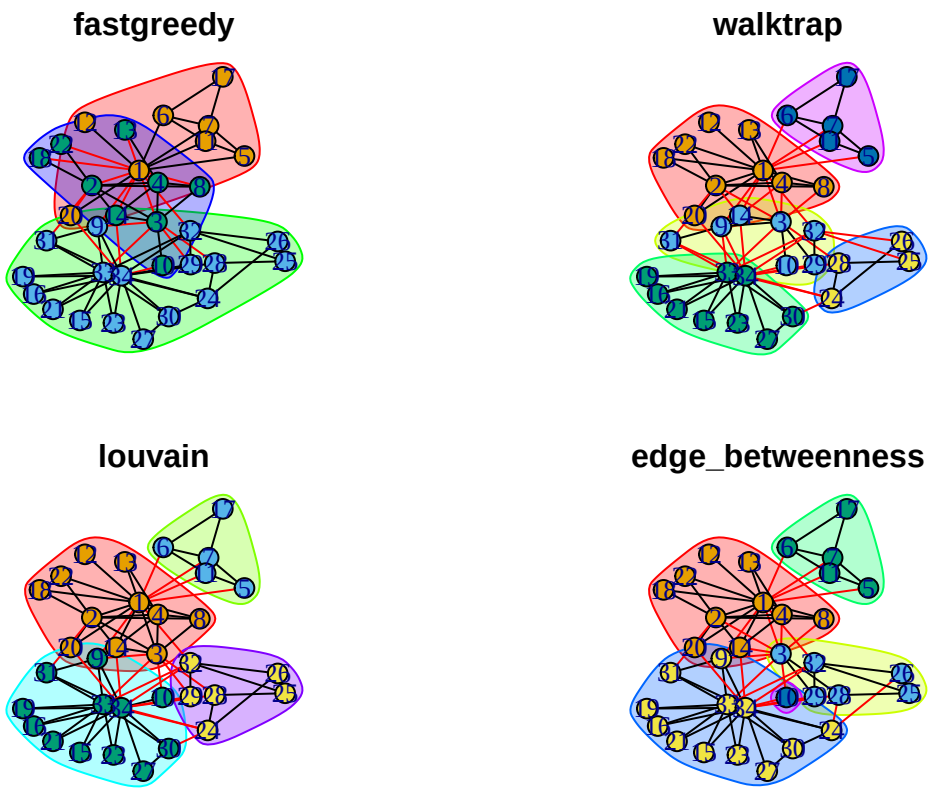
Louvain: Parte igual que Fastgreedy pero va testeando qué pasa si cambia a los nodos de comunidad

Edge Betweeness: Este algoritmo calcula la centralidad de intermediación de las aristas (Girvan-Newman)

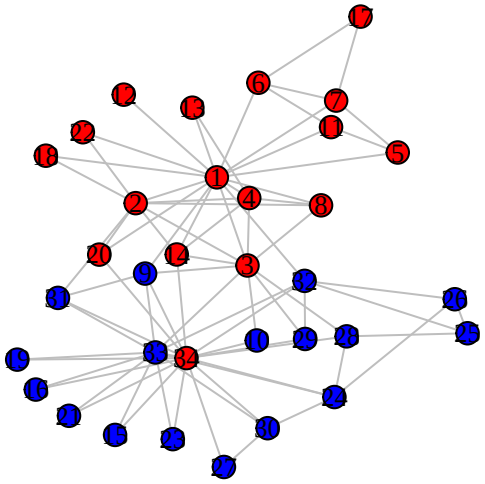
Veamos estos algoritmos sobre una red aleatoria:



Veamos ahora estos algoritmos sobre una red conocida y usada regularmente para testear algoritmos de detección de comunidades:



Veamos ahora cómo se dividió en realidad el club de Karate estudiado por Zachary:



¿Cuándo estamos realmente frente a una comunidad en una red?

En términos técnicos, la detección de comunidades en redes consiste en partir la red entre sub-estructuras o sub-grafos para comprender sus roles/funciones en la estructura de la red que, de otro modo, permanecen latentes. Existe una gran diversidad de estructuras comunitarias en diferentes dominios y su importancia en la comprensión de sistemas complejos varía junto con esas diferencias. En igraph el listado es grande: *link*

Fundamentos Teóricos y Desafíos:

En un comienzo no se estudiaban mucho las estructuras de clusters a nivel teórico porque se trabajada con redes aleatorias (donde son poco comunes) o regulares (donde son ad-hoc). Eso era insatisfactorio desde la perspectiva de la sociología.

