Análisis de redes

Jordi Casas Roma

PID_00229646

Universitat Oberta de Catalunya

Índice

Introducción 5

1. Introducción a la teoría de grafos 7
1.1. Tipos elementales de grafos 7
1.1.1. Grafos simétricos 7
1.1.2. Grafos dirigidos 8
1.1.3. Otros tipos elementales de grafos 8
1.2. Tipos básicos de nodos y aristas 9
1.3. Formas de representación de los grafos 10
1.4. Métodos para recorrer grafos 11
1.5. Propiedades básicas de los grafos 12

2. Las redes en el mundo real 14
2.1. Tipos básicos de redes 14
2.2. Propiedades en redes reales 16
2.2.1. La propiedad *small world* o 6 grados de separación ... 17
2.2.2. La ley de la potencia (*power law*) 17
2.2.3. La dispersión en las redes reales 18

3. Métricas y propiedades de las redes 20
3.1. Métricas de nodos 20
3.1.1. Basadas en centralidad 20
3.1.2. Basadas en propiedades espectrales 21
3.1.3. Basadas en otras propiedades 22
3.2. Métricas de aristas y arcos 24
3.3. Métricas de red 25
3.4. Complejidad en redes grandes 26

4. Detección de comunidades 28
4.1. Estrategias de detección de comunidades 29
4.1.1. Agrupamiento jerárquico 29
4.1.2. Agrupamiento espectral 30
4.1.3. Agrupamiento dinámico 30
4.2. Algoritmos 30
4.2.1. El algoritmo de Girvan-Newman 30
4.2.2. El algoritmo de propagación de etiquetas de Raghavan 33
4.2.3. El algoritmo *leading eigenvector* 34
4.2.4. Algoritmo *Infomap* 36
4.3. Evaluación de la modularidad 37

4.4. Selección del algoritmo 38

5. Visualización de redes 40
5.1. Estrategias de visualización 40
5.2. La problemática del orden 43

6. Apéndice A: Notación 46

Resumen 47

Bibliografía 48

Introducción

En los últimos años la representación de datos en formato de red ha experimentado un importante auge en todos los niveles. Este formato permite representar estructuras y realidades más complejas que los tradicionales datos relacionales, que utilizan el formato de tuplas. En un formato semiestructurado cada entidad puede presentar, al igual que los datos relacionales, una serie de atributos en formato numérico, nominal o categórico. Pero además, el formato de red permite representar de una manera más rica las relaciones que puedan existir entre las distintas entidades que forman el conjunto de datos. Un claro ejemplo de esta situación lo presentan las redes sociales.

En la actualidad las redes sociales se han convertido en un fenómeno muy popular, que hace que millones de usuarios de todo el mundo estén presentes en una o varias redes sociales. Independientemente de su temática u objetivo, las redes sociales presentan una gran cantidad de información muy interesante para estudios en distintos ámbitos (psicología, ciencias sociales, etc). En este sentido, la explotación de estos datos resulta de gran interés para científicos y empresas de todo el mundo.

En la literatura científica los términos *red* y *grafo* se utilizan indistintamente. Generalmente, podemos encontrar referencias a redes o a grafos indistintamente, y sin diferencias importantes en su significado. Aun así, algunos autores apuntan a una sutil diferencia entre ambas terminologías. Por ejemplo, A. Barabási (Barabási, 2015) apunta que la terminología de red (*network*), nodo (*node*) y enlace (*link*) se utilizan a menudo para referirse a sistemas reales. Por ejemplo, internet es una red de páginas conectadas mediante enlaces URL; una sociedad es una red de individuos conectados a través de relaciones familiares, profesionales, sentimentales, etc. Por otro lado, la terminología de grafo (*graph*), vértice (*vertex*) y arista (*edge*) se utilizan generalmente para referirnos a las representaciones abstractas de las redes. La teoría de grafos es un campo de las matemáticas, dedicado desde hace mucho tiempo (mucho antes de que aparecieran las redes sociales actuales) al estudio de las propiedades de los grafos, entendiendo los grafos como un conjunto de entidades y las relaciones entre dichas entidades. Entre los problemas clásicos de la teoría de grafos podemos hallar el problema de los puentes de Königsberg*, el problema de los cuatro colores** o el problema del viajante***, entre muchos otros.

*https://es.wikipedia.org/wiki/Problema_de_los_puentes_de_Königsberg
**https://es.wikipedia.org/wiki/Teorema_de_los_cuatro_colores
***https://es.wikipedia.org/wiki/Problema_del_viajante

En este texto utilizaremos los términos *red* y *grafo* indistintamente, entendiendo que el contexto nos permitirá en cada caso discernir si se trata de una red real o de su representación en formato de grafo. De la misma manera, usaremos indistintamente los términos *nodos* o *vértices* para referirnos a las entida-

des de las redes o grafos, y los términos *enlaces*, *aristas* o *arcos* para referirnos a las relaciones entre las entidades de las redes o grafos.

En el apartado 1 se presentan algunos conceptos básicos de la teoría de grafos. La teoría de grafos es un campo complejo y su estudio detallado supera, con creces, los objetivos de este texto. Por lo tanto, presentaremos algunos conceptos básicos necesarios para poder desarrollarlos de manera correcta en el análisis de redes. A continuación, el apartado 2 nos introducirá en las redes y sus principales características. Más allá de la teoría de grafos, es interesante observar algunas propiedades muy habituales en las redes reales. En este apartado veremos algunas de sus principales propiedades, que serán un punto de gran interés para el análisis de redes.

En el apartado 3 se introducirán algunas de las métricas más habituales para analizar la estructura de las redes. Cada una de estas métricas permite cuantificar alguna propiedad de la red, facilitando la comparación entre redes y la comprensión de su estructura interna. Seguidamente, veremos los principios y algoritmos básicos para la detección de comunidades, que representa uno de los campos más activos en el análisis de redes. El apartado 5 nos presentará las nociones básicas para la representación de redes en un espacio bidimensional.

1. Introducción a la teoría de grafos

En este apartado introduciremos los conceptos básicos de la teoría de grafos. Los grafos son la forma más natural de representación de las redes reales, y en este sentido necesitamos introducir los conceptos básicos para poder representar y analizar las redes reales. Es importante destacar que queda fuera del alcance de este texto un estudio en profundidad de la teoría de grafos, pero el lector interesado puede encontrar más información en la bibliografía al respecto.

Lectura complementaria

Borges, J.; Clarisó, R.; Masià, R.; Pujol, J.; Rifà, J.; Vancells, Villanueva, M. (2007). *Grafos y complejidad*. Barcelona: Fundació per a la Universitat Oberta de Catalunya.

1.1. Tipos elementales de grafos

Existen una cantidad notable de tipos de grafos, pero algunos de ellos no son demasiado útiles para representar redes reales. A continuación veremos los tipos más elementales de grafos que nos serán de interés para poder representar algunos de los tipos más habituales de redes reales.

1.1.1. Grafos simétricos

Un **grafo simétrico** o no-dirigido es una pareja de conjuntos $G = (V,E)$, donde $V = \{v_1,v_2...,v_n\}$ es el conjunto de **nodos** o **vértices** y $E = \{e_1,e_2,\ldots,e_m\}$ es un conjunto de **aristas** de la forma $e_i = \{v_i,v_j\}$ tal que $v_i,v_j \in V$ y $v_i \neq v_j$, de manera que cada par une a dos de los nodos de forma bidireccional. Es decir, la arista $\{v_i,v_j\}$ permite unir al nodo v_i con el nodo v_j y viceversa. Un **lazo** o bucle es una arista (o arco) que relaciona al mismo nodo $\{v_i,v_i\}$; es decir, una arista donde el nodo inicial y final coinciden. Se llama **orden** de G a su número de nodos, $|V|$, que por convenio es referenciado por la letra n. Asimismo, el número de aristas, $|E|$, es referenciado por la letra m y se le llama **tamaño** del grafo.

Cardinalidad

La cardinalidad de un conjunto es el número de elementos que posee. Se representa mediante el símbolo $|\cdot|$, donde $\cdot$ indica el conjunto.

Los nodos **adyacentes** o vecinos, denotados como $\Gamma(v_i)$, se definen como el conjunto de nodos unidos a v_i a través de una arista. En este caso se define el **grado** de un nodo como el número de nodos adyacentes, es decir, $|\Gamma(v_i)|$.

Un **camino** es una secuencia ordenada de aristas donde el nodo final de una arista es el nodo inicial de la siguiente. La **distancia** entre dos nodos, $d(v_i,v_j)$, se define como el número de aristas del camino más corto entre los nodos v_i y v_j. Formalmente se define la distancia entre dos nodos como $d(v_i,v_j) = min\{\ell(C)|C$ es un camino $v_i - v_j\}$. Un camino es simple si no repite aristas y es

elemental en caso de no repetir nodos. A un camino cerrado, es decir, que empieza y termina en el mismo nodo, se le llama **circuito**.

1.1.2. Grafos dirigidos

De forma similar, un **grafo dirigido** o asimétrico es una pareja de conjuntos $G = (V,A)$, donde $V = \{v_1,v_2...,v_n\}$ es el conjunto de **nodos** o **vértices** y $A = \{a_1,a_2,\ldots,a_m\}$ es un conjunto de **arcos** de la forma $a_i = (v_i,v_j)$ tal que $v_i,v_j \in V$ y $v_i \neq v_j$, de manera que cada par une a dos de los nodos de forma unidireccional. En este caso el arco (v_i,v_j) permite unir al nodo v_i con el nodo v_j, pero no en el sentido contrario. De modo similar a los grafos simétricos, el número de arcos, $|A|$, es referenciado por la letra m y se le llama **tamaño** del grafo.

En un grafo dirigido G se define a los **sucesores** de un nodo v_i, $\Gamma(v_i)$, como el conjunto de nodos a los cuales se puede llegar usando un arco desde v_i. Se define el **grado exterior** de un nodo como el número de sucesores $|\Gamma(v_i)|$. De modo similar, se puede definir a los **antecesores** de un nodo v_i, $\Gamma^{-1}(v_i)$, como el conjunto de nodos desde los cuales es posible llegar a v_i usando un arco. Se define el **grado interior** de un nodo como el número de antecesores, es decir, $|\Gamma^{-1}(v_i)|$.

Las definiciones de camino, distancia y circuito introducidas para los grafos simétricos son similares para los grafos dirigidos, exceptuando que en el caso de los grafos dirigidos hay que considerar que un arco solo permite la unión de dos nodos en un único sentido.

1.1.3. Otros tipos elementales de grafos

En los subapartados anteriores ya se han definido dos tipos básicos muy importantes de grafos: los **grafos simétricos** y los **grafos dirigidos**. Sin embargo, existe una cantidad considerable de otros tipos de grafos. En este subapartado solo destacaremos algunos por su especial relevancia en los temas tratados en este texto:

- Se llama **grafo subyacente** al grafo simétrico que se obtiene eliminando la orientación de los arcos de un grafo dirigido.

- Un grafo $G_p = (V_p,E_p)$ es un **subgrafo** o **grafo parcial** de $G = (V,E) \leftrightarrow E_p \subseteq E$ y $V_p \subseteq V$.

- Se llama **multigrafo** a un grafo que contiene aristas (o arcos) repetidos. Si además se permite que un nodo esté relacionado consigo mismo mediante **bucles** o **lazos**, es decir, mediante una arista $\{v_i,v_i\}$ o arco (v_i,v_i), entonces se le llama **pseudografo**.

Si nos fijamos en la estructura de una red de carreteras, nos daremos cuenta de que en algunos casos existen dos o más carreteras que unen dos mismos pueblos o ciudades. Para representar esta red mediante un grafo emplearemos un multigrafo, dado que deberemos permitir que existan aristas repetidas entre dos vértices.

- Un grafo G cuyas aristas o arcos tienen asociados valores reales, llamados **pesos**, se denomina **grafo ponderado**. Un **grafo ponderado** se representa mediante $G = (V,E,\omega)$, donde G es un grafo y ω es una función $\omega : E \to \mathbb{R}$ que asigna pesos a las aristas o arcos del grafo.

- Un **grafo completo** es un grafo con el mayor número posible de aristas, sin lazos y sin repeticiones. El grafo simétrico completo de t nodos se denota como K_t. Se puede ver que dado $K_t = (V,E)$, entonces $|V| = t$ y $|E| = \binom{t}{2}$, es decir, $\frac{t(t-1)}{2}$, todas las aristas posibles.

- Un **grafo conexo** es un grafo simétrico en donde para cada par de nodos existe un camino que los une. A cada subgrafo maximal conexo en un grafo simétrico G se le llama **componente** de G. Se puede expresar un grafo G como la unión de los distintos componentes que lo forman, es decir, $G = G_1 \cup \ldots \cup G_k$ donde cada $G_i | i \in (1,\ldots,k)$ es una componente de G.

Propiedad maximal

La propiedad maximal se refiere a que el subgrafo es máximo, es decir, no podemos añadirle más vértices sin que pierda su propiedad de subgrafo conexo.

- Un grafo $G = (V,E)$ se llama **bipartito** si existe una partición del conjunto de nodos $V = V_1 \cup V_2$ con $V_1 \cap V_2 = \emptyset$ de tal modo que las aristas conectan los nodos de V_1 con los nodos de V_2. Es decir, $\{v_i,v_j\} \in E$ implica que $v_i \in V_1$ y $v_j \in V_2$ o viceversa. Generalizando se obtiene el concepto de los grafos k-partitos. En este caso se tiene una partición $V_1,\ldots,V_k$ del conjunto de nodos, de tal forma que las aristas conectan nodos que pertenecen a distintas particiones.

Ejemplo de grafo bipartito

Un ejemplo de grafo bipartito puede ser la red de representación de usuarios y compras de un establecimiento. En este caso, tenemos dos conjuntos de nodos: V_U representa los usuarios o clientes y V_P, los productos del establecimiento. La arista $\{v_u,v_p\}$ une los nodos $v_u \in V_U$ y $v_p \in V_P$ e indica que el usuario v_u ha adquirido el producto v_p. Podemos ver que se cumple la condición $V_U \cap V_P = \emptyset$.

1.2. Tipos básicos de nodos y aristas

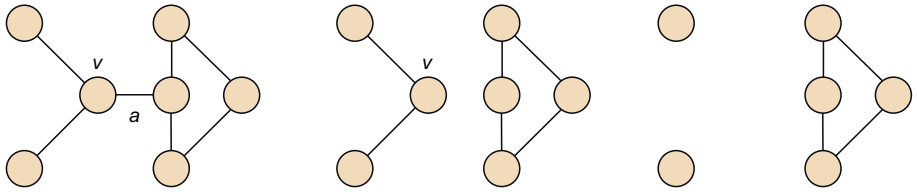
Algunos nodos y aristas pueden presentar propiedades importantes dentro de un grafo. A continuación se enumeran algunas de las más relevantes:

- Un **puente** en G es toda arista e_j tal que $G = (V,E - \{e_j\})$ tiene más componentes que G. En la figura 1a, la arista a es un puente. La figura 1b muestra el grafo después de eliminar el puente.

