Universidad del País Vasco
Euskal Herriko Unibertsitatea

Facultad de Informática / Informatika Fakultatea

TITULACIÓN: Ingeniería Informática

Detección de comunidades en redes complejas basada en la sincronización.

Alumno/a: D./Dña. Gaizka San Sebastian Artola

Director/a: D./Dña. Abdelmalik Moujahid Moujahid

Proyecto Fin de Carrera, Julio de 2014

Resumen

El estudio de las redes complejas atrae cada vez más el interés de muchos investigadores por muchas razones obvias. Muchos sistemas tanto reales como tecnológicos pueden representarse como redes complejas, es decir, un conjunto de entidades en interacción de acuerdo a propiedades topológicas no triviales. La interacción entre los elementos de la red puede describir comportamientos globales tales como el tráfico en Internet, el servicio de suministro de electricidad o la evolución de los mercados.

Una de las propiedades topológicas de los grafos que caracterizan estos sistemas complejos es la estructura de comunidad. La detección de comunidades tiene como objetivo la identificación de los módulos o grupos con alguna o varias propiedades en común basándose únicamente en la información codificada en la topología del grafo.

La detección de comunidades es importante no sólo para caracterizar el grafo, sino que además ofrece información sobre la formación de la red así como sobre su funcionalidad. El estudio de las leyes subyacentes que gobiernan la dinámica y evolución de los sistemas complejos y la caracterización de sus grafos revela que las redes a gran escala, generalmente, se caracterizan por topologías complejas y estructuras heterogéneas. La estructura de conectividad de estas redes se manifiesta por la presencia de comunidades (clusters o grupos), es decir, conjuntos de nodos que comparten propiedades comunes o juegan roles similares en la red. Las comunidades pueden representar relaciones de amistad en las redes sociales, páginas web con una temática similar, o rutas bioquímicas en las redes metabólicas. Formalmente, una red es un grafo compuesto por un gran número de nodos altamente interconectados donde una comunidad se resalta por la presencia de un gran número de aristas conectando nodos dentro de grupos individuales, pero con baja concentración de aristas entre estos grupos.

El mejor modo para establecer la estructura de comunidad de una red compleja es un problema todavía sin resolver. Durante los últimos años, se han propuesto muchos algoritmos que persiguen extraer la partición óptima de una red en comunidades. El clustering

espectral, los algoritmos de particionamiento de grafos, los métodos basados en la modularidad o los algoritmos basados en la sincronización son sólo algunos de estos algoritmos de extracción de comunidades.

Los algoritmos dinámicos basados en la sincronización han sido estudiados por varios autores, y han demostrado que la monitorización del proceso dinámico de la sincronización permite revelar las diferentes escalas topologicas presentes en una red compleja. Muchos de estos algoritmos se basan en el modelo Kuramoto o en algunas de sus variantes como el modelo de opinión, donde cada oscilador aislado es modelado en un espacio unidimensional.

El objetivo principal del presente proyecto es la implementación de un algoritmo de detección de comunidades basado en la sincronización de osciladores acoplados. Cada oscilador ha sido modelado mediante el sistema dinámico de Rossler, un sistema de ecuaciones diferenciales definido en un espacio tridimensional.

Agradecimientos

En primer lugar, me gustaría agradecer la dedicación de la persona que ha supervisado la realización de éste proyecto desde el principio hasta el final, Abdelmalik Moujahid. Gracias por tener plena confianza en mí,y por guiarme en cada momento que lo he necesitado.

A mi pareja, por el apoyo continuo recibido y por animarme en los momentos más difíciles.

Por último, a mis padres que se han interesado por el progreso del proyecto y que han supuesto también un gran apoyo.

Índice general

Índice de figuras

Indice de tablas

1. CAPÍTULO

Introducción

En 1736, Leonhard Euler consiguió resolver el Problema de los puentes de Königsberg, el cual consistía en saber si era posible recorrer la ciudad pasando por sus siete puentes y cruzando sólo una vez cada uno de ellos, volviendo de nuevo al punto de partida. Para ello, representó cada puente mediante una línea (arista,link) que unía dos puntos(nodo/vértcie), cada uno de los cuales representaba una región distinta de la ciudad.

A partir de ese momento, considerado como el origen de la Teoría de Grafos, el estudio de los grafos y sus propiedades ha sido constante dentro del ámbito matemático. La aparición y sobre todo la amplia difusión de los ordenadores hace que el análisis de redes adquiera otra perspectiva y pasa a ser ampliamente utilizado en la mayoría de campos científicos.La posibilidad de generar computacionalmente grandes cantidades de datos y el creciente interés por el estudio de sistemas complejos y las interrelaciones entre sus componentes, hacen que la representación mediante redes sea idónea para la resolución de muchos problemas en áreas como la sociología, la física, la informática o la biología.