Los avances informáticos permitieron manejar redes de gran tamaño y eso empezó a cambiar el panorama. Hoy, la detección de comunidades es un área activa de investigación en la forma de abordar la teoría de redes puesto que los aprendizajes a partir de redes teóricas (tales como las redes regulares) con propiedades conocidas y fácilmente detectables no sirven mucho para caracterizar redes reales que son una mezcla de orden y desorden.

Las redes regulares emanaban naturalmente a partir del trabajo de Erdös y Rényi sobre grafos aleatorios, en el cual la probabilidad de una arista entre dos vértices es igual para todos los pares, resultando en una distribución homogénea de aristas. Sin embargo, las redes reales presentan grandes heterogeneidad y una distribución de grados más amplia de las generadas en grafos aleatorios. Esta diferencia que veremos nuevamente y en detalle en el módulo sobre formación de redes indica un alto nivel de orden y organización (lo contrario a aleatoridad).

Una parte de la historia sobre esa diferencia entre las propiedades de redes observadas y redes aleatorais está explicado por la existencia de estructuras comunitarias. Tal es el caso del club de karate de Zachary. Por ello,

identificar comunidades ayuda a clasificar vértices según su posición estructural dentro de la red y, por ende, a entender las dinámcias de dicha red. En el caso del club mencionado, en una primera aproximación, los algoritmos que vimos quedan al debe.

Anteriormente indicamos que las comunidades o clústeres en redes son grupos de vértices con propiedades similares (color por ejemplo), pero también podrían ser similares en otros aspectos (roles, por ejemplo). Se encuentran en una amplia variedad de sistemas, desde redes sociales hasta redes biológicas y de Internet y tienen aplicaciones prácticas en diversos campos.

En esta área, los primeros desarrollos vienen de la mano de la evolución del pensamiento sociológico (por su interés natural por estudiar grupos). En la historia de esta disciplina, uno de los primeros análisis sistemáticos desde un enfoque de redes apareció en 1955 en la American Sociological Review bajo el título A Method for the Analysis of the Structure of Complex Organizations. Allí Weiss y Jacobson usaron encuestas para medir actitudes e interacciones dentro de una gran organización. En paralelo, surgían ideas similares en antropología y psicología. Por la misma época, Stuart Rice estudiaba clusters en cuerpos colegiados. A su modo, cada uno de estos trabajos estaban dando forma a un método hoy habitual de clusterización jerarquizada (hierarchical clustering).

El aumento de la escala de los fenómenos (y de los datos sobre los fenómenos) dejaron obsoletos muchos de los métodos usados desde entonces hasta finales del siglo XX. Y una nueva corriente de estudios emergió para desarrollar algoritmos capaces de detectar comunidades a gran escala. Entre ellos, uno particularmente interesante es el de **Girvan y Newman** visto más arriba (edge_betweeness). Ellos propusieron el 2002 un algoritmo importante para la detección de comunidades, basado en la centralidad de intermediación. Este trabajo ha influido en el desarrollo de nuevos métodos y en la aplicación de conceptos de diversas disciplinas (física, informática, dinámica no lineal, sociología y matemáticas discretas) en la detección de comunidades.

La detección de comunidades en grafos tiene como objetivo identificar módulos y su organización jerárquica, utilizando solo la topología del grafo. Este problema ha sido abordado en varias disciplinas y se han desarrollado métodos modernos, algoritmos para encontrar comunidades superpuestas y técnicas multiresolución y jerárquicas.

En su texto, Fortunato muestra ejemplos de comunidades en redes diversas. Tales como:

Redes Sociales: Las redes sociales son ejemplos paradigmáticos de grafos con comunidades. Ya vimos la red del club de karate de Zachary. Esta red es usada frecuentemente para probar algoritmos de detección de comunidades. Contiene 34 miembros de un club de karate en EE.UU. que fue observado por tres años. En ese período se produjo un conflicto entre los miembros que llevó a la división del club en dos grupos. Lo interesante es que, conocido el resultado del conflicto y conocida la red de interacción previa a éste, era posible deducir la red posterior al conflicto.