- Una **articulación** en G es un nodo v_i tal que $G = (V - \{v_i\},E - E_i)$ tiene más componentes que G, siendo $E_i \subseteq E$ el conjunto de aristas que tienen v_i como nodo terminal. En la figura 1, el nodo v es una articulación. La figura 1c muestra el grafo después de eliminar la articulación.

- Un **hub** en G es un nodo v_i tal que su grado es mucho mayor que el grado de la mayoría de los nodos de G. Un claro ejemplo de *hub* en las redes sociales son las celebridades, que tienen un número de amigos o seguidores muy superior a la media de los demás usuarios.

Figura 1. Ejemplo de puente y articulación en un grafo



(a) G **(b)** $G - \{a\}$ **(c)** $G - \{v\}$

1.3. Formas de representación de los grafos

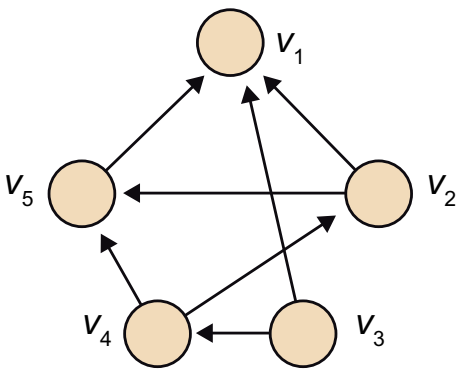
Una forma muy habitual de representar un grafo es mediante la matriz de adyacencia. La **matriz de adyacencia** es una matriz cuadrada que se utiliza para representar relaciones binarias. Dado un grafo G su matriz de adyacencia $A = (a_{ij})$ es cuadrada, de orden n y viene dada por:

$$a_{ij} = \begin{cases} 1 & \text{si } \exists(v_i, v_j) \text{ o } \{v_i, v_j\} \\ 0 & \text{caso contrario} \end{cases} \quad (1)$$

La matriz de adyacencia del grafo G, representado en la figura 2, es:

$$A(G) = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \end{pmatrix} \quad (2)$$

Figura 2. Ejemplo de grafo G para el cálculo de la matriz de adyacencia



Otra forma de representación matricial de un grafo G sin lazos es la matriz de incidencia. La **matriz de incidencia** $B = (b_{ij})$ de un grafo $G = (V,E)$ contiene n filas y m columnas, y viene dada por:

$$b_{ij} = \begin{cases} 1 & \text{si } v_i \text{ es el vértice inicial de } a_j \\ -1 & \text{si } v_i \text{ es el vértice final de } a_j \\ 0 & \text{si } a_j \text{ no afecta a } v_i \end{cases} \quad (3)$$

Si el grafo es simétrico, es habitual cambiar todos los -1 por 1.

La matriz de adyacencia presenta la ventaja de ser cuadrada y binaria, lo cual permite realizar operaciones de una forma sencilla y rápida. En cambio, la matriz de incidencia no presenta esta ventaja, por lo que no suele ser tan utilizada en las operaciones sobre grafos. Su uso se reduce a algunas operaciones concretas que recorren todas las aristas del grafo.

Una tercera forma interesante de poder representar un grafo es mediante la **lista de adyacencia**. En esta estructura se asocia a cada vértice v_i una lista que contiene todos los vértices adyacentes $v_j|\{v_i,v_j\} \in E$. En el caso de un grafo dirigido, se asocia a cada vértice v_i una lista que contiene todos los vértices accesibles desde él, es decir, $v_j|(v_i,v_j) \in A$. La principal ventaja de este método de representación es el ahorro de espacio para almacenar la información.

Ejemplo de almacenamiento de grafos

Supongamos que queremos almacenar un grafo no dirigido de 100 vértices y 1.000 aristas (n = 100 y m = 1.000). Si utilizamos la matriz de adyacencia, necesitaremos una matriz cuadrada de $n \times n$, es decir, 10^4 posiciones. En el caso de emplear la matriz de incidencia, deberemos reservar $n \times m = 10^5$ posiciones. Finalmente, utilizando la lista de adyacencia requeriremos una posición para cada vértice del grafo y otras dos para cada arista (debido a que representamos la adyacencia entre ambos vértices), es decir, $n + 2m$ = 2.100.

1.4. Métodos para recorrer grafos

Para examinar la estructura de un grafo, es común tener que realizar un recorrido por todo este. Las dos estrategias más comunes para tal fin son:

- **BFS** (*Breadth-First Search*) o búsqueda en anchura prioritaria: Este método prioriza la búsqueda en paralelo de todas las alternativas posibles desde el nodo actual.

- **DFS** (*Depth-First Search*) o búsqueda en profundidad prioritaria: Este método prioriza la apertura de una única vía de exploración a partir del nodo actual.

Ejemplo de recorridos en un grafo

La figura 3 presenta un grafo de siete vértices. Los recorridos de este grafo, empezando en v_1, en BFS y DFS son:

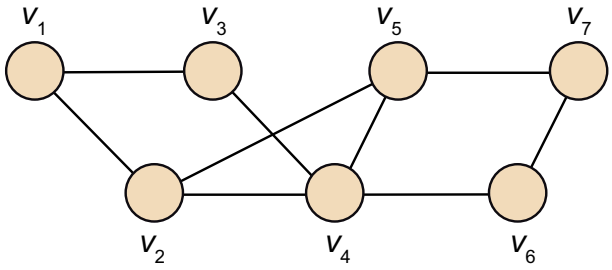
- BFS: 1,2,3,4,5,6,7. A partir del vértice v_1, visitamos v_2 y v_3 en el primer paso. A continuación, en el segundo paso, visitamos v_4 y v_5 desde v_2. También podemos visitar v_4 desde v_3, pero como ya hemos accedido a él, no lo consideramos. En el siguiente paso, visitamos v_6 desde v_4, y por último v_7 desde v_5.

- DFS: 1,2,4,5,7,6,3. A partir del vértice v_1 visitamos v_2, v_4, v_5, v_7 y v_6. En este punto, volvemos atrás hasta visitar el único vértice que nos queda pendiente, v_3.

Nota

En estos ejemplos hemos utilizado el criterio de que el algoritmo selecciona el vértice con identificador menor cuando explora los vecinos. Es decir, si v_1 está enlazado con v_2 y v_3, el algoritmo selecciona primero v_2. Este convenio solo pretende simplificar la comprensión del ejemplo.

Figura 3. Ejemplo de grafo G para el cálculo de la matriz de adyacencia



1.5. Propiedades básicas de los grafos

A continuación describiremos brevemente algunas de las propiedades más importantes de los grafos, que frecuentemente se usan para describir la estructura de un grafo. Ejemplificaremos cada una de ellas utilizando la figura 4:

- **Grado máximo**: grado máximo de todos los nodos de la red. En el ejemplo, el grado máximo es 5, que corresponde al grado del nodo v_6.

- **Grado mínimo**: grado mínimo de todos los nodos del grafo. En nuestro ejemplo, el grado mínimo es 1, que corresponde a los nodos v_3, v_7 y v_{12}.

- **Secuencia de grados** (*degree sequence*): Una secuencia de grados es una secuencia numérica de n posiciones en la que cada posición i indica el grado del nodo v_i. La secuencia de grados para el ejemplo dado es {3,2,1,3,3,5,1,2, 3,4,2,1}. Es decir, la secuencia de grados nos indica que el nodo v_1 tiene grado 3, el nodo v_2 tiene grado 2, ..., y el nodo v_{12} tiene grado 1.

- **Grado medio**: grado medio de todos los nodos del grafo. Se calcula como la suma del grado de todos los nodos dividido entre el número total de nodos. Por lo tanto, en el ejemplo el valor del grado medio es $\frac{3+2+1+3+3+5+1+2+3+4+2+1}{12} = 2{,}5$.

- **Histograma de grados** (*degree histogram*): Un histograma de grados es una distribución de la frecuencia del grado de los nodos en un grafo. El histograma de grados del ejemplo es {0,3,3,4,1,1}. El histograma indica que el grafo no tiene ningún nodo de grado 0, tiene tres nodos de grado 1, tres

nodos de grado 2, cuatro nodos de grado 3, un nodo de grado 4 y un nodo de grado 5. La figura 5 muestra la gráfica del histograma de grados del grafo de ejemplo.

Figura 4. Grafo de ejemplo

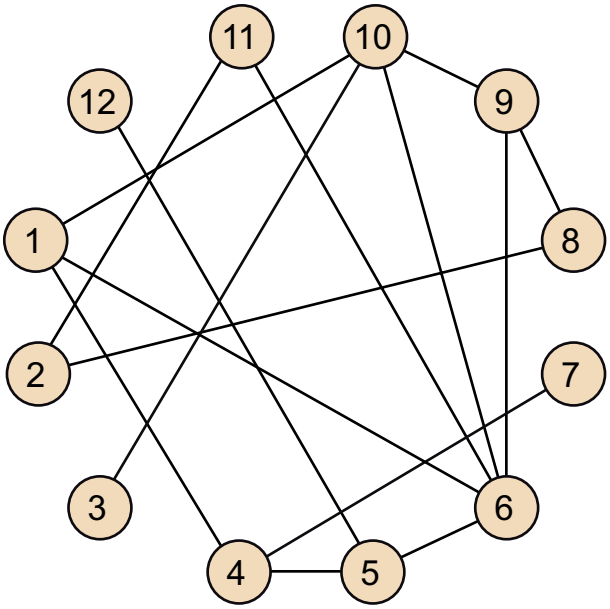
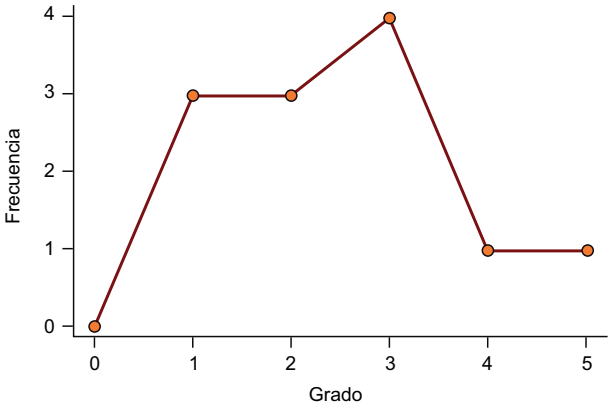


Figura 5. Histograma de grados



2. Las redes en el mundo real

En este apartado introduciremos algunas propiedades que se repiten con mucha frecuencia en las redes reales. En este sentido, es interesante estudiar estos fenómenos, que permiten caracterizar y clasificar las redes reales. Lógicamente, dependiendo de su índole, las redes reales pueden seguir todas, algunas o ninguna de las propiedades que en este apartado se mencionan. Aun así, cuando se realiza un análisis de redes es interesante ver si se cumplen estas propiedades.

2.1. Tipos básicos de redes

Como hemos visto, existen multitud de tipos de grafos que nos permitirán representar de manera concreta los distintos tipos de redes reales que existen. Aun así, algunos de ellos ocurren con una muy baja frecuencia en las redes reales. Por ejemplo, en teoría de grafos se estudian los grafos completos, pero difícilmente podremos hallar una red real en donde todos los nodos estén conectados entre sí. Por lo tanto, rara vez encontraremos redes reales que respondan a este tipo.

Cuando analizamos redes reales, distinguimos algunas características de la red que nos permiten identificar el tipo de red, y el tipo de grafo subyacente que deberemos emplear para representarla. Las características más básicas y elementales son las siguientes:

- **El tipo de relación**: Según el tipo de relación que se establezca entre las entidades de la red, podemos hablar de relaciones dirigidas o simétricas. Es decir, una relación dirigida $a \rightarrow b$ indica que existe una relación de la entidad a hacia la entidad b, pero no en la dirección contraria. Dependiendo del tipo de red que estamos analizando, puede corresponder a distintas realidades. Por ejemplo, este tipo de relación se presenta en redes sociales como Twitter, en donde un usuario establece la relación de "seguir" a otro usuario, pero el segundo no tiene por qué "seguir" al primero. En una red de calles o carreteras se puede representar una calle de dirección única, en la que solo puedes circular en un único sentido. Este tipo de redes se representan mediante un grafo dirigido, tal y como hemos visto, y las relaciones se llaman *arcos*. Por otro lado, podemos tener relaciones simétricas, es decir, relaciones $a \leftrightarrow b$, en donde si la entidad a está relacionada con la entidad b, entonces la relación se produce en el sentido contrario de forma implícita. Por ejemplo, este tipo de relación se utiliza para representar el

concepto de amistad en redes sociales como Facebook. En estas redes, si un usuario tiene una relación de "amistad" con otro usuario, implica que el segundo usuario tiene la misma relación con el primero. Este tipo de redes se representan mediante un grafo simétrico, y sus relaciones se llaman *aristas*.

- **Existencia de pesos o etiquetas en las relaciones**: Algunas relaciones utilizan valores numéricos, nominales o categóricos para especificar su significado. Por ejemplo, en una red de carreteras se puede utilizar un atributo numérico para indicar la distancia entre los dos puntos terminales. En una red social se puede utilizar un atributo categórico para determinar el tipo de relación que existe entre dos individuos, indicando, por ejemplo, si tienen una relación profesional, familiar, sentimental, etc. Cuando necesitamos representar atributos en las relaciones, ya sean numéricos, nominales o categóricos, se requiere el uso de grafos etiquetados (*labeled graph*) o ponderados (*weighted graph*) en el caso de atributos numéricos. En caso contrario, se puede hablar simplemente de grafos o se puede especificar que se trata de grafos no etiquetados.

- **Grupos de entidades:** Las redes bipartitas son un tipo de red en el cual existen dos tipos de entidades $A = \{a_1, \ldots, a_n\}$ y $B = \{b_1, \ldots, b_m\}$ y las relaciones siempre se producen entre entidades del tipo A y entidades del tipo B. Es decir, las relaciones pueden ser $a_i \rightarrow b_j$ en caso de ser unidireccionales (pero no $a_i \rightarrow a_j$ o $b_i \rightarrow b_j$), o $a_i \leftrightarrow b_j$ en caso de ser bidireccionales (pero no $a_i \leftrightarrow a_j$ o $b_i \leftrightarrow b_j$). Las redes sociales en donde se relacionan usuarios y productos, como por ejemplo la red de Amazon, son ejemplos típicos de redes bipartitas en las que los usuarios valoran o compran ciertos productos, estableciendo relaciones entre usuarios y productos. Para representar estas redes emplearemos los grafos bipartitos. En caso de tener más de dos conjuntos de entidades, hablaremos de grafos tripartitos o, en general, de grafos k-partitos.