Por otra parte, la ciencia de la sincronización, hoy por hoy, dista mucho de ser totalmente resuelta. Desde aquel febrero de 1665, en el que el astrónomo, físico y matemático neerlandés, Christiaan Huygens (1629-1695), mediante su análisis del fenómeno de los relojes de péndulo pusiera la piedra fundamental de lo que hoy en día se conoce como la Teoría de los Osciladores Acoplados, este tipo de fenómenos han sido estudiados por diversos científicos y profesionales. Fenómenos conocidos como el de los chirridos de un grupo de grillos, el de la luz de las luciérnagas, el de las millones de células que se relajan y contraen para hacer latir el corazón, o el de cientos de aplausos que se sincronizan en

un concierto a partir de la cacofonía inicial son sólo unos ejemplos de la presencia de la sincronización en la naturaleza y en nuestras vidas.

Nuestro objetivo en este proyecto ha sido monitorizar el proceso hacia la sincronización con el fin de revelar la estructura de comunidad presente en una red compleja. Se trata entonces de resolver numéricamente el sistema de ecuaciones diferenciales que gobiernan la dinámica de una red cuya topología es dada por la matriz de conectividad del grafo. También, en este proyecto hemos considerado los algoritmos de generación grafos a partir de datos en formato registros. Lo cual permite la detección de comunidades tanto en redes descritas por grafos así como en datos que provienen del campo de la minería de datos (UCI Maching learning datasets).

Una vez terminada la investigación, hemos preparado una interface para que el usuario pueda realizar las pruebas que nosotros hemos realizado y consultar los resultados obtenidos.

2. CAPÍTULO

Documento Objetivos del proyecto

2.1. Objetivos

El objetivo principal de este proyecto es el diseño e implementación de un algoritmo de detección de comunidades en redes complejas basándose en el proceso de sincronización de osciladores acoplados. Para ello, se han considerado diferentes redes o grafos de uso extendido en la literatura para la validación de los algoritmos de detección de comunidades.

El algoritmo deberá ser flexible a las exigencias de la investigación y el usuario podrá definir los valores de ciertos parámetros transcendentes en la ejecución del mismo. Esto permitirá un posterior análisis de la incidencia de cada valor que influye en el algoritmo.

2.2. Metodología

El progreso del proyecto irá determinado por las conclusiones que se extraigan de las reuniones que se llevarán a cabo junto con el tutor del proyecto. El proyecto se divide en tres partes. En la primera se realiza lo que hemos denominado "selección y preparación de datos", en el cual se seleccionan los conjuntos de datos que se usarán en la investigación. Hemos considerado datos en formato de grafo disponibles en los diferentes repositorios de redes complejas, así como datos en formato de registros utilizados en maching learning. En ese último caso, es necesario transformar los datos en grafos requiriendo el uso de

diferentes algoritmos de generación de grafos. La segunda parte del proyecto es la implementación de un algoritmo de detección de comunidades basado en la sincronización de osciladores acoplados. Cada oscilador ha sido modelado mediante el sistema dinámicos de Rossler, un sistema de ecuaciones diferenciales definido en un espacio tridimensional. Los osciladores están acoplados de acuerdo a la matriz de conectividad describiendo la topología de la red. En la tercera y última parte, se desarrollará una fácil y sencilla interfaz para poder ejecutar el algoritmo desarrollado y obtener resultados. Una vez tengamos los resultados, se procederá a su análisis y resolución de conclusiones.

El desglose general del proyecto tiene varias partes:

- Procesos Tácticos: Abarca la parte administrativa y determina los pasos a llevar a cabo para poder completar el proyecto, así como determinar si en cada momento se están cumpliendo los objetivos.
- Procesos Formativos: Son el conjunto de tareas formativas que se llevarán a cabo para poder utilizar las herramientas necesarias para efectuar con éxito las tareas del proyecto.
- Procesos Operativos: Son las diferentes partes que forman los entregables del proyecto. Tras una reunión se determinará si cada entregable cumple los objetivos esperados o si requiere alguna modificación.

2.3. Gestión del proyecto

Cada uno de los apartados del proyecto tendrá posibles futuros puntos de trabajo y mejoras pendientes. Las reuniones nos permitirán durante el proyecto nos permitirán, por una parte, conocer en qué situación se encuentra el proyecto en cada momento y, por otra, proponer y acometer nuevas vías de mejora del proyecto base.