Redes de Colaboración Científica: Otro ejemplo es la red de colaboraciones de científicos del Santa Fe Institute (SFI), que al momento del artículo de Forunato contaba con 118 vértices entre científicos residentes y colaboradores. Aquí, las agrupaciones disciplinarias son visibles, con muchas cliques formadas por coautores de los mismos trabajos.

Redes de Delfines Nariz de Botella: Esta red es interesante porque muestra clusterización en otras especies. Se trata de 62 delfines en Doubtful Sound, Nueva Zelanda. Y el resultado mostraba dos grupos cohesivos después de la salida temporal de un delfín, con pocas interacciones entre los grupos.

Redes de Interacción Proteína-Proteína (PPI): La idea que las comunidades pueden ayudar a entender mejor la estructura de una red puede ser más persuasiva si la vemos a otro nivel de análisis. Por ejemplo, en biología y bioinformática, las redes de proteína-proteína son fundamentales para comprender los procesos celulares. Un ejemplo es la red PPI en células cancerígenas en ratas que interactúan mucho más de lo que lo hacen en células sanas.

World Wide Web: La web se puede representar como un grafo con páginas web como vértices y enlaces como aristas. Las comunidades aquí son grupos de páginas con similitudes temáticas. Es el estudio de esta red en particular la que llegó a Barabasi a proponer un mecanimos explicativo de las distribuciones Power Laws.

Redes de Asociación de Palabras: Basadas en normas de asociación libre de la Universidad de Florida del Sur. Esta red en particular está conformada del mismo modo que está la red del trabajo sobre recetas de cocina. Lo interesante de ese ejercicio es que al sacar a la palabra "bright" con la que se hizo el ejercicio de construir la red se pueden detectar comunidades semánticas.

El desafío computacional de la detección de comunidades

Conseguir eficiencia en los algoritmos de detección de comunidades es un desafío computacional importante y suelen salir a colación referencias al problema de si **NP=P**.

La *complejidad computacional* de un algoritmo se refiere a la estimación de los recursos necesarios para realizar una tarea, incluyendo tanto los pasos de cálculo necesarios como las unidades de memoria que deben asignarse simultáneamente.

La complejidad computacional se denota con la notación $O(n^a, m^b)$ donde a y b son los exponentes que indican cómo crece el tiempo de cómputo en función del número de vértices n y aristas m. Idealmente, se desean valores bajos para estos exponentes para minimizar la demanda computacional. Por ejemplo, grafos de la web con millones de vértices y miles de millones de aristas no pueden ser procesados por algoritmos cuyo tiempo de ejecución crece más rápido que $O(n)$ o $O(m)$.

En términos de complejidad, los algoritmos con complejidad polinómica pertenecen a la clase **P**. Sin embargo, hay problemas importantes de decisión y optimización para los cuales no se conocen algoritmos polinómicos. Estos problemas pueden requerir una búsqueda exhaustiva con un tiempo de crecimiento más rápido que cualquier función polinómica del tamaño del sistema, como el crecimiento exponencial.

Los problemas cuyas soluciones pueden verificarse en un tiempo polinómico pertenecen a la clase **NP** (tiempo polinómico no determinista), que incluye a **P**. Un problema es **NP-hard** si una solución para él se puede traducir en una solución para cualquier problema **NP**. La pregunta de si **NP** es igual a **P** sigue siendo el problema abierto más importante en ciencias de la computación teórica.

Muchos algoritmos de agrupamiento o problemas relacionados son **NP-hard**. En estos casos, no tiene sentido utilizar algoritmos exactos que solo podrían aplicarse a sistemas muy pequeños. Incluso si un algoritmo tiene complejidad polinómica, puede ser demasiado lento para sistemas grandes. Por ello, se utilizan comúnmente algoritmos de aproximación que no ofrecen una solución exacta, sino aproximada, con la ventaja de tener una complejidad menor. Estos algoritmos suelen ser no deterministas, ofreciendo diferentes soluciones para el mismo problema bajo diferentes condiciones iniciales o parámetros.

Conceptos Básicos En el agrupamiento de grafos, el primer desafío es definir cuantitativamente qué es una comunidad. Para ello no existe una definición universalmente aceptada, y esta suele depender del sistema específico en estudio. Intuitivamente, como ya vimos, una comunidad debería tener más aristas "internas" que aristas conectando sus vértices con el resto del grafo. Saber eso es necesario, pero no es suficiente.