A partir de estos tipos más elementales de redes, debemos decidir la información que queremos incorporar a nuestro modelo o grafo, y cómo queremos representar dicha información. Veamos un par de casos, solo a modo de ejemplos básicos.

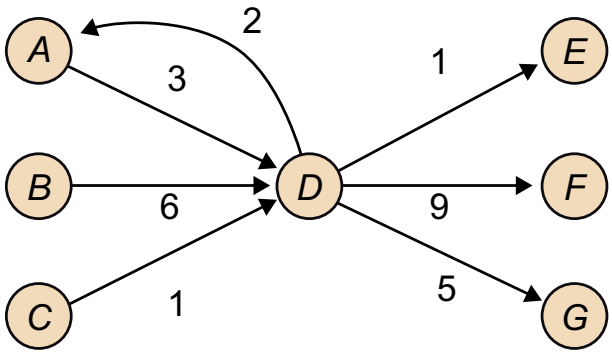
Ejemplo de red dirigida y ponderada

En el primer ejemplo vamos a suponer que deseamos modelar las relaciones entre los usuarios de Twitter, pero no nos interesa ver quién sigue a quién, sino que deseamos ver quién hace retuits de quién. Es decir, queremos considerar la posibilidad de evaluar la importancia de los usuarios, no en función de su número de seguidores, sino a partir de la cantidad de retuits que los demás usuarios han hecho de sus tuits. Para representar la información que nos interesa podemos utilizar un grafo dirigido y ponderado, en donde creamos un arco entre los nodos a y b si el usuario a ha hecho uno o más retuits de cualquier tuit del usuario b. Lógicamente, cada arco contendrá un valor numérico, que indicará el número exacto de retuits que el usuario a ha hecho del usuario b. La figura 6 muestra una posible configuración para este ejemplo. Podemos ver que los usuarios A, B y C han realizado, respectivamente, 3, 6 y 1 retuits del usuario D. Por su parte, el usuario D ha realizado retuits del mismo usuario A y de los usuarios E, F y G.

Flocker

Este tipo de análisis puede realizarse con la herramienta *flocker* del colectivo *outliers*. http://flocker.outliers.es/

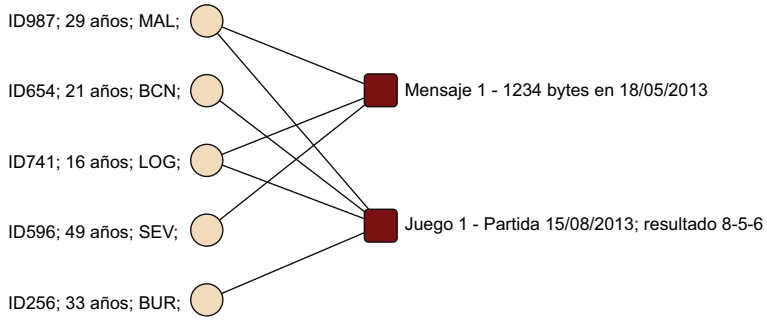
Figura 6. Ejemplo de grafo dirigido y ponderado



Ejemplo de red bipartita

En el segundo ejemplo vamos a suponer que deseamos modelar la relación entre los usuarios de un juego en línea. En este caso supondremos que los jugadores pueden participar juntos en uno o más juegos y que, además, pueden enviarse mensajes entre ellos (para, por ejemplo, quedar para jugar una nueva partida). Nos interesa modelar la información de quién juega con quién y quién se envía mensajes con quién. Además, queremos almacenar la información de la edad y ciudad de los jugadores, así como la fecha de la partida. Una buena forma de representar esta red podría ser utilizando un grafo bipartito, en donde tenemos los jugadores como un grupo de nodos y los mensajes y partidas como otro grupo de nodos. Entonces, las relaciones se producen siempre entre los nodos del grupo de los jugadores y los nodos del grupo de los mensajes y partidas. Para almacenar la información de los jugadores y las partidas, podemos usar simplemente atributos de los nodos, ya sean de un tipo o de otro. En este caso, las relaciones serán simétricas y no etiquetadas. La figura 7 ejemplifica el modelo descrito. En ella vemos cinco jugadores, cuatro de los cuales han intervenido en la misma partida y tres de los cuales se han enviado mensajes entre ellos.

Figura 7. Ejemplo de grafo bipartito



En este texto se entiende por **red o grafo simple** (o *simple graph*) aquella red que no contiene atributos en los nodos ni etiquetas o pesos en las aristas o arcos. Es decir, un grafo simétrico o dirigido, sin atributos y sin etiquetas o pesos. Por el contrario, se entiende por **red o grafo complejo** (o *rich graph*) aquella red que contiene atributos en los nodos y etiquetas o pesos en las aristas o arcos, lo que permite distintas formas de relación entre los nodos de la red.

2.2. Propiedades en redes reales

Dentro de las investigaciones recientes en redes, las centradas en redes sociales ocupan una parte muy importante de estas. En este sentido, cabe destacar

algunas características propias de las redes sociales que no tienen por qué darse en otro tipo de redes. Entre las principales, cabe destacar:

2.2.1. La propiedad *small world* o 6 grados de separación

El fenómeno conocido como *small world* o 6 grados de separación es la hipótesis de que la cadena de conocidos necesarios para conectar a dos personas arbitrarias en cualquier parte del mundo es generalmente corta. El concepto surge del famoso experimento **six degrees of separation*** que realizó el psicólogo Stanley Milgram en el año 1967. Aplicado a las redes en general, y las redes sociales en particular, implica que la distancia entre dos nodos o individuos cualquiera de la red suele ser pequeña.

*http://en.wikipedia.org/wiki/Six_degrees_of_separation

En 2012, Backstrom y otros realizaron un estudio similar utilizando toda la red social de Facebook, que contaba con $\approx$ 721 millones de usuarios y $\approx$ 69 miles de millones de relaciones de amistad. La distancia media observada fue de 4,74, correspondiendo a 3,74 intermediarios o "grados de separación" entre usuarios, lo que demostró que el mundo *online* es aun menor de lo esperado. Un estudio reciente reduce esta cifra a 3,57**.

Referencia bibliográfica

Backstrom, L.; Boldi, P.; Rosa, M.; Ugander, J.; Vigna, S. (2012). *Four Degrees of Separation*. Disponible en http://arxiv.org/abs/1111.4570

**https://research.facebook.com/blog/three-and-a-half-degrees-of-separation/

2.2.2. La ley de la potencia (*power law*)

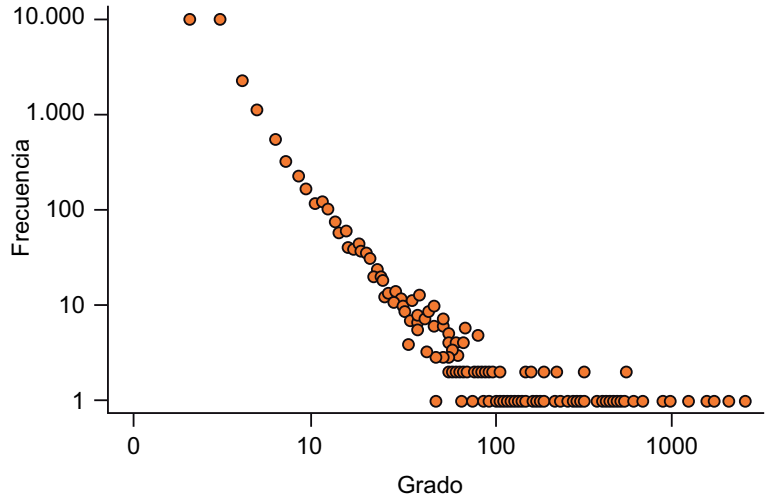
La ley de la potencia*** aplicada a las redes sociales implica que la distribución de los nodos en función de su grado sigue la ley de la potencia.

***http://en.wikipedia.org/wiki/Power_law

La figura 8 muestra el histograma de grados de una red social real, concretamente de la red *Amazon product co-purchasing network, June 01 2003*****. En el eje horizontal podemos ver el grado de los nodos (en escala logarítmica) y en el eje vertical podemos ver su frecuencia de aparición (en escala logarítmica). En la figura podemos ver que existe una gran cantidad de nodos (usuarios) con grados relativamente bajos, mientras que solo existen unos pocos usuarios con un grado muy superior a los demás. Estos usuarios o nodos son llamados *hubs*, ya que concentran una gran cantidad de relaciones a su alrededor. En las redes sociales, muchas veces corresponden a celebridades o personajes de gran interés público. Es interesante notar que cuando mostramos el histograma de grados en una escala logarítmica podemos ver cómo su distribución sigue, más o menos, una línea recta descendiente. Cuando esto ocurre, estamos hablando de una red que cumple con la ley de la potencia. Estas redes también son llamadas redes de escala libre (*scale-free networks*).

****Obtenida de https://snap.stanford.edu/data/index.html

Figura 8. Histograma de grados de una red social real (en escala log-log)



2.2.3. La dispersión en las redes reales

Actualmente encontramos redes de todos los tamaños e índoles, pero la gran mayoría de las redes reales son dispersas (*sparse networks*). Con esta propiedad queremos enfatizar que el número de aristas o arcos de una red real es muy inferior al número máximo de aristas o arcos que podría tener como límite, es decir, si fuera una red o grafo completo.

Para determinar el número máximo de aristas que puede tener una red de n nodos, solo debemos considerar que cada nodo de la red tendrá $n-1$ aristas en caso de ser completa. Por lo tanto, el número máximo de aristas m_{max} sería:

$$m_{max} = \binom{n}{2} = \frac{n!}{2!(n-2)!} = \frac{n \cdot (n-1)}{2} \quad (4)$$

Por ejemplo, supongamos el caso de una red social con tan solo 10.000 usuarios. Comparando con las cifras que hemos visto de Facebook ($\approx$ 721 millones de usuarios) parece más bien de tamaño pequeño/medio. Para que la red tenga el número máximo de aristas, sería necesario que cada usuario tuviera $n-1$ relaciones o "amigos", donde n es el número de usuarios de la red. Es decir, cada usuario debería tener 9.999 amigos; una cifra que parece muy superior a las cifras que normalmente conocemos. Por eso, las redes reales contienen muy pocas relaciones en comparación al número total de relaciones que pueden albergar, con lo que cumplen la propiedad de dispersión.

Para representar las redes que presentan la propiedad de dispersión es aconsejable utilizar formatos como la lista de adyacencia, que optimiza el espacio en función del número de nodos y aristas existentes. Por el contrario, la representación basada en la matriz de adyacencia utiliza un espacio proporcional al número total de aristas o arcos que pueden existir en el grafo, es decir, de orden cuadrático en el número de nodos. Si intentamos representar una red

dispersa de gran tamaño utilizando la matriz de adyacencia, consumiremos una gran cantidad de memoria para poder representar la red, incluso puede llegar a ser imposible la representación de redes muy grandes de millones o cientos de millones de nodos.

3. Métricas y propiedades de las redes

En el análisis de redes se utilizan un conjunto de métricas para evaluar y cuantificar ciertos aspectos o propiedades de estas. Esta evaluación o cuantificación nos permite comparar distintas redes en términos numéricos. Según el aspecto o propiedad que se evalúe, distinguimos entre métricas asociadas a los nodos, métricas asociadas a las aristas o arcos, y métricas asociadas a toda la red. A continuación profundizaremos en cada uno de estos tipos de métricas.

3.1. Métricas de nodos

Las métricas de nodos son aquellas métricas que evalúan cada nodo de la red de manera independiente. Por lo tanto, se obtiene un valor distinto e independiente para cada nodo de la red. Dicho valor puede ser calculado para un único nodo o para cada uno de los nodos de la red, formando un vector de resultados de longitud n.

3.1.1. Basadas en centralidad

A continuación describiremos algunas de las métricas de nodos basadas en centralidad más conocidas y utilizadas en el análisis de redes.

Existen diferentes tipos de centralidad basada en el grado según si aplicamos este concepto a una red dirigida o simétrica. El concepto de **grado interior** (*in-degree*) y **grado exterior** (*out-degree*) se aplica a redes dirigidas y contabiliza el número de antecesores de un nodo (grado interior) y el número de sucesores (grado exterior). En el caso de las redes simétricas, ambos coinciden y simplemente hablamos de **grado** (*degree*) de un nodo y representa el número de nodos adyacentes.

A partir de estos conceptos básicos, se define la **centralidad basada en el grado** (V_{DC}, *degree centrality*) como la fracción de nodos conectados a un determinado nodo. Formalmente se define la centralidad basada en el grado del nodo v_i como:

$$V_{DC}(v_i) = \frac{deg(v_i)}{n-1} \tag{5}$$

Nota

$deg(v_i) = |\Gamma(v_i)|$

La **centralidad basada en la intermediación** (V_{BC}, *betweenness centrality*) de un nodo calcula la fracción de caminos más cortos que pasan a través de este nodo. Esta métrica mide la centralidad de un nodo basándose en el flujo a través de toda la red. Un nodo con un valor elevado de esta métrica indica que el nodo es parte de muchos caminos más cortos en la red, lo que indica que es un nodo clave en la estructura y flujo de la información en la red. Formalmente se define la centralidad basada en la intermediación de un nodo v_i como:

$$V_{BC}(v_i) = \sum_{s \neq v_i \neq t} \frac{g_{st}(v_i)}{g_{st}} \tag{6}$$

donde $g_{st}(v_i)$ es el número de caminos más cortos entre s y t que pasan a través de v_i, y g_{st} es el número total de caminos más cortos entre s y t. Para el cálculo de los caminos más cortos entre dos vértices se suele utilizar la búsqueda en anchura prioritaria (BFS) en el caso de un grafo sin pesos en las aristas o arcos, y el algoritmo de Dijkstra* si el grafo es ponderado.

*https://es.wikipedia.org/wiki/Algoritmo_de_Dijkstra

La **centralidad basada en la proximidad** (V_{CC}, *closeness centrality*) de un nodo se define como la inversa de la distancia media entre este nodo y todos los nodos accesibles desde él. Formalmente se define como:

$$V_{CC}(v_i) = \frac{1}{\sum_{j \neq i} d(i,j)} \tag{7}$$

donde $d(i,j)$ representa la distancia entre los nodos v_i y v_j.

3.1.2. Basadas en propiedades espectrales

Los vectores y valores propios del espectro** de una red contienen información importante sobre la estructura y topología de la red. Por ello, en el análisis de redes se utilizan ciertos valores para analizar propiedades de la estructura y topología de red.

**https://en.wikipedia.org/wiki/Spectral_graph_theory

Dos métricas importantes se relacionan con los valores (*eigenvalues*) y vectores (*eigenvectors*) propios del espectro de la red. La descomposición espectral de una matriz, por ejemplo la matriz de adyacencia A de una red, se realiza de la siguiente manera:

$$A = \sum_i \lambda_i e_i e_i^T \tag{8}$$

donde e_i es el vector propio correspondiente al valor propio λ_i.