2.4. Estructura de Descomposición de Tareas（EDT）

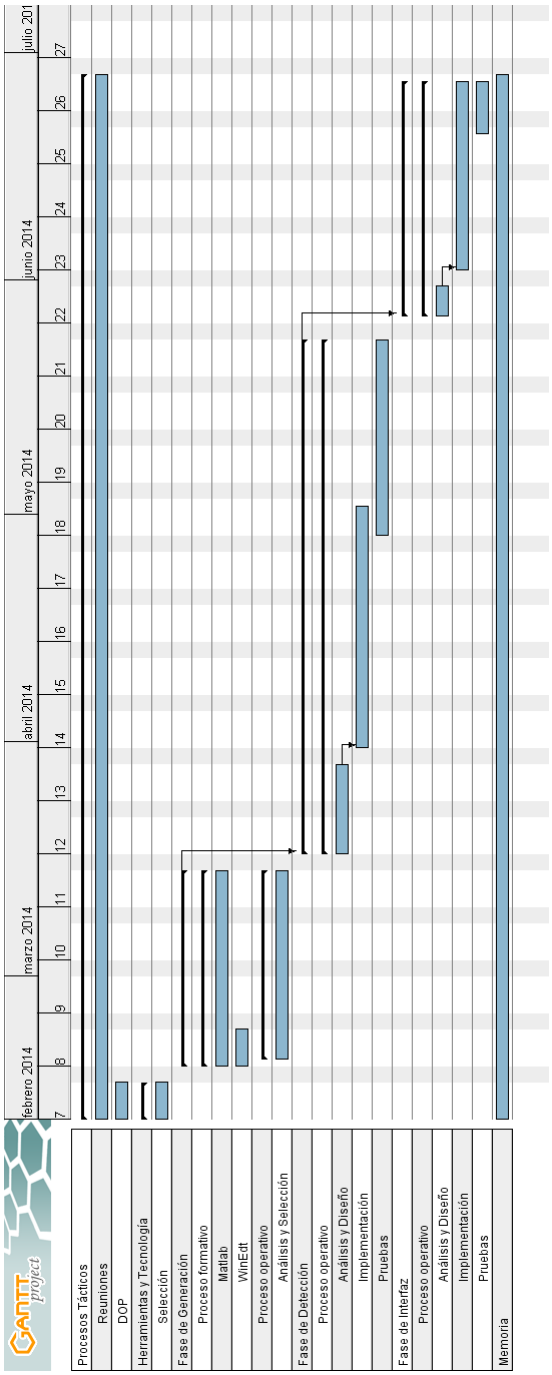

Figura 2.1: Diagrama *Gantt*.