En la práctica, las comunidades suelen definirse algorítmicamente, es decir, son el producto final de un algoritmo sin existir una definición a priori precisa.

Considérese un subgrafo C de un grafo G, con $|C| = n_c$ y $|G| = n$ vértices, respectivamente. Definimos el grado interno k^v_{int} y externo k^v_{ext} de un vértice $v \in C$ como el número de aristas que conectan v con otros vértices de C o con el resto del grafo, respectivamente. Si $k^v_{ext} = 0$, el vértice solo tiene vecinos dentro de C; si $k^v_{int} = 0$, está desconectado de C. El grado interno total de C, k^C_{int}, es la suma de los grados internos de sus vértices, y el grado externo total, k^C_{ext}, es la suma de los grados externos. Por definición, $k^C = k^C_{int} + k^C_{ext}$.

Definimos la densidad intra-cluster $\delta_{int}(C)$ del subgrafo C como la proporción entre el número de aristas internas de C y el número máximo posible de aristas internas, es decir,

$\delta_{int}(C) = \frac{\#\text{ aristas internas de } C}{n_c(n_c-1)/2}$

De manera similar, la densidad inter-cluster $\delta_{ext}(C)$ es la proporción entre el número de aristas que van de C al resto del grafo y el número máximo posible de aristas inter-cluster, es decir,

$delta_{ext}(C) = \frac{\#\text{ aristas inter-cluster de } C}{n_c(n-n_c)}$

Para que C sea una comunidad, esperamos que $\delta_{int}(C)$ sea significativamente mayor que la densidad promedio de enlaces $\delta(G)$ de G, y que $\delta_{ext}(C)$ sea mucho menor que $\delta(G)$. Buscar el mejor equilibrio entre un alto $\delta_{int}(C)$ y un bajo $\delta_{ext}(C)$ es implícita o explícitamente el objetivo de la mayoría de los algoritmos de agrupamiento.

Definiciones Locales Las comunidades son partes del grafo con pocas conexiones con el resto del sistema, consideradas como entidades separadas con su propia autonomía. Las definiciones locales se centran en el subgrafo en estudio, incluyendo posiblemente su vecindario inmediato, pero ignorando el resto del grafo.

Criterios comunes para definir comunidades en una red:

- **Mutualidad Completa**: Subgrupos donde todos los miembros son "amigos" entre sí, correspondiendo a un clique.
- **Alcanzabilidad**: Definida por la existencia y longitud de caminos entre vértices.
- **Grado de Vértices**: Un vértice debe ser adyacente a un número mínimo de otros vértices en el subgrafo.
- **Cohesión Interna vs. Externa**: Comparación entre la cohesión interna y externa de un subgrafo.

Definiciones Alternativas:

- **n-Clan y n-Club**: Variaciones de n-cliques con restricciones adicionales.
- **k-Plex y k-Core**: Subgrafos donde cada vértice es adyacente a todos los otros vértices del subgrafo excepto a lo sumo k de ellos (k-Plex) o a al menos k otros vértices (k-Core).

Particiones

Conceptos Básicos Una aproximación diferente a la detección de comunidades es pensar en particiones. Esto es: en vez de ir nodo por nodo para ver con quién está más conectado, evaluar distintas agrupaciones de una red y evaluar su intra e inter densidad. Una partición es una división de un grafo en grupos, de manera que cada vértice pertenece a un solo grupo. En sistemas reales, los vértices pueden compartirse entre diferentes comunidades.

Es una buena idea... pero hay un problema. Como señala Fortunato: El número de particiones posibles en k grupos de un grafo con n vértices es el número de Stirling de segundo tipo $S(n,k)$. El número total de posibles particiones es el n-ésimo número de Bell $B_n = \sum_{k=0}^{n} S(n,k)$. En el límite de grandes n, B_n tiene la forma asintótica:

$$B_n \approx \frac{1}{\sqrt{n}}\Theta(n)U_{n+1/2}e^{\Theta(n)-n-1}$$

donde $\Theta(n) = e^{W(n)} = n/W(n)$, y $W(n)$ es la función Lambert W. Por lo tanto, B_n crece más rápido que exponencialmente con el tamaño del grafo n, haciendo imposible la enumeración y/o evaluación de todas las particiones de un grafo, a menos que consista de muy pocos vértices. Es decir... es un problema computacionalmente complejo.