La primera de ellas hace referencia al **primer valor propio de la matriz de adyacencia A** (λ_1, *largest eigenvalue of the adjacency matrix A*). Este se calcula a partir de los valores propios de la matriz de adyacencia A, λ_i, donde $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_n$. Los valores propios de A codifican información relevante acerca de los ciclos en la red y del tamaño del diámetro.

La segunda métrica hace referencia al **segundo valor propio más pequeño de la matriz de Laplace** L (μ_2, *second smallest eigenvalue of the Laplacian matrix L*), donde μ_i son los valores propios de L y $0 = \mu_1 \leq \mu_2 \leq \ldots \leq \mu_m \leq m$. Los valores propios de L codifican información sobre la estructura de árbol dentro de la red. μ_2 es un importante valor propio de la matriz de Laplace y puede ser utilizado para cuantificar el grado de disgregación entre las distintas comunidades en la red, mostrando valores menores para identificar comunidades con estructuras más claras. La matriz de Laplace se define como:

$$L = D - A \tag{9}$$

donde $D_{n\times n}$ es la matriz diagonal que contiene la suma de filas de A a lo largo de la diagonal y 0 en las demás posiciones.

3.1.3. Basadas en otras propiedades

El **coeficiente de agrupamiento** (V_C, *clustering coefficient*) es una métrica muy utilizada en la literatura de análisis de redes. Para cada nodo indica la fracción de posibles triángulos en los que participa el nodo. Dicho de otra manera, este coeficiente indica la probabilidad de que dos nodos adyacentes a un determinado nodo sean también adyacentes entre ellos. El valor del coeficiente de agrupamiento de un nodo se calcula de la siguiente forma:

$$V_C(v_i) = \frac{2T(v_i)}{deg(v_i)(deg(v_i)-1)} \tag{10}$$

donde $T(v_i)$ es el número de triángulos en los que participa el nodo v_i.

Ejemplo de cálculo de métricas de nodos

La red *Zachary's karate club* es una de las redes sociales más conocidas y estudias en distintos ámbitos de las ciencias sociales y detección de comunidades. La red presenta un club de kárate de 34 miembros que fue estudiado por Wayne W. Zachary durante un periodo de tres años, desde 1970 hasta 1972. Los resultados fueron publicados en el artículo "An Information Flow Model for Conflict and Fission in Small Groups"*.

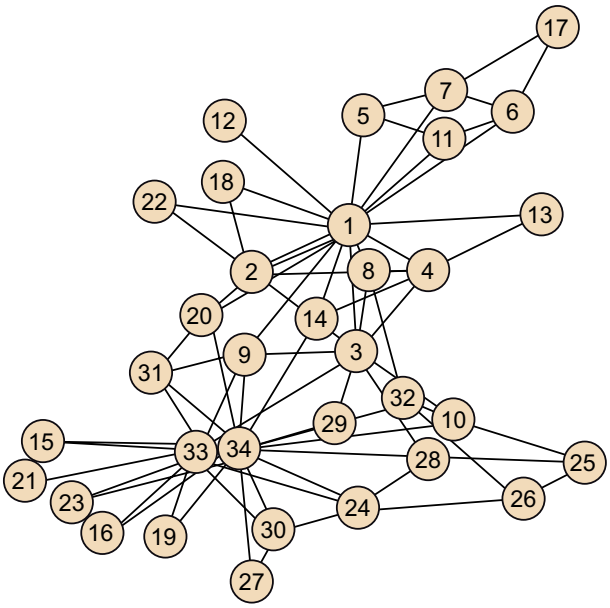
*Disponible en http://www1.ind.ku.dk/complexLearning/zachary1977.pdf

Durante el estudio surgió un conflicto entre el administrador (nodo 34) y el instructor (nodo 1) del club, que llevó a la división del club en dos. La mitad de los miembros formó un nuevo club en torno al instructor, mientras que los miembros de la otra parte encontraron un nuevo instructor o renunciaron al kárate. Zachary predijo correctamente la decisión final de todos los miembros, excepto para el miembro 9.

La figura 9 introduce la red Zachary's karate club*, una pequeña red simétrica de 34 nodos y 78 aristas, donde una arista entre dos nodos indica una relación de amistad fuera del club de kárate.

*Disponible en https://networkdata.ics.uci.edu/data.php?id=105

Figura 9. Representación de la red *Zachary's karate club*



Lectura complementaria

W. Zachary (1977). "An Information-flow model for conflict and fission in smallgroups". *Journal Of Anthropological Research* (vol. 33, pp. 452-473).

A continuación, vamos a calcular los tres tipos de centralidad para los nodos v_1, v_{17} y v_{28} del grafo de la figura 9. Aplicando la fórmula 5, vemos que el número de vecinos es 16, 2 y 4, respectivamente, mientras que el número total de nodos del grafo es 34. Los resultados se muestran en la tabla 1, y las funciones necesarias para este cálculo se muestran en el código 1.

Tabla 1. Cálculo de las métricas basadas en centralidad

Métrica	v_1	v_{17}	v_{28}
V_{DC}	0,485	0,060	0,121
V_{BC}	0,438	0,000	0,022
V_{CC}	0,569	0,284	0,458

Cálculo $V_{DC}(v_1)$

Aplicando la fórmula 5 vemos que el grado del nodo v_1 es igual a 16 y $n = 34$. Por lo tanto, tenemos $V_{DC}(v_1) = \frac{16}{34-1} = 0{,}485$.

Código 1: Cálculo de métricas basadas en centralidad con "igraph" (fichero metricas-1.R)

```
# carga de la libreria
library(igraph);

# carga del grafo usado en las demostraciones
karate <- read.graph(file="http://networkdata.ics.uci.edu/data/karate/karate.gml", format=
     "gml");

# calculo de la centralidad basada en el grado
degree(karate, v=c(1,17,28), normalized=TRUE);
# calculo de la centralidad basada en la intermediacion
betweenness(karate, v=c(1,17,28), normalized=TRUE);
# calculo de la centralidad basada en la proximidad
closeness(karate, v=c(1,17,28), normalized=TRUE);
```

Podemos ver que la centralidad basada en la intermediación del nodo v_{17} es 0. Observando la topología de la red en la figura 9, apreciamos que el nodo en cuestión se encuentra en la parte "exterior" de la red, y que ningún camino pasa a través del nodo v_{17}, excepto los que terminan en el propio nodo. Por lo tanto, ningún flujo de información pasará a través del nodo v_{17}. Contrariamente, casi la mitad de los caminos cortos entre los nodos de la red pasarán a través del nodo v_1. Es decir, v_1 es un nodo clave en la estructura de la red, ya que una parte muy importante del flujo de información circula a través de él.

La centralidad basada en la proximidad muestra valores altos (≈ 1) cuando los demás nodos de la red se encuentran muy cerca del nodo explorado, mientras que el valor es bajo (≈ 0) cuando la distancia a los demás nodos crece. Aunque el orden de importancia se mantiene, los tres nodos consiguen valores más próximos entre ellos que en los casos anteriores.

Aunque cada una de estas métricas evalúan la centralidad desde una perspectiva diferente, los resultados que muestran, aunque con matices importantes, están correlacionados entre ellos. Este resultado no nos sorprende; un nodo con un número elevado de vecinos es probable que aparezca en un número significativo de caminos cortos en el grafo y, por otro lado, que su distancia al resto de los nodos sea pequeña.

3.2. Métricas de aristas y arcos

Las métricas de aristas (y arcos) son aquellas métricas que se calculan para cada arista de la red de manera independiente. En consecuencia, se obtiene un valor distinto e independiente para cada arista de la red. Dicho valor puede ser calculado para una única arista o para cada una de las aristas de la red, formando un vector de resultados de longitud m.

La **centralidad basada en la intermediación** de una arista (E_{BC}, *edge betweenness centrality*) se calcula como la fracción de caminos más cortos que pasan a través de esta arista. Esta métrica mide la centralidad de una arista basándose en el flujo a través de toda la red. Una arista con un valor elevado de esta métrica indica que la arista es parte de muchos caminos más cortos en la red, y que por lo tanto será clave en la estructura y flujo de la información en la red. Formalmente se define la centralidad basada en la intermediación de una arista $\{i,j\}$ como:

$$E_{BC}\{i,j\} = \sum_{s \neq t} \frac{g_{st}\{i,j\}}{g_{st}} \tag{11}$$

donde $g_{st}\{i,j\}$ es el número de caminos más cortos entre s y t que pasan a través de la arista $\{i,j\}$, y g_{st} es el número total de caminos más cortos en s y t.

Ejemplo de cálculo de métricas de aristas

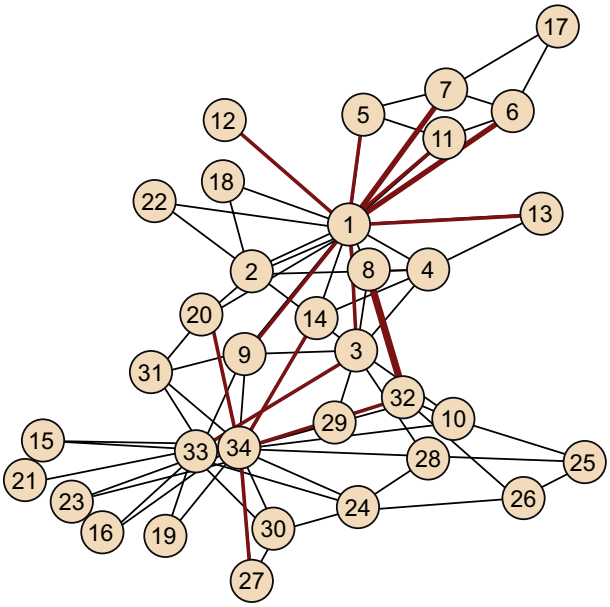
Continuando con el mismo ejemplo, la figura 10 muestra el resultado en forma de gráfico de calcular la centralidad basada en la intermediación de las aristas de la red. En la línea 8 del código 2 calculamos el valor de centralidad de cada arista del grafo, y luego asociamos un valor proporcional entre 0 y 10 al grosor de la arista, de tal modo que simplificamos la visualización de los datos obtenidos.

Código 2: Cálculo de métricas de aristas con "igraph" (fichero metricas-2.R)

```
1  # carga de la libreria
2  library(igraph);
3
4  # carga del grafo usado en las demostraciones
5  karate <- read.graph(file="http://networkdata.ics.uci.edu/data/karate/karate.gml", format=
       "gml");
6
7  # valores de centralidad basada en la intermediacion
8  ebc <- edge.betweenness(karate, directed=FALSE);
9  # nodes are proportional to their degree value
10 plot(karate, edge.width=(ebc/max(ebc))*10);
```

Observando la figura 10 es fácil identificar la arista $\{v_1,v_{32}\}$ como la arista con un valor de centralidad de intermediación más elevado.

Figura 10. Centralidad basada en la intermediación de las aristas de la red *Zachary's karate club*



3.3. Métricas de red

Las métricas de red son aquellas que se aplican a toda la red, es decir, que evalúan una cierta propiedad global a toda la red. El resultado de aplicar una métrica de red es un único valor para toda la red.

La **distancia media** (N_{AD}, *average distance*) se define como la media de las distancias entre cada par de nodos de la red. Evalúa el número medio mínimo de aristas entre cualquier par de nodos de la red. Su valor se calcula utilizando la ecuación 12.

$$N_{AD}(G) = \frac{\sum_{i,j} d(i,j)}{\binom{n}{2}} \tag{12}$$

donde $d(i,j)$ es la longitud del camino más corto entre los nodos v_i y v_j.

El **diámetro** (N_D, *diameter*) se define como la mayor de las distancias mínimas entre cualquier par de nodos de la red. Formalmente:

$$N_D(G) = max\{d(v_i,v_j)|v_i,v_j \in V\} \tag{13}$$

El **coeficiente de agrupamiento** global se calcula como la media de los valores del coeficiente de agrupamiento de cada nodo. Esta métrica calcula la fracción de posibles triángulos que existen en la red. En algunos textos se refieren a esta propiedad como **transitividad** (*transitivity*), pero el concepto es el mismo. Se expresa de la siguiente forma:

$$N_C(G) = \frac{1}{n}\sum_{i=1}^{n} V_C(v_i) \tag{14}$$

donde $V_C(v_i)$ es el valor del coeficiente de agrupamiento para el nodo v_i, descrito en la ecuación 10.

Ejemplo de cálculo de métricas de red

Continuando con el ejemplo anterior de la red *Zachary's karate club*, figura 9, vamos a ver el cálculo de algunas de las métricas de red. La distancia media (línea 8 del código 3) de la red es 2,408. Es decir, dos nodos cualquiera de la red están unidos por un camino con una longitud media de 2,408. Por otro lado, el diámetro de la red es de 5. Es decir, de entre todos los caminos más cortos que unen dos nodos de la red, el más largo de ellos es el camino {15,33,3,1,6,17} de longitud 5. Finalmente, el valor del coeficiente de agrupamiento es de 0,255. Este valor indica que si dado un nodo v_i, elegimos dos nodos vecinos al azar, v_p y v_k, la probabilidad de que v_p y v_k estén conectados entre ellos es de 0,255.

Código 3: Cálculo de métricas de red con "igraph" (fichero metricas-3.R)

```
1  # carga de la libreria
2  library(igraph);
3
4  # carga del grafo usado en las demostraciones
5  karate <- read.graph(file="http://networkdata.ics.uci.edu/data/karate/karate.gml", format=
       "gml");
6
7  # distancia media
8  average.path.length(karate, directed=FALSE);
9  # diametro de la red
10 get.diameter(karate, directed=FALSE);
11 # coeficiente de agrupamiento
12 transitivity(karate);
```

3.4. Complejidad en redes grandes

Las métricas que hemos presentado en los subapartados anteriores pueden ser utilizadas sin problemas en redes pequeñas o medianas. En el caso de redes grandes, cuando hablamos de cientos de miles, millones o cientos de millones de nodos, la complejidad espacial y temporal* debe ser considerada para poder aplicar las métricas con garantías de éxito.

*https://es.wikipedia.org/wiki/Eficiencia_Algorítmica

Veamos, en primer lugar, el caso de la centralidad basada en el grado (V_{DC}). Esta métrica se computa de manera independiente para cada uno de los nodos de la red. Para obtener el valor de esta métrica en un nodo concreto, solo hemos de conocer el número de vecinos o de aristas del nodo. Este cálculo se puede hacer fácilmente a partir de una representación en formato de lista de adyacencia o, incluso, a partir de una matriz de adyacencia. Para calcular el valor de todos los nodos de la red, debemos aplicar la misma operación sobre cada uno de los nodos. Esta métrica tiene una complejidad temporal de orden $\mathcal{O}(n)$**, es decir, que crece de forma lineal con el número de nodos de la red. Por lo tanto, puede ser aplicada a una red grande, siempre y cuando dispongamos de los recursos de cálculo necesarios para cargar y procesar la

**https://es.wikipedia.org/wiki/Notación_de_Landau

información.

Por el contrario, algunas métricas pueden tener una complejidad muy elevada si se quieren aplicar en redes de gran tamaño. La complejidad temporal puede crecer de forma exponencial, lo que hace imposible el uso de ciertas operaciones de cálculo.

Veamos, por ejemplo, el uso de métricas que implican el cálculo del camino más corto entre dos nodos. Entre otras métricas, la centralidad basada en la intermediación de un nodo (V_{BC}), la centralidad basada en la intermediación de una arista (E_{BC}), la distancia media (N_{AD}) y el diámetro (N_D) requieren el cálculo del camino más corto entre todos los pares de nodos del grafo. Este proceso tiene una complejidad temporal de orden $\mathcal{O}(n^3)$ utilizando el algoritmo de Floyd-Warshall*. En el caso de redes dispersas, el algoritmo de Johnson** es más eficiente, con una complejidad temporal de orden $\mathcal{O}(n^2 log(n) + nm)$. En ambos casos supone una complejidad temporal impracticable en redes grandes, con millones o cientos de millones de nodos.

*https://es.wikipedia.org/wiki/Algoritmo_de_Floyd-Warshall

**https://es.wikipedia.org/wiki/Algoritmo_de_Johnson

Los científicos trabajan para desarrollar nuevos algoritmos, con complejidades menores que los algoritmos existentes en la actualidad. Sin embargo, la propia naturaleza de algunos de estos problemas dificulta en gran medida la obtención de algoritmos suficientemente rápidos para lidiar con redes grandes. En estos casos, dos técnicas ampliamente utilizadas son las siguientes:

- En primer lugar, el desarrollo de algoritmos paralelos permite reducir la complejidad de los problemas. Dependiendo del problema que hayamos de tratar, su paralelización puede resultar desde sencilla hasta tremendamente compleja. Por eso, el desarrollo de algoritmos paralelos puede aportar mejoras significativas para el cálculo de ciertas métricas en redes grandes, pero no es aplicable en todos los casos.

- En segundo lugar, otra de las técnicas más empleadas es la aproximación. Es decir, en lugar de calcular la métrica para todos los nodos del grafo, se calcula solo para un subconjunto de ellos y se extrapola para el resto. En estos casos, obtener una muestra significativa del conjunto total de nodos o aristas es de gran importancia.

4. Detección de comunidades

Las redes sociales son uno de los ejemplos más claros de estructuras de red que presentan una clara organización en forma de comunidades. En estas redes, es habitual poder distinguir comunidades representado familias, grupos de amigos, compañeros de trabajo, etc. De todos modos, las redes sociales no son las únicas estructuras de red que presentan una arquitectura de comunidades: las redes de coautoría de artículos académicos presentan comunidades que agrupan tópicos de investigación; las de redes de interacción de proteínas muestran comunidades que agrupan proteínas que tienen la misma función dentro de la célula; el grafo de la Web agrupa páginas que escriben sobre un mismo tópico, y así un largo etcétera.

Lectura complementaria

Fortunato, S. (2010). "Community detection in graphs". *Physics Reports* (vol. 486 (3-5), pp. 75-174). Disponible en `http://arxiv.org/abs/0906.0612`

Mientras que esta definición informal de comunidad es ampliamente aceptada por investigadores que realizan análisis de redes, no existe una única definición formal consensuada de comunidad. La mayoría de las definiciones se basan en dos conceptos intuitivos que definen informalmente una **comunidad**:

- Por un lado, podemos definir informalmente una comunidad como un conjunto de nodos que se encuentran altamente conectados entre ellos y, a la vez, poco conectados con los nodos del resto de la red.

- Por otro lado, podemos definir una comunidad como un conjunto de nodos que comparten alguna propiedad y/o tienen un rol similar dentro de la red.

Generalmente, intentaremos identificar comunidades que cumplan ambas propiedades, dado que empíricamente suelen estar relacionadas. Es decir, los nodos que comparten alguna propiedad o tienen un rol similar dentro de la red suelen estar conectados a nodos con los que comparten la misma propiedad o el mismo rol, y viceversa. Por ejemplo, en el caso de las redes sociales, podemos encontrar comunidades formadas por familias, grupos de amigos o compañeros de trabajo. En todos estos casos, los individuos de una misma comunidad suelen estar fuertemente relacionados y, al mismo tiempo, compartir una propiedad (familia, grupo de amistad o trabajo).

La detección de comunidades (*community detection* en inglés) se centra en el descubrimiento o identificación de comunidades en una red concreta de manera automática.

El problema de **detección de comunidades** consiste en clasificar los nodos $v \in V$ de un grafo $G = (V,E)$ en subgrupos C_i, de manera que $C_i \subseteq V$, $1 \leq i \leq k$ y que los nodos dentro de cada C_i sean similares entre ellos, para alguna definición de similitud y para algún valor de k.

Cuando se afronta un problema de detección de comunidades, el número de comunidades que obtener es normalmente un valor desconocido. Además, las comunidades no suelen tener un tamaño constante y es habitual encontrar divisiones que presentan comunidades de tamaños significativamente distintos. Del mismo modo, también es común que la densidad que presenta cada comunidad como subgrupo sea distinta.

Dependiendo de la información que se quiera obtener en el proceso de detección de comunidades, puede ser interesante permitir que un mismo vértice pertenezca a más de una comunidad o, por el contrario, restringir la pertenencia de cada nodo a una única comunidad. Una **partición** es una división del grafo en grupos tales que cada vértice $v \in V$ pertenece a un solo grupo. Por el contrario, se usa el término **cobertura** para nombrar la división del grafo en comunidades solapadas, tales que un mismo vértice puede pertenecer a más de una comunidad.

4.1. Estrategias de detección de comunidades

Existen tres estrategias básicas, según la forma en que proceden para determinar las comunidades presentes en la red (tanto el número como los individuos que la forman).

4.1.1. Agrupamiento jerárquico

El **agrupamiento jerárquico** es un método de detección de comunidades que busca construir una jerarquía de grupos. El resultado de estos algoritmos es una agrupación multinivel, que permite analizar las comunidades con distintas granularidades, y que generalmente se presenta en forma de un dendrograma.

Dendrograma

Un dendrograma es un tipo de representación gráfica en forma de árbol que organiza los datos en subcategorías que se van dividiendo en otras hasta llegar al nivel de detalle deseado.

Las estrategias para el agrupamiento jerárquico pueden clasificarse en dos grandes grupos:

- Los algoritmos **aglomerativos** parten de una configuración inicial que asigna cada vértice a su propio clúster, y van agrupando los clústeres que presentan mayor similitud para formar clústeres de tamaño superior sucesivamente. Los algoritmos aglomerativos utilizan un procedimiento *de abajo arriba*, ya que parten de las unidades individuales y van construyendo componentes de mayor tamaño.

- Los algoritmos **divisivos** parten de un único clúster que contiene todos los nodos del grafo y van dividiendo sucesivamente los clústeres, eliminando las aristas que presentan una similitud más baja. Los algoritmos divisivos usan un procedimiento *de arriba abajo*, partiendo de una división a muy alto nivel con poco detalle y creando cada vez comunidades más específicas.

Para poder decidir qué grupos deberían ser combinados (en caso de aglomerativo), o cuándo un grupo debería ser dividido (en caso de divisivo), se requiere una medida de similitud entre conjuntos de observaciones. En la mayoría de los métodos de agrupamiento jerárquico, esto se logra mediante el uso de una métrica apropiada y un criterio de enlace que especifica la similitud de conjuntos como una función de las distancias dos a dos entre observaciones en los conjuntos.

4.1.2. Agrupamiento espectral

Las técnicas de **agrupamiento espectral** hacen uso de los valores y vectores propios de ciertas matrices relacionadas con el espectro de los grafos para realizar una reducción de dimensionalidad antes de la agrupación en un menor número de dimensiones.

La matriz de similitud es una de las más usadas en los algoritmos de agrupamiento espectral. Se puede definir como una matriz simétrica A, donde $A_{ij} \geq 0$ representa una medida de la similitud entre los nodos v_i y v_j. Existen multitud de medidas para cuantificar la similitud entre los pares de nodos de un grafo, lo que da lugar a distintas aproximaciones y algoritmos para agrupamiento espectral.

4.1.3. Agrupamiento dinámico

Los **algoritmos dinámicos** se basan en recorrer el grafo para extraer su estructura de comunidades. Por ejemplo, algunos algoritmos realizan *random walks* sobre el grafo, con la idea de que un recorrido aleatorio del grafo pasará mucho tiempo dentro de una comunidad densa debido al gran número de caminos posibles que esta contiene antes de saltar a otra comunidad.

4.2. Algoritmos

4.2.1. El algoritmo de Girvan-Newman

El algoritmo de Girvan-Newman es un algoritmo jerárquico divisivo que utiliza la centralidad basada en la intermediación para crear comunidades cada vez más específicas. En cada paso de la iteración, el algoritmo busca la arista con un valor mayor de intermediación y la elimina. Una arista con un valor alto de intermediación es aquella que forma parte de muchos de los caminos

cortos entre los nodos de la red. Por lo tanto, estas aristas son claves en la unión de los grupos de nodos. La idea que subyace a este proceso es que eliminando las aristas con mayor valor de intermediación, las comunidades o grupos de nodos se van aislando las unas de las otras. Este proceso se repite de forma iterativa hasta que el valor de modularidad es óptimo.

Detección de comunidades con el algoritmo de Girvan-Newman

La librería "igraph" incluye varias funciones para el cálculo de comunidades. Cada una de ellas se basa en una característica de la red para determinar la partición óptima de los nodos en distintas comunidades o grupos.

En el ejemplo de código 4 utilizamos el algoritmo de Girvan-Newman, llamado *cluster_edge_betweenness* en la librería "igraph", (línea 8) para determinar las comunidades de la red. Este algoritmo determina los grupos de nodos en función de las aristas que existen entre los miembros del grupo y los nodos externos al grupo. En la siguiente parte del código (líneas 9 hasta la 13) creamos un abanico de colores, pintamos los nodos con el color de la comunidad a la que corresponden y mostramos el resultado en un gráfico donde cada comunidad o grupo presenta un color distinto.

Referencia bibliográfica

Girvan, M.; Newman, M. (2004). "Finding and evaluating community structure in networks". *Physical Review E* (vol. 69, 026113).

Código 4: Cálculo y visualización de las comunidades utilizando el algoritmo de Girvan-Newman (fichero comunidades-1.R)

```
1  # carga de la libreria
2  library(igraph);
3
4  # carga del grafo usado en las demostraciones
5  karate <- read.graph(file="http://networkdata.ics.uci.edu/data/karate/karate.gml", format=
       "gml");
6
7  # ejecutar algoritmo Girvan-Newman
8  com <- cluster_edge_betweenness(karate);
9  # asignar el color a cada comunidad y pintar los nodos
10 colbar <- rainbow(max(com$membership));
11 V(karate)$color <- colbar[com$membership];
12 # mostrar el grafo con las comunidades
13 plot(karate, vertex.label.dist=0);
14 # mostrar el dendograma
15 dendPlot(com);
```

La figura 11 muestra el dendrograma generado por el algoritmo divisivo (línea 15), donde podemos ver cómo parte un primer clúster inicial que agrupa todos los nodos del grafo, y progresivamente se dividen en grupos más pequeños. El resultado de la detección de comunidades se puede ver en la figura 12.

Figura 11. Dendrograma de las comunidades detectadas por el algoritmo Girvan-Newman

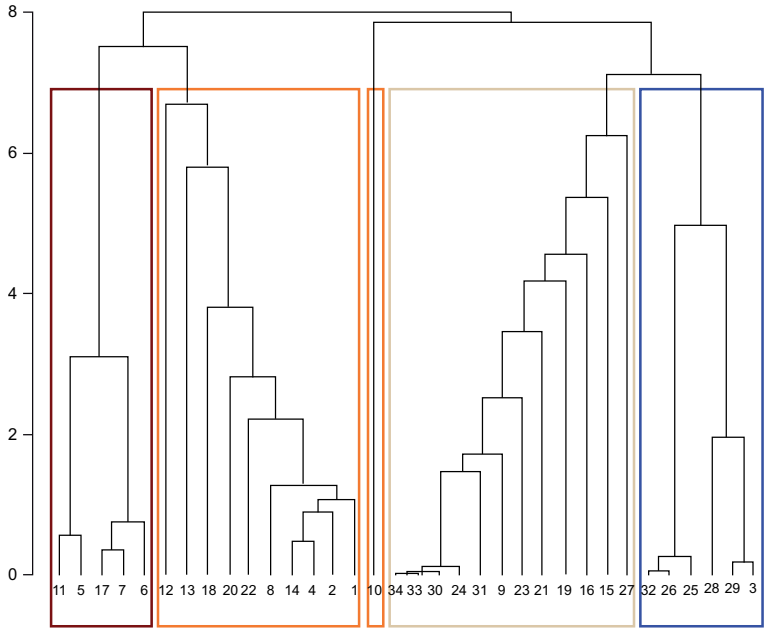
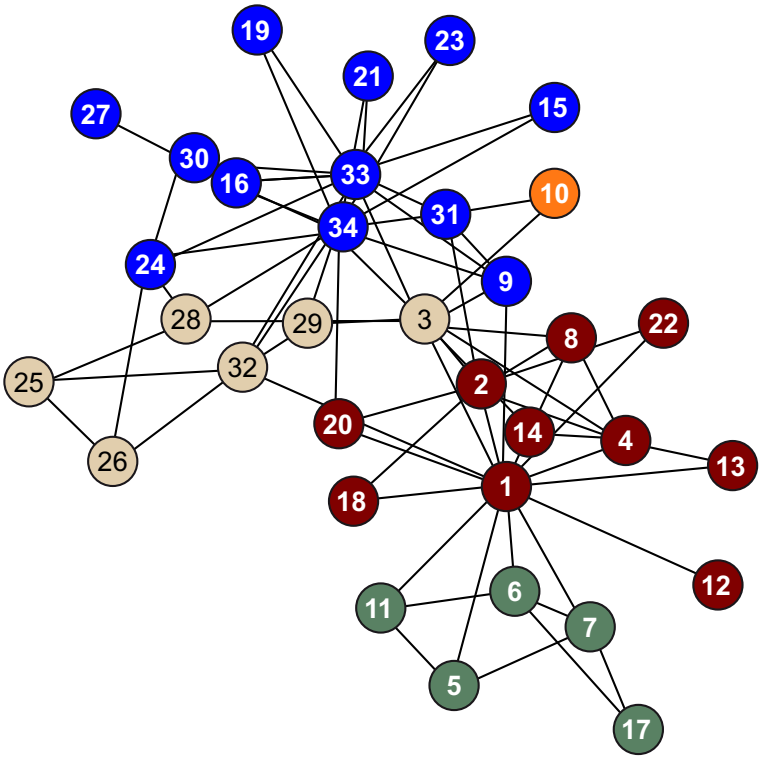