Pese a ello, este método es uno de los más populares. Y el concepto de **modularidad** se ha popularizado como una de las formas más estándar de detectar comunidades.

Modularidad Se supone que los algoritmos confiables identifican buenas particiones. Pero, ¿qué es una buena agrupación? Jon Kleinberg estudió este tema y demostró un importante teorema de imposibilidad. Dado un conjunto S de puntos y una función de distancia d definida, positiva definida y simétrica, Kleinberg mostró que no existe una agrupación que satisfaga simultáneamente las siguientes tres propiedades: - (a) invarianza de escala: es decir si multiplicamos la escala por un escalar se obtiene la misma clusterización) - (b) riqueza: cualquier posible partición del conjunto de puntos dado se puede recuperar si se elige una función de distancia adecuada - (c) consistencia: cada una partición, cualquier modificación de la función de distancia

que no disminuya la distancia entre puntos de diferentes conglomerados y que no aumenta la distancia entre puntos del mismo conglomerado, produce el mismo agrupamiento.

En el caso de la agrupación de grafos, no se puede definir en general una función de distancia para un grafo que no sea completo. Para grafos completos ponderados, como matrices de correlación, a menudo es posible definir una función de distancia. En un grafo genérico, excepto por la propiedad de invarianza de escala, las otras dos propiedades están bien definidas.

Mientras algunos algoritmos son eficientes al identificar un número limitado de particiones significativas, idealmente una o unas pocas, otros, como las técnicas basadas en agrupamiento jerárquico, generan una gran cantidad de particiones. Sin embargo, la cantidad de particiones generadas no implica que todas sean igualmente válidas o útiles.

Dada esta variabilidad en la calidad de las particiones generadas, resulta útil y a veces necesario contar con un criterio cuantitativo para evaluar la bondad de una partición gráfica. De eso trata la **función de calidad**, Q, que asigna un valor numérico a cada partición del grafo. Las particiones se pueden clasificar según las puntuaciones otorgadas por esta función, donde aquellas con puntuaciones más altas se consideran "mejores". La partición con la puntuación más alta se considera, por definición, la mejor.

No obstante, es importante reconocer que la evaluación de qué partición es mejor que otra es subjetiva y depende del concepto específico de comunidad y de la función de calidad adoptada. Esto implica que no existe un criterio absoluto o universal para determinar la mejor partición, ya que la "bondad" de una partición puede variar según el enfoque y los objetivos del análisis.

Con todo, la modularidad se puede escribir como:

$$Q = \frac{1}{2m} \sum_{ij} \left(A_{ij} - P_{ij}\right) \delta(C_i, C_j)$$

donde la suma recorre todos los pares de vértices, A es la matriz de adyacencia, m el número total de aristas del grafo, P_{ij} representa el número esperado de aristas entre los vértices i y j en el modelo nulo, y la función δ da uno si los vértices i y j están en la misma comunidad ($C_i = C_j$), cero en caso contrario. La elección del modelo nulo es arbitraria y varias posibilidades existen. La modularidad se puede expresar tanto en términos de aristas intra-cluster como inter-cluster. Un subgrafo es un módulo si la contribución correspondiente a la modularidad en la suma es positiva. Grandes valores positivos de la modularidad indican buenas particiones.

Ejemplo Numérico de Modularidad

Para ilustrar la fórmula de la modularidad, consideremos un grafo simple con 4 vértices, divididos en dos comunidades y con un total de 3 aristas.

Matriz de Adyacencia

La matriz de adyacencia `A` para nuestro grafo es:

```
##      [,1] [,2] [,3] [,4]
## [1,]    0    1    0    0
## [2,]    1    0    1    0
## [3,]    0    1    0    1
## [4,]    0    0    1    0
```

En total hay 3 aristas.

Asumamos que en el Modelo Nulo (P_ij) cada par de vértices tiene la misma probabilidad de estar conectado, y esa probabilidad es de 0.5. Por lo tanto, P_ij = 0.5 para todos los pares de vértices.

Definamos la función $\delta(C_i, C_j)$. Esta función adquiere el valor 1 si los vértices i y j están en la misma comunidad y 0 en caso contrario. Supongamos que los vértices 1 y 2 están en una comunidad, y los vértices 3 y 4 en otra.

Con lo anterior podemos calcular la Modularidad (Q)

Para el par (1, 2): (1 - 0.5) × 1 = 0.5 Para el par (3, 4): (1 - 0.5) × 1 = 0.5

Obtenemos:

$$Q = \sum((A - 0.5) * \delta(C_i, C_j))/(2 * m)$$

$$Q = \frac{0.5 + 0.5}{6} = \frac{1}{6}$$

Este valor de Q indica la calidad de la partición en términos de modularidad. En este caso, un valor positivo de Q sugiere que la división en comunidades es razonablemente buena, aunque en la práctica, se debe comparar con otros valores de Q obtenidos para diferentes particiones del mismo grafo para determinar cuál es la mejor.

Métodos tradicionales de detección de comunidades

Métodos Tradicionales

Particionamiento de Grafos

El particionamiento de grafos, crucial en computación paralela, diseño de circuitos y algoritmos, implica dividir los vértices en g grupos de tamaño predefinido para minimizar el número de aristas entre los grupos, conocido como el tamaño del corte. Este es un problema NP-duro con varias soluciones subóptimas disponibles.

- **Tamaño del Corte (Cut Size)** El tamaño del corte representa el número de aristas entre clusters. La fórmula para calcularlo es:

$R = \frac{1}{4} s^T L s$

Donde L es la matriz laplaciana (la diferencia entre la matriz de grado D y la matriz de adyacencia A) y s^T la transpuesta del vector índice s.

El algoritmo de Kernighan-Lin, propuesto por Kernighan y Lin (1970), optimiza una función de beneficio Q, representando la diferencia entre el número de aristas dentro de los módulos y entre ellos. Es eficiente $(O(n^2 \log n))$ y depende de la configuración inicial.

- **Método de Bisección Espectral** Basado en las propiedades del espectro de la matriz Laplaciana, este método utiliza la siguiente fórmula:

$R = \sum a_i^2 \lambda_i$ Donde λ_i son los autovalores de la Laplaciana y a_i los coeficientes de la expansión del vector índice s.

Teorema de Max-Flow Min-Cut Propuesto por Ford y Fulkerson (1956), se utiliza para determinar cortes mínimos a partir de flujos máximos en algoritmos de agrupamiento.

Otras Medidas Relacionadas con el Tamaño del Corte incluyen la conductancia ($\Sigma(C)$), el corte de razón y el corte normalizado, que favorecen particiones en clusters de tamaño aproximadamente igual.

Conductancia: $\Sigma(C) = \frac{c(C, G \setminus C)}{\min(k_C, k_{G \setminus C})}$

Corte de Razón: $\Sigma_C(C) = \frac{c(C, G \setminus C)}{n_C n_{G \setminus C}}$

Corte Normalizado: $\Sigma_N(C) = \frac{c(C, G \setminus C)}{k_C}$

El particionamiento de grafos, aunque esencial en varias aplicaciones, no es ideal para la detección de comunidades, ya que generalmente requiere información previa sobre el número y tamaño de los grupos.

- **Agrupamiento jerarquico (Hierarchical clustering)** El agrupamiento jerárquico es útil cuando se desconoce la estructura comunitaria de un grafo, como el número de clústeres o la pertenencia de los vértices. Estos algoritmos revelan la estructura multinivel del grafo y son comunes en análisis de redes sociales, biología, ingeniería y marketing.

Punto de Partida El primer paso es definir una medida de similitud entre vértices, lo que resulta en una matriz de similitud $n \times n, X$. Se han propuesto variadas definiciones de similitud.

Hay dos aproximaciones:
Algoritmos Aglomerativos: Comienzan con vértices como clústeres separados y los fusionan iterativamente si su similitud es alta. Son algoritmos de abajo hacia arriba.
Algoritmos Divisivos: Parten de un clúster único y lo dividen iterativamente eliminando aristas entre vértices con baja similitud. Son algoritmos de arriba hacia abajo.

Representación y Condiciones de Parada Los dendrogramas ilustran el proceso. A veces se imponen condiciones de parada para seleccionar una partición que satisfaga un criterio específico, como un número dado de clústeres o la optimización de una función de calidad (por ejemplo, modularidad).