Figura 12. Gráfico que muestra las comunidades detectadas por el algoritmo Girvan-Newman

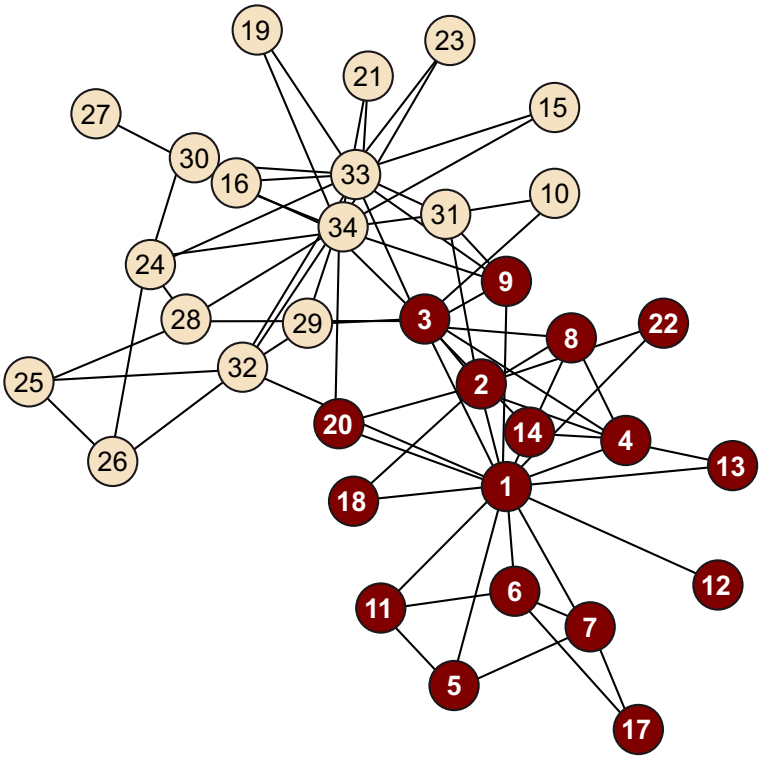


Tal y como hemos comentado, el club de kárate se partió en dos bloques, uno que apoyaba al administrador (nodo 34) y otro al instructor (nodo 1) del club. La figura 13 muestra los dos grupos que se generaron en torno al administrador y al instructor después de la división del club. Por lo tanto, podemos utilizar el resultado final de la división como test para evaluar la precisión del algoritmo de detección de comunidades.

Precisión

La precisión indica el porcentaje de nodos correctamente agrupados considerando una configuración como resultado válido. En este caso concreto, consideramos la división real del club de kárate como el resultado válido.

Figura 13. Gráfico que muestra las comunidades reales en que se dividió el club de kárate



Si consideramos la partición óptima según el algoritmo de Girvan-Newman, que considera la modularidad (podéis ver el subapartado 4.3) para determinar el punto óptimo dentro del dendrograma, obtenemos cinco particiones o clústeres. En este caso, la precisión del resultado es del 47 %. Obviamente, la precisión se ve afectada por el hecho de tener cinco grupos o particiones en lugar de los dos grupos del resultado que hemos considerado "válido".

Si nos fijamos en el dendrograma, podemos modificar el punto de división de las particiones, de manera que si lo modificamos podemos obtener solo tres particiones o clústeres. Es decir, uno formado por la rama principal izquierda, otro formado únicamente por el nodo 10 y un tercer grupo formado por la rama principal derecha. Es interesante notar que el administrador y el instructor están en la rama izquierda y derecha respectivamente. Es decir, quedan separados en particiones distintas. Si evaluamos esta configuración, obtenemos una precisión del 94 %, es decir, el algoritmo clasifica correctamente 32 de los 34 nodos del grafo.

4.2.2. El algoritmo de propagación de etiquetas de Raghavan

El algoritmo de propagación de etiquetas (*label propagation algorithm*) es un algoritmo jerárquico aglomerativo. Inicialmente, el algoritmo asigna una etiqueta única a cada nodo del grafo, e iterativamente actualiza las etiquetas de los nodos en función de las etiquetas de sus vecinos, de manera que la nueva etiqueta de un nodo es aquella que más aparece entre sus vecinos. Cuando el algoritmo converge, las etiquetas restantes identifican las diferentes comunidades detectadas.

A diferencia del algoritmo anterior, este no es determinista. Es decir, diferentes ejecuciones del algoritmo sobre los mismos datos pueden producir resultados distintos (diferentes particiones o grupos de nodos en las mismas particiones). Esto se debe a que la componente aleatoria introducida por el algoritmo detecta ciertos empates en las etiquetas vecinas de un nodo.

Detección de comunidades con el algoritmo de propagación de etiquetas

En el código 5 se utiliza el algoritmo de propagación de etiquetas, llamado *cluster_label_prop* en la librería "igraph", (línea 8) para determinar las comunidades de la red. Este algoritmo determina los grupos de nodos propagando las etiquetas de cada nodo, que en el momento inicial son únicas para cada nodo.

Referencia bibliográfica

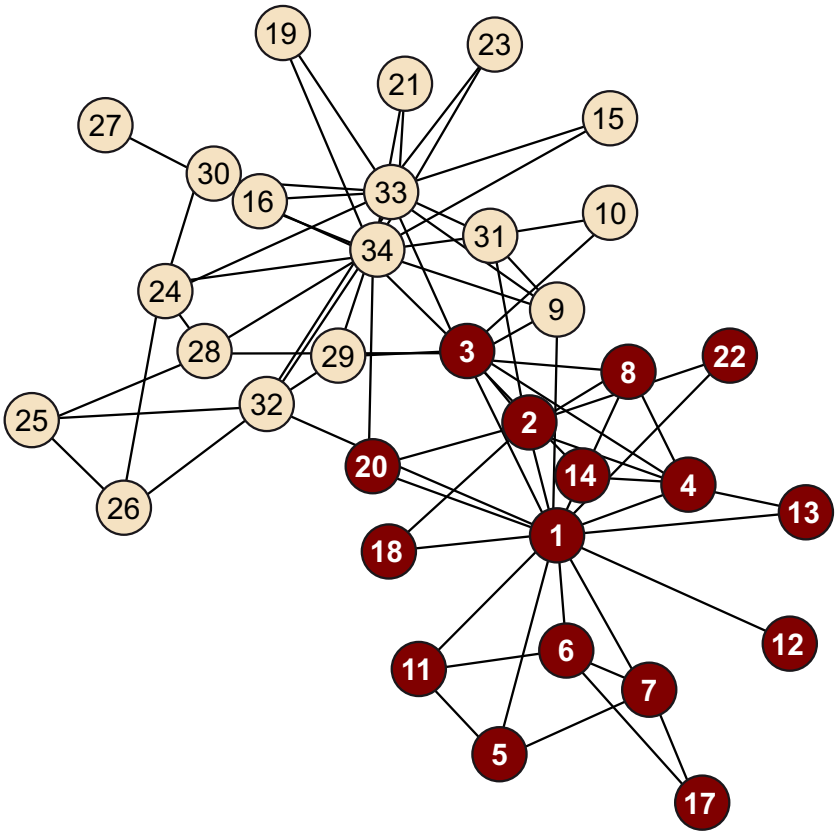
Raghavan, U. N.; Albert, R.; Kumara, S. (2007). "Near linear time algorithm to detect community structures in large-scale networks". *Physical Review E* (vol. 76, 036106).

Código 5: Cálculo y visualización de las comunidades utilizando el algoritmo de propagación de etiquetas (fichero comunidades-2.R)

```
1  # carga de la libreria
2  library(igraph);
3
4  # carga del grafo usado en las demostraciones
5  karate <- read.graph(file="http://networkdata.ics.uci.edu/data/karate/karate.gml", format=
       "gml");
6
7  # ejecutar algoritmo propagacion etiquetas
8  com <- cluster_label_prop(karate);
9  # asignar el color a cada comunidad y pintar los nodos
10 colbar <- rainbow(max(com$membership));
11 V(karate)$color <- colbar[com$membership];
12 # mostrar el grafo con las comunidades
13 plot(karate, vertex.label.dist=0);
```

En este caso, el algoritmo detecta dos comunidades dentro del grafo, como se puede ver en la figura 14.

Figura 14. Gráfico que muestra las comunidades detectadas por el algoritmo de propagación de etiquetas



Como hemos comentado, este algoritmo no es determinista y puede generar soluciones distintas en cada ejecución. Comparando el resultado del algoritmo con la división real del club, obtenemos unos valores de precisión de entre un 60 y un 97 %, según la ejecución. Es interesante remarcar la notable diferencia entre dos ejecuciones consecutivas con los mismos parámetros, debido a la aleatoriedad que incorpora el algoritmo.

4.2.3. El algoritmo *leading eigenvector*

El algoritmo *leading eigenvector* intenta encontrar subgrafos densamente conectados dentro del grafo original mediante cálculos basados en el vector propio de la matriz de modularidad del grafo. La matriz de modularidad se define como $B = A - P$, donde A es la matriz de adyacencia y P es una matriz que contiene, para cada elemento $P(i,j)$, la probabilidad de que exista una arista entre los nodos v_i y v_j en una red generada aleatoriamente pero manteniendo los grados de los nodos.

Detección de comunidades usando el algoritmo *leading eigenvector*

La librería "igraph" incluye el algoritmo "leading eigenvectors", que se basa en los vectores propios de la matriz de adyacencia del grafo y el cálculo de la modularidad para determinar las comunidades. Para este ejemplo, utilizaremos el código 6. En la línea 8 vemos que ejecutamos el algoritmo "cluster_leading_eigen", que es el nombre que asigna la librería "igraph" a este algoritmo. El resto del código es similar al mostrado en los ejemplos anteriores.

Referencia bibliográfica

Newman, M. E. J. (2006). "Finding community structure using the eigenvectors of matrices". *Physical Review E* (vol. 74, 036104).

Código 6: Cálculo y visualización de las comunidades utilizando el algoritmo *leading eigenvector* (fichero comunidades-3.R)

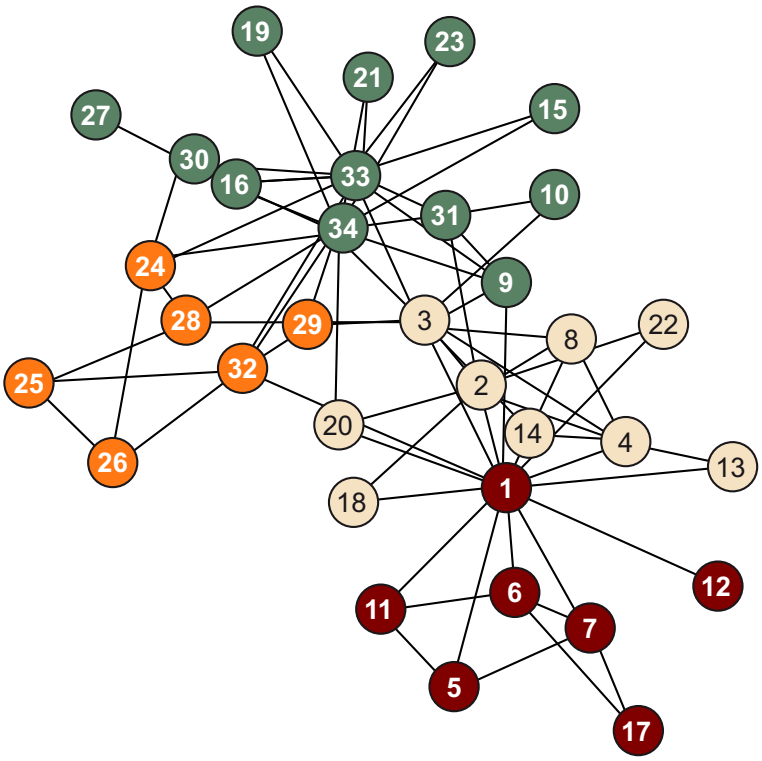
```
# carga de la libreria
library(igraph);

# carga del grafo usado en las demostraciones
karate <- read.graph(file="http://networkdata.ics.uci.edu/data/karate/karate.gml", format=
      "gml");

# ejecutar agoritmo leading eigenvector
com <- cluster_leading_eigen(karate);
# asignar el color a cada comunidad y pintar los nodos
colbar <- rainbow(max(com$membership));
V(karate)$color <- colbar[com$membership];
# mostrar el grafo con las comunidades
plot(karate, vertex.label.dist=0);
```

La figura 15 muestra el resultado de la detección de comunidades. En este caso vemos la división del conjunto de nodos en cuatro comunidades, y contrariamente al ejemplo del algoritmo de Girvan-Newman, podemos ver que las comunidades están más compensadas (considerando el número de nodos en cada una de ellas).

Figura 15. Gráfico que muestra las comunidades detectadas por el algoritmo *leading eigenvector*



Comparando el resultado obtenido por el algoritmo con el resultado real de la división del club, figura 13, podemos ver que en este caso el número de comunidades detectadas por el algoritmo vuelve a ser mayor (4 en lugar de 2).

Si consideramos las cuatro particiones de la red, obtenemos una precisión del 55 %. Sin embargo, si agrupamos las particiones dos a dos, obtenemos una distribución de clústeres muy similar al resultado original obtenido por Zachary en su estudio. Se clasifican correctamente 33 de los 34 nodos de la red, con lo que se obtiene una precisión del 97 %.

4.2.4. Algoritmo *Infomap*

El algoritmo *Infomap* utiliza una estrategia de agrupamiento dinámica. Este algoritmo determina la estructura de las comunidades que minimizan la trayectoria de un recorrido aleatorio, o *random walks*. El algoritmo aprovecha el hecho de que un recorrido basado en el azar debe quedar atrapado dentro de las comunidades, permaneciendo allí por mucho más tiempo que el que pasará cambiando entre comunidades.

Detección de comunidades usando *Infomap*

La librería "igraph" incluye dos algoritmos basados en el concepto de *random walks*: "Infomap" y "Walktrap". Para este ejemplo, utilizaremos el código 7. En la línea 8 ejecutamos el algoritmo "cluster_infomap", que es el nombre que asigna la librería "igraph" al algoritmo *Infomap*. El resto del código es similar al mostrado en los ejemplos anteriores.