Ventajas y Desventajas La principal ventaja del hierarchical clustering es que no requiere conocimientos previos sobre el número y tamaño de los clústeres. Sin embargo, no proporciona un método para discriminar entre las múltiples particiones obtenidas y elegir la que mejor represente la estructura comunitaria del grafo. Los resultados dependen de la medida de similitud específica adoptada y la estructura jerárquica obtenida puede ser artificial.

- **Agrupamiento Particional** El agrupamiento particional es una clase popular de métodos para encontrar clústeres en un conjunto de puntos de datos. Aquí, el número de clústeres se preasigna, k. Los puntos se incrustan en un espacio métrico, y se define una medida de distancia (disimilitud) entre pares de puntos. El objetivo es separar los puntos en k clústeres para maximizar/minimizar una función de costo dada basada en distancias entre puntos y/o desde puntos a centroides. Algunas funciones de costo comunes son:

k-clustering Mínimo: La función de costo es el diámetro de un clúster, la mayor distancia entre dos puntos de un clúster. Se clasifican los puntos de manera que el mayor de los diámetros de los k clústeres sea lo más pequeño posible.

Suma de k-clustering: Similar al k-clustering mínimo, pero el diámetro se reemplaza por la distancia promedio entre todos los pares de puntos de un clúster.

k-center: Para cada clúster i se define un punto de referencia x_i, el centroide, y se calcula la máxima distancia d_i de cada punto del clúster al centroide. Los clústeres y centroides se eligen de forma que minimicen el mayor valor de d_i

k-mediana: Similar al k-center, pero la distancia máxima desde el centroide se reemplaza por la distancia promedio.

Ahora bien, la técnica particional más popular es el agrupamiento *k-medias*. Imaginemos que tenemos un conjunto de datos en un espacio bidimensional (2D) con una distribución aleatoria. Queremos agrupar estos datos en un número específico de clústeres, digamos 3. Seleccionamos al azar 3 puntos como los centroides iniciales de los clústeres. Cada punto del conjunto de datos se asigna al centroide más cercano, basándose en la distancia euclidiana. Como resultado, obtenemos 3 grupos iniciales. Después de la asignación, recalculamos el centroide de cada clúster como el promedio de todos los puntos asignados a ese clúster. Luego se repite el ejercicio hasta que los centroides no cambien significativamente entre iteraciones, lo que indica que hemos encontrado una solución estable. El agrupamiento particional tiene la limitación de que el número de clústeres debe especificarse al principio, y no es capaz de derivarlo.

- **Clustering Espectral**

El clustering espectral se basa en el uso de los vectores propios de matrices como la matriz de similitud S o derivadas de ella para particionar un conjunto de n objetos $x_1, x_2, \ldots, x_n$ con una función de similitud entre pares, la cual es simétrica y no negativa $S(x_i, x_j) = S(x_j, x_i) \geq 0$.

En el clustering espectral, la matriz Laplaciana es frecuentemente utilizada. Para un grafo con k componentes conectados, esta matriz tiene k valores propios iguales a cero.

El métodos de Clustering Espectral

1. **Clustering Espectral No Normalizado**: Utiliza la matriz Laplaciana no normalizada L. Se calculan los vectores propios de los k valores propios más pequeños y se construye una matriz $n \times k$. Los vértices del grafo se representan en un espacio Euclidiano k-dimensional y se agrupan en k clústeres.
2. **Clustering Espectral Normalizado**: Emplea la matriz Laplaciana normalizada, con variantes según Shi y Malik, utilizando L_{rw}, y Ng et al., utilizando L_{sym}. La normalización adicional se aplica en el método de Ng et al.

El clustering espectral está relacionado con la partición mínima de grafos y con caminatas aleatorias, donde se busca minimizar la probabilidad de transiciones entre clústeres.

- La elección del tipo de Laplaciana depende de las inhomogeneidades en los grados de los vértices del grafo.
- El clustering espectral normalizado puede ser más efectivo ya que optimiza tanto la densidad de aristas intraclúster como la densidad interclúster.
- El número de clústeres debe ser determinado de antemano; brechas significativas en los valores propios pueden sugerir el número adecuado de clústeres.
- La convergencia y la calidad del resultado pueden depender del tamaño de la brecha entre los valores propios $|\lambda_{k+1} - \lambda_k|$.