Referencia bibliográfica

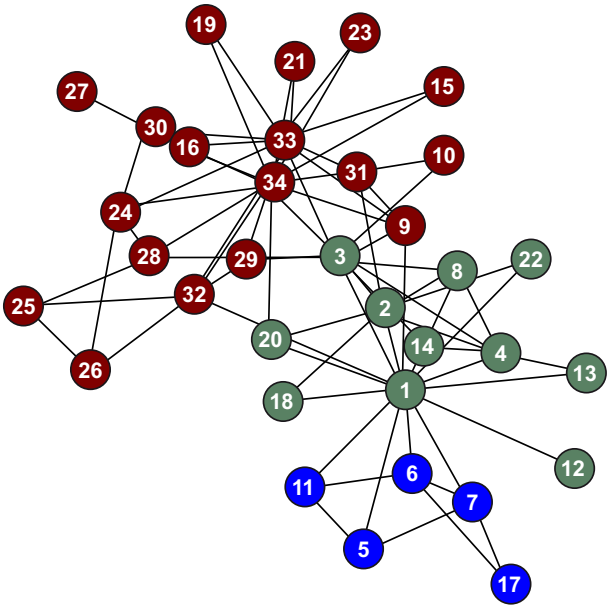
Rosvall, M.; Bergstrom, C. T. (2008). "Maps of information flow reveal community structure in complex networks". *PNAS* (105, 1118).

Código 7: Cálculo y visualización de las comunidades utilizando el algoritmo *Infomap* (fichero comunidades-4.R)

```
1  # carga de la libreria
2  library(igraph);
3
4  # carga del grafo usado en las demostraciones
5  karate <- read.graph(file="http://networkdata.ics.uci.edu/data/karate/karate.gml", format=
       "gml");
6
7  # ejecutar agoritmo Infomap
8  com <- cluster_infomap(karate);
9  # asignar el color a cada comunidad y pintar los nodos
10 colbar <- rainbow(max(com$membership));
11 V(karate)$color <- colbar[com$membership];
12 # mostrar el grafo con las comunidades
13 plot(karate, vertex.label.dist=0);
```

La figura 16 muestra el resultado de la detección de comunidades. En este caso, el número de comunidades es tres, y el valor de la precisión es de un 85 %. Si agrupamos dos particiones obtenemos la misma configuración que el resultado original después de la división del club, es decir, todos los nodos correctamente clasificados y una precisión del 100 %.

Figura 16. Gráfico que muestra las comunidades detectadas por el algoritmo *Infomap*



4.3. Evaluación de la modularidad

De la ejecución de un algoritmo de detección de comunidades sobre un grafo concreto obtendremos una partición del grafo en comunidades. Estas comunidades resultantes serán normalmente distintas dependiendo del algoritmo usado, de sus parámetros y, si este incluye aleatoriedad, de incluso de la ejecución de este. El número de comunidades resultantes también puede variar mucho con el algoritmo elegido. Por este motivo, es interesante disponer de herramientas que permitan evaluar las comunidades obtenidas.

La **modularidad** es una de las métricas más usadas para evaluar la división en comunidades de un grafo. La idea básica de la modularidad es considerar que un grafo aleatorio no tiene estructura de comunidades. Entonces, las comunidades detectadas en el grafo se evalúan comparando la densidad dentro de cada comunidad con la densidad que tendrían en un grafo aleatorio de características similares.

Existen diferentes formulaciones que definen la modularidad dependiendo de con qué modelo se compara el grafo. Una de las más populares consiste en crear el grafo que vamos a comparar a partir del grafo original, modificando las aristas de forma aleatoria pero manteniendo el grado de los nodos. De este modo, se consigue un grafo con la misma secuencia de grados que el grafo original, pero con una configuración de las aristas tal que no existe estructura de comunidades. Entonces, se define la modularidad Q como:

$$Q = \frac{1}{2m} \sum_{\forall i,j} \left(A_{ij} - \frac{deg(v_i)\ deg(v_j)}{2m} \right) \delta(v_i, v_j) \tag{15}$$

$$\delta(v_i, v_j) = \begin{cases} 1, & \text{si } C(v_i) = C(v_j) \\ 0, & \text{en cualquier otro caso} \end{cases} \tag{16}$$

donde $C(v_i)$ indica la comunidad del nodo v_i.

Así, la modularidad se define teniendo en cuenta la diferencia entre el número de aristas entre cada par de nodos (A_{ij}) y el número de aristas que tendría un grafo aleatorio manteniendo el grado de los nodos implicados $\left(\frac{deg(v_i)\ deg(v_j)}{2m}\right)$, para todos los pares de nodos que pertenecen a una misma comunidad. Nótese que la inclusión del factor $\delta(v_i, v_j)$ permite considerar únicamente las aristas cuyos nodos pertenecen a una misma comunidad. En caso contrario, la diferencia entre las aristas existentes y las estimadas para un grafo aleatorio no contribuye al sumatorio, ya que se anula al hacer el producto con la función δ.

Cálculo de la modularidad

El código 8 muestra la comparación del valor de la modularidad de los ejemplos de algoritmos vistos anteriormente, más otro algoritmo de agrupamiento jerárquico, llamado "multilevel", y otro algoritmo dinámico basado en *random walks*, llamado "walktrap".

Código 8: Cálculo de la modularidad (fichero modularity.R)

```
# carga de la libreria
library(igraph);

# carga del grafo usado en las demostraciones
karate <- read.graph(file="http://networkdata.ics.uci.edu/data/karate/karate.gml", format=
      "gml");

# lista de algoritmos a testear
modularity(edge.betweenness.community(karate))
modularity(multilevel.community(karate))
modularity(leading.eigenvector.community(karate));
modularity(infomap.community(karate));
modularity(walktrap.community(karate));
```

El código 9 muestra los valores de modularidad obtenidos por cada uno de los algoritmos. Como podemos observar, los valores son similares para todos ellos, excepto el algoritmo "Walktrap", que consigue un valor de modularidad sensiblemente inferior a los demás.

Código 9: Resultado de la modularidad

```
[1] 0.4012985
[1] 0.4188034
[1] 0.3934089
[1] 0.4020381
[1] 0.3532216
```

Es interesante remarcar que los algoritmos vistos en los ejemplos anteriores, es decir Girvan-Newman, *leading eigenvectors* e *Infomap*, consiguen un valor similar de modularidad. No obstante, el número y la estructura de las comunidades, como hemos visto anteriormente, es sensiblemente diferente.

4.4. Selección del algoritmo

La selección del algoritmo de detección de comunidades idóneo para cada caso es una tarea compleja. Cada uno de ellos presenta sus puntos fuertes, así como sus debilidades e inconvenientes, que deben ser evaluados en el contexto y según los datos en que van a ser utilizados.

Adicionalmente, deberemos considerar dos aspectos determinantes antes de seleccionar un algoritmo de detección de comunidades adecuado para un problema concreto:

- **El tipo de red**. En primer lugar, es muy importante considerar el tipo de red o grafo con el que debemos trabajar. La mayoría de los algoritmos de *clustering* solo pueden lidiar con un tipo de redes (o unos pocos). Por ejemplo, el algoritmo *leading eigenvector* solo puede ser utilizado con grafos simétricos. Por lo tanto, si deseamos analizar una red dirigida, no podemos elegir este método. Aun así, también es interesante notar que todos los grafos pueden ser convertidos al tipo más elemental de grafo que existe: el grafo simétrico no ponderado. El proceso es simple, se eliminan los pesos o etiquetas de los nodos y aristas, los arcos (en caso de existir) se convierten en aristas,

se eliminan los lazos, las aristas múltiples, etcétera. Lógicamente, durante este proceso se pierde una gran cantidad de información, que por supuesto implicará la pérdida de conocimiento en el proceso de minería de datos aplicado posteriormente.

- **El tamaño de la red**. En segundo lugar, el tamaño de la red es también un factor determinante cuando se debe elegir un algoritmo de detección de comunidades. En el caso de lidiar con grafos de tamaño pequeño o medio, no suele haber ningún problema. Los problemas aparecen cuando queremos trabajar con una red de gran tamaño. Por ejemplo, el algoritmo de Girvan-Newman tiene una complejidad temporal de orden $\mathcal{O}(n^3)$. Por lo tanto, es inviable aplicar este algoritmo a redes de gran tamaño.

En la literatura de detección de comunidades en redes podemos encontrar distintos trabajos que realizan comparaciones entre los algoritmos más conocidos. Entre otros, destaca el trabajo "Community detection algorithms: a comparative analysis"*, de Lancichinetti y Fortunato. Para poder comparar de forma objetiva los distintos algoritmos de detección de comunidades, los autores hacen uso de los llamados bancos de pruebas (en inglés, *benchmark*).

Uno de los bancos de pruebas clásicos en la detección de comunidades en grafos fue desarrollado por Girvan y Newman. Este contiene un conjunto de redes de 128 nodos, divididos en cuatro comunidades de 32 nodos cada una de ellas. Los mismos autores de este trabajo proponen otro banco de pruebas**, que incluye distintos tipos de redes (simétricas, dirigidas y ponderadas).

Lancichinetti y Fortunato concluyen su estudio determinando que uno de los mejores algoritmos para la detección de comunidades en redes es Infomap. Según los autores, Infomap obtiene excelentes resultados en grafos simétricos, dirigidos y ponderados, por lo que proporciona flexibilidad ante distintos tipos de redes al mismo tiempo que unos valores de precisión elevados.

*Disponible en http://arxiv.org/abs/0908.1062

Lectura complementaria

Lancichinetti, A.; Fortunato, S. (2009). "Community detection algorithms: a comparative analysis". *Physical Review E* (vol. 80(5), 056117).

**Disponible en https://sites.google.com/site/santofortunato/inthepress2

Referencia bibliográfica

M. Girvan; M. E. Newman (2002), Proc. Natl. Acad. Sci. USA 99, 7821.

5. Visualización de redes

La visualización de redes o grafos es un área situada a caballo entre las matemáticas y las ciencias de la computación, que combina los métodos de la teoría de grafos y la visualización de la información para derivar representaciones bidimensionales de grafos. La visualización de una red es una representación pictórica de los vértices y las aristas que forman el grafo. Este gráfico o representación no debe confundirse con el grafo en sí mismo; configuraciones gráficas muy distintas pueden corresponder al mismo grafo. Durante los procesos de manipulación de los grafos, lo que importa es qué pares de vértices están conectados entre sí mediante aristas. Sin embargo, para la visualización de los grafos, la disposición de estos vértices y aristas dentro del gráfico afecta a su comprensibilidad, su facilidad de uso y su estética.

5.1. Estrategias de visualización

En este subapartado veremos algunas de las principales estrategias de visualización de redes enfocadas a la distribución de los nodos en un espacio bidimensional. Es decir, estas estrategias se centran en asignar posiciones (o coordenadas) a los nodos de la red con la finalidad de facilitar la visualización de la red, pero no consideran aspectos como el color o el tamaño de los nodos.

Se han definido múltiples métricas para evaluar de forma objetiva la calidad de una estrategia de visualización de grafos. Algunas de estas métricas, además, son utilizadas por las estrategias de disposición como una función objetivo que se trata de optimizar durante la colocación de los vértices, para mejorar la visibilidad de la información. Algunas de las métricas más relevantes son las siguientes:

- El número cruces se refiere al número de pares de aristas que se cruzan entre sí. Generalmente no es posible crear una disposición sin cruces de aristas, pero si se intenta minimizar el número de cruces se suelen obtener representaciones más sencillas y, por lo tanto, más fáciles de interpretar de forma visual.

- El área de representación del grafo es el tamaño del marco delimitador más pequeño. Las representaciones con un área más pequeña son, en general, preferibles a aquellas con un área mayor, ya que permiten que las características del grafo sean más legibles.

- La simetría permite hallar grupos con cierta simetría dentro del grafo dado y mostrarlas en la representación de este, de manera que se puedan identificar rápidamente en una observación visual.

- Es importante representar las aristas y los vértices utilizando formas simples para facilitar la identificación visual. Generalmente se aconseja representar las aristas utilizando formas rectas y simples, evitando las curvas innecesarias.

- La longitud de las aristas también influye en la representación de un grafo. Es deseable minimizar la longitud de las aristas y, además, mantener longitudes uniformes para todas las aristas que se representen.

- La resolución angular mide los ángulos entre las aristas que salen (o llegan) a un mismo nodo. Si un grafo tiene vértices con un grado muy grande (*hubs*), entonces la resolución angular será pequeña, pero en caso contrario se debe mantener una proporcionalidad que ayuda a la estética y legibilidad del grafo.

Existen múltiples estrategias de visualización de grafos, y de forma similar a la mayoría de los escenarios, cada una posee ciertos puntos fuertes y débiles. Consecuentemente, es importante conocer estas estrategias básicas para elegir en cada momento la que mejor se adapte al planteamiento y los datos con los que se está trabajando:

1) La **disposición aleatoria** (*random layout*) es la estrategia más simple de visualización. Consiste en distribuir los vértices de manera aleatoria en el espacio bidimensional y añadir las aristas correspondientes entre los pares de vértices.

2) La **disposición circular** (*circular layout*) coloca los vértices del grafo en un círculo; se elige el orden de los vértices alrededor del círculo intentando reducir los cruces de aristas y se colocan los vértices adyacentes cerca el uno del otro.

3) Los **diagramas de arcos** (*arc diagrams*) sitúan los vértices sobre una línea imaginaria. A continuación, las aristas se pueden dibujar como semicírculos por encima o por debajo de la línea.

4) Los sistemas de **disposición basados en fuerzas** (*force-based layout*) modifican de forma continua una posición inicial de cada vértice de acuerdo con un sistema de fuerzas basado en la atracción y repulsión entre vértices. Típicamente, estos sistemas combinan fuerzas de atracción entre los vértices adyacentes y fuerzas de repulsión entre todos los pares de vértices, con el fin de buscar una disposición en la que las longitudes de las aristas sean pequeñas y los vértices estén bien separados para facilitar la visualización.

5) La **disposición basada en el espectro** del grafo (*spectral layout*) utiliza los vectores propios de la matriz de Laplace, u otras matrices relacionadas con la matriz de adyacencia del grafo, como coordenadas de los vértices en el espacio bidimensional.

Matriz de Laplace

Es una matriz cuadrada de orden $n \times n$ tal que $L = D - A$, donde D es la matriz diagonal que contiene la suma de filas a lo largo de la diagonal y 0 en las demás posiciones, y A es la matriz de adyacencia.

6) Los métodos de **disposición ortogonal** (*orthogonal layout*) fueron diseñados originalmente para problemas de diseño de VLSI*. Típicamente emplean un enfoque de múltiples fases, en las que se intenta reducir el número de cruces entre aristas, trazar aristas que sean lo más rectas posibles y reducir el espacio de representación del grafo.

***Integración en escala muy grande (*very large scale integration*) de sistemas de circuitos basados en transistores en circuitos integrados.**

7) La **disposición basada en capas** (*layered-based layout*), a menudo también llamada de estilo Sugiyama, es la más adecuada para grafos dirigidos acíclicos o grafos casi acíclicos. En estos métodos, los vértices del grafo están dispuestos en capas horizontales, de tal manera que la mayoría de las aristas se encuentran en posición vertical desde una capa a la siguiente.

Ejemplo de representaciones de redes

Existe una gran cantidad de algoritmos de visualización de grafos. En este ejemplo, veremos algunos de ellos, que presentamos brevemente a continuación:

- layout_randomly: Esta disposición coloca los vértices de forma aleatoria en el espacio de dos dimensiones.
- layout_in_circle: Los vértices son ubicados de forma equidistante en torno a un círculo.
- layout_with_mds: El método de escalado dimensional sitúa los puntos de un espacio dimensional grande (≥ 3) en un espacio bidimensional, de manera que se intenta mantener la distancia entre los puntos en el espacio original.
- layout_with_gem: Este algoritmo sitúa los vértices en el plano utilizando la disposición basada en fuerzas del método GEM.

Las instrucciones para reproducir este ejemplo se pueden ver en el código 10. En primer lugar, la función *plot()* permite crear un gráfico de la red eligiendo la colocación de los nodos (*layout*) y el tamaño y color de nodos y aristas, entre muchos otros parámetros disponibles. En este caso, para ver la ayuda de la función *plot()* debemos teclear "?plot.igraph" en la línea de comando, ya que la función *plot()* está sobrecargada (es decir, existen distintas funciones según el objeto al que se aplica la función). Existen distintas funciones para crear la colocación de los nodos en el espacio 2D. En este caso hemos utilizado cuatro algoritmos para crear diferentes disposiciones, pero la librería "igraph" incorpora muchas más.

Referencia bibliográfica

Frick, A.; Ludwig, A.; Mehldau, H. (1995). "A Fast Adaptive Layout Algorithm for Undirected Graphs". *Proc. Graph Drawing* (1994, LNCS 894, pp. 388-403).

Código 10: Generación de varios ejemplos de visualización de redes (fichero representacion-1.R)

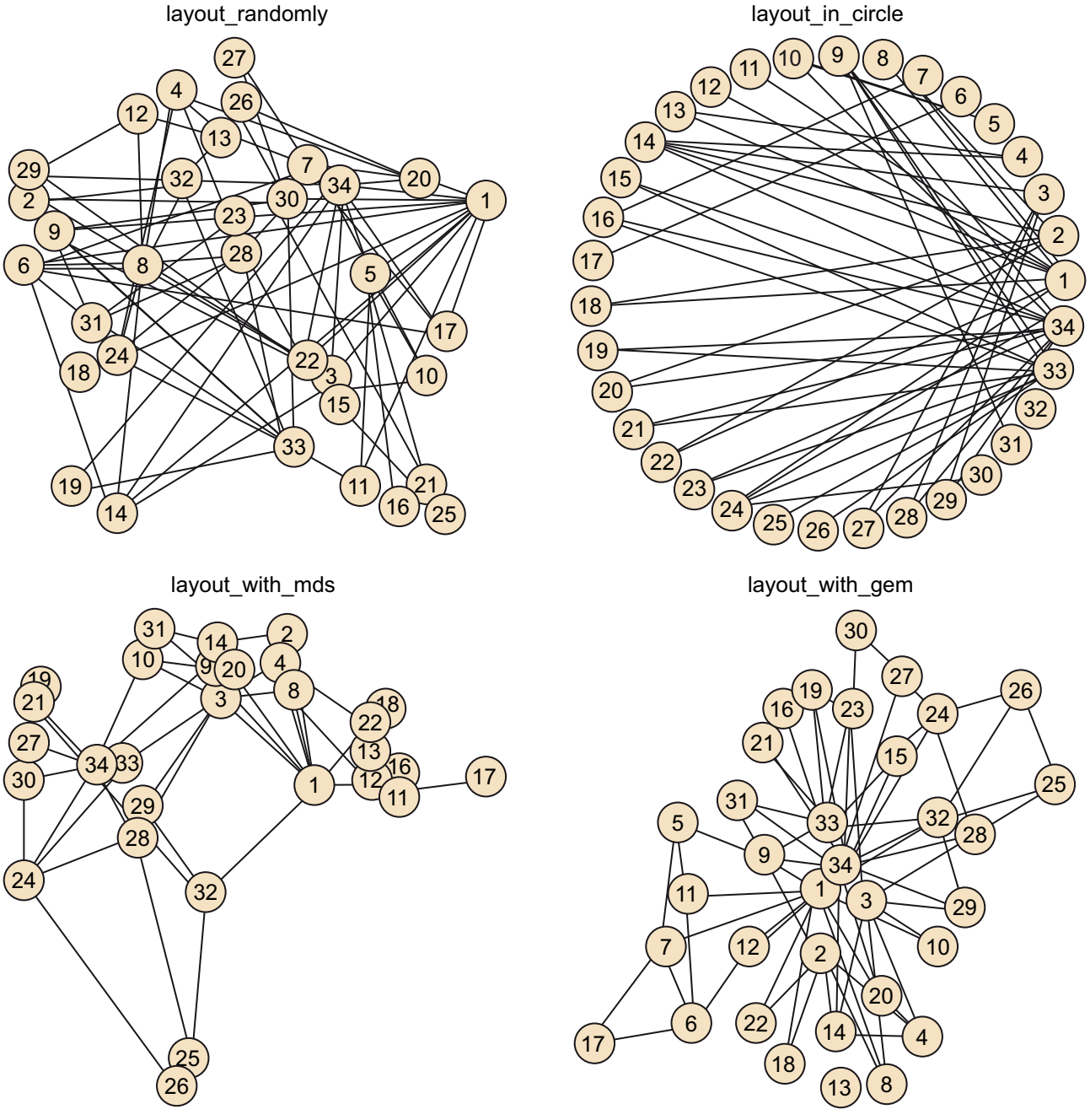
```
# carga de la libreria
library(igraph);

# carga del grafo usado en las demostraciones
karate <- read.graph(file="http://networkdata.ics.uci.edu/data/karate/karate.gml", format=
      "gml");

# disposicion aleatoria
plot(karate, layout=layout_randomly(karate));
# disposicion en circulo
plot(karate, layout=layout_in_circle(karate));
# disposicion utilizando escalado multidimensional
plot(karate, layout=layout_with_mds(karate));
# disposicion basada en fuerzas del metodo GEM
plot(karate, layout=layout_with_gem(karate));
```

La figura 17 muestra el resultado de los cuatro algoritmos descritos anteriormente. En primer lugar, se puede ver el algoritmo aleatorio, que muestra los vértices sin ningún tipo de agrupamiento, lo que dificulta su visibilidad. En segundo lugar, vemos la disposición en círculo de los vértices. Esta disposición puede ser útil en algunos casos donde existen pocas aristas y el grafo no presenta ningún tipo de estructura interna. En tercer y cuarto lugar, podemos ver los algoritmos de escalado multidimensional y basado en fuerzas. La disposición de los vértices en ambos casos ayuda a una correcta visualización de la red.

Figura 17. Ejemplos de representación de una red



5.2. La problemática del orden

La representación de datos mediante gráficos presenta algunos problemas, especialmente cuando queremos representar grandes volúmenes de datos. La visualización de grafos suele ser muy efectiva cuando se manejan conjuntos

de datos de un tamaño razonable, pero la complejidad aumenta mucho cuando se pretenden representar grandes conjuntos de datos, es decir, grafos con un orden (número de vértices) elevado.

Existen dos metodologías básicas para tratar con grafos de orden elevado:

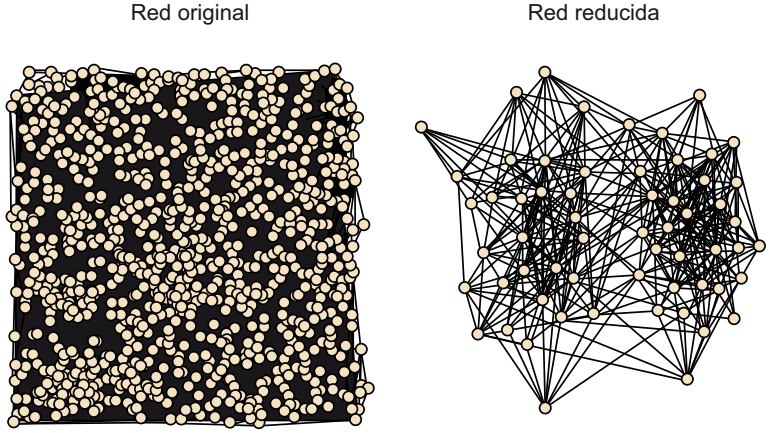
- Una primera alternativa consiste en representar no el conjunto entero de todos los vértices, sino un subconjunto reducido de valores agregados a partir de los datos originales. De esta manera se pretende reducir el tamaño de los datos y facilitar su visualización.

- Un segundo enfoque consiste en utilizar una estrategia de visualización dinámica, que permita un cierto granulado de la información. Por ejemplo, se puede visualizar la red en un primer momento mostrando solo los vértices más importantes (por ejemplo, los vértices con un grado elevado), y permitir al usuario focalizar la visualización en un punto de la red concreto, mostrando los vértices de menor grado en este punto. Es decir, se trata de hacer un tipo de "zoom" de las distintas zona que permita ver los "detalles" que puedan quedar ocultos al mostrar solo los vértices más importantes.

Ejemplo del problema de visualización de grafos grandes

La figura 18 muestra la estructura de dos redes. Como se puede observar, la figura de la izquierda muestra una red de tamaño medio, con 1.224 usuarios y 16.715 relaciones entre ellos. Como se puede ver en el gráfico, es muy difícil, si no imposible, obtener información alguna de los datos presentados en esta figura. El número de nodos y aristas es demasiado grande, con lo que resulta imposible identificar la estructura de la red, las comunidades que la componen o los nodos principales con un simple análisis visual.

Figura 18. Gráfico de una red social 1.224 vértices (izquierda) y una versión reducida que incluye los 60 vértices más importantes en función del grado (derecha)



Por otro lado, la figura de la derecha muestra un resumen de esta misma red social, donde solo se muestran los vértices más importantes. En este caso se ha definido la importancia de un vértice utilizando una métrica basada en el grado. Solo los vértices con grado superior a 100 son mostrados en esta red. Obtenemos, por lo tanto, un resumen de 60 vértices y 702 relaciones entre ellos. Dado el nuevo volumen de datos mostrados, es mucho más fácil identificar claramente la estructura de la red; identificando las distintas comunidades que la forman y los nodos principales, es decir, los individuos que están conectados a un gran número de usuarios y que actúan como concentradores de información (*hubs*).

El código 11 muestra los pasos seguidos para crear el resumen de la red (línea 8) y las visualizaciones correspondientes.

Código 11: Generación del ejemplo del problema del orden (fichero representacion-2.R)

```
1  # carga de la libreria
2  library(igraph);
3
4  # carga del grafo usado en las demostraciones
5  polblogs <- read.graph(file="http://networkdata.ics.uci.edu/data/polblogs/polblogs.gml",
       format="gml");
6
7  # mostrar solo los vertices con grado > 100
8  polblogs_red <- induced_subgraph(polblogs, which(degree(polblogs)>100));
9
10 # red original
11 plot(polblogs, layout=layout_with_fr, vertex.size=5);
12 # red reducida
13 plot(polblogs_red, layout=layout_with_fr, vertex.size=5);
```

6. Apéndice A: Notación

La tabla 2 presenta la notación empleada en este módulo en referencia a los grafos simétricos o no dirigidos.

Tabla 2. Notación empleada en este módulo para grafos simétricos

Símbolo	Descripción
$G = (V,E)$	Grafo simétrico o no dirigido
$V = \{v_1,v_2,\ldots,v_n\}$	Conjunto de nodos o vértices
$n = \|V\|$	Orden del grafo (número de nodos o vértices)
$E = \{e_1,e_2,\ldots,v_m\}$	Conjunto de aristas
$\{v_i,v_j\}$	Arista conectando $v_i \leftrightarrow v_j$
$m = \|E\|$	Tamaño del grafo (número de aristas)
$\Gamma(v_i)$	Conjunto de los nodos adyacentes o vecinos del nodo v_i
$\|\Gamma(v_i)\| = deg(v_i)$	Grado del nodo v_i
$\overline{deg}(G)$	Grado medio del grafo G
$d(v_i,v_j)$	Distancia del nodo v_i a v_j

De manera similar, la tabla 3 presenta la notación empleada en este módulo en relación con los grafos dirigidos o no simétricos.

Tabla 3. Notación empleada en este módulo para grafos dirigidos o asimétricos

Símbolo	Descripción
$G = (V,A)$	Grafo dirigido o asimétrico
$V = \{v_1,v_2,\ldots,v_n\}$	Conjunto de nodos o vértices
$n = \|V\|$	Orden del grafo (número de nodos o vértices)
$A = \{a_1,a_2,\ldots,a_m\}$	Conjunto de arcos
(v_i,v_j)	Arco conectando $v_i \rightarrow v_j$, pero no viceversa
$m = \|A\|$	Tamaño del grafo (número de arcos)
$\Gamma(v_i)$	Conjunto de sucesores del nodo v_i
$\|\Gamma(v_i)\| = deg_{out}(v_i)$	Grado exterior del nodo v_i
$\Gamma^{-1}(v_i)$	Conjunto de antecesores del nodo v_i
$\|\Gamma^{-1}(v_i)\| = deg_{in}(v_i)$	Grado interior del nodo v_i

Resumen

En este módulo didáctico hemos presentado los conceptos básicos relacionados con la representación y el análisis de redes sociales. Hemos iniciado el módulo introduciendo los conceptos básicos de teoría de grafos necesarios para el correcto desarrollo del resto del módulo. A continuación, hemos presentado algunas de las características más relevantes de las redes reales, tales como la ley de la potencia, la dispersión en redes reales o el conocido efecto *small world*.

En el siguiente apartado hemos definido algunas de las métricas más utilizadas para el análisis de redes. Estas métricas permiten cuantificar alguna propiedad de la red, lo que facilita la comparación entre redes y la comprensión de su estructura interna. Las principales estrategias y algoritmos para la detección de comunidades han sido presentados en el apartado siguiente. A continuación, hemos introducido las nociones básicas para la representación de redes en un espacio bidimensional, así como algunas estrategias comúnmente utilizadas y la problemática de representación de redes de orden elevado.

Bibliografía

Barabási, A. L. (2015). *Network Science*. Disponible en http://barabasi.com/networksciencebook/.

Borges, J.; Clarisó, R.; Masià, R.; Pujol, J.; Rifà, J.; Vancells, Villanueva, M. (2007). *Grafos y complejidad*. Barcelona: Fundació per a la Universitat Oberta de Catalunya.

Newman, M. E. J. (2011). *Networks: An introduction*. Nueva York: Oxford University Press.

Paradis, E. *R para Principiantes*. Disponible en http://cran.r-project.org/doc/contrib/rdebuts_es.pdf.