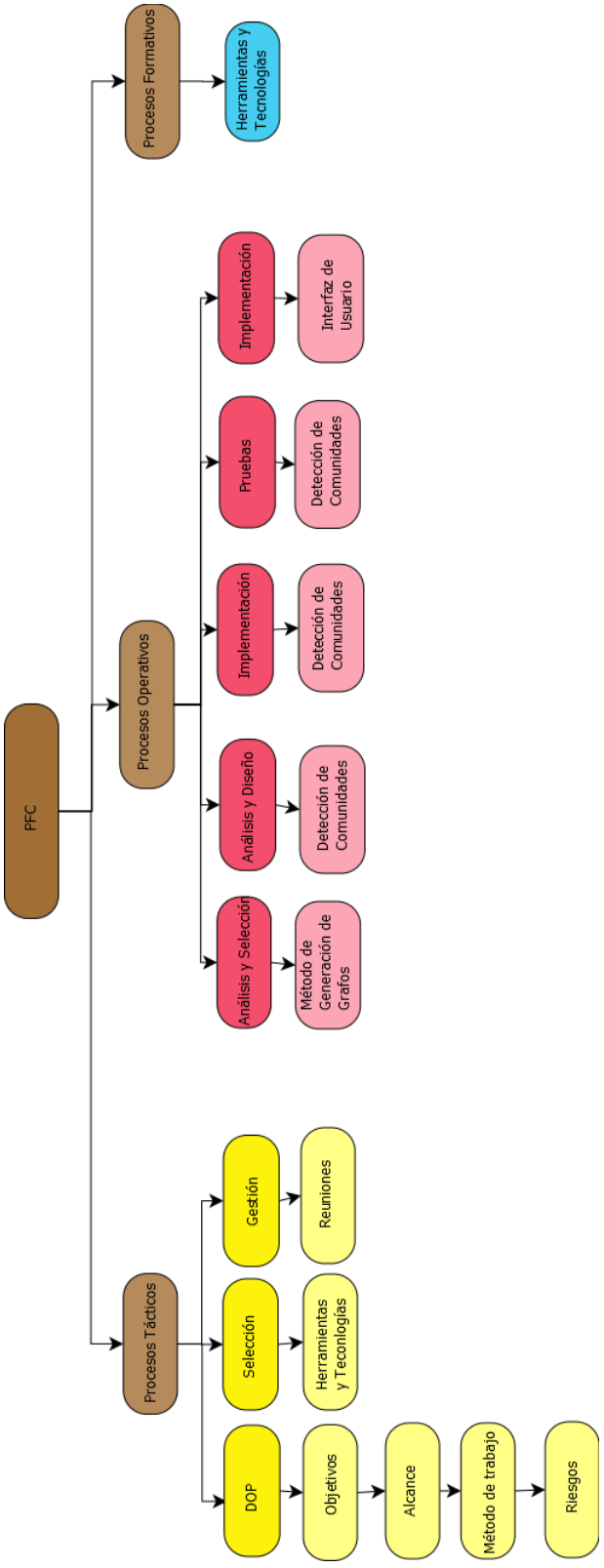

Figura 2.2: Diagrama *EDT*.

2.5. Riesgos y planes de contingencia

- Equipo estropeado
  - Descripción: un fallo en el hardware del equipo en el que se está desarrollando el proyecto. Puede que implique pérdida parcial o total del trabajo realizado.
  - Probabilidad: baja
  - Plan de contingencia: realizaremos copias de seguridad al finalizar cada sesión de trabajo en Dropbox, en el repositorio Bildu de la facultad y en un dispositivo USB Esto nos permitirá minimizar la pérdida a las últimas horas trabajadas. Por otra parte, se tendrán dos equipos preparados con el software necesario a nuestra disposición para poder trabajar en todo momento evitando así pérdida de tiempo en el caso de avería.
- Fallo del repositorio Bildu de la facultad
  - Descripción: El repositorio Bildu de la facultad deja de funcionar.
  - Probabilidad: baja
  - Plan de contingencia: tenemos varias copias del trabajo, en la nube mediante Dropbox y en un dispositivo USB.
- Fallo del servicio Dropbox
  - Descripción: El servicio de ficheros en la nube Dropbox deja de funcionar.
  - Probabilidad: baja
  - Plan de contingencia: tenemos varias copias del trabajo, en la el repositorio Bildu de la facultad y en un dispositivo USB.
- Fallo del dispositivo USB
  - Descripción: El dispositivo USB deja de funcionar.
  - Probabilidad: baja
  - Plan de contingencia: tenemos varias copias del trabajo, en la el repositorio Bildu de la facultad y en la nube mediante Dropbox.
- Enfermedad

  - Descripción: no poder trabajar en el proyecto durante unos días por enfermedad.
  - Probabilidad: baja
  - Plan de contingencia: hablar con el tutor y modificar si fuera necesario la planificación inicial.

- Cambios en los requisitos

  - Descripción: pueden darse pequeños cambios tanto en el desarrollo del algoritmo com en el desarrollo de la interfaz a causa de la naturaleza de investigación del proyecto.
  - Probabilidad: alta
  - Plan de contingencia: Los resultados que se vayan obteniendo irán marcando el camino a seguir. Adaptar la planificación inicial a los cambios.

- No poder acudir a una reunión

  - Descripción: el tutor del proyecto o el alumno no pueden acudir a una reunión.
  - Probabilidad: media
  - Plan de contingencia: comunicarse vía email y modificar la fecha de la reunión. Abordar ciertos puntos en el mail para poder avanzar en el proyecto mientras llegue la siguiente reunión.

2.6. Métodos de trabajo

A la hora de acometer el proyecto, las distintas fases del mismo necesitarán de varios procesos:

- Procesos tácticos

  - Gestión del proyecto: se llevará a cabo realizando un seguimiento del diagrama Gantt, así como en reuniones con el tutor del proyecto.
  - Reuniones con el tutor del proyecto: la periodicidad de las mismas será dinámica y dependerá de la situación de los miembros en cada momento. Se comentarán los resultados que se van obteniendo en la realización del proyecto y se irán marcando nuevos hitos.

    - Memoria: la redacción se llevará a cabo en WinEdt.

- Procesos formativos

    - El tutor del proyecto decidirá las herrammientas adecuadas para el desarrollo de este proyecto.
    - Una vez decidido, el alumno se formará en las herramientas para adquirirlos conocimientos necesarios para el desarrollo del proyecto.

- Procesos operativos

    - Se buscará información relativa al estado del arte actual. Se consultarán diferentes métodos de generación de grafos y se consultará el sistema de Rossler.
    - Se procederá a realizar un análisis en profundidad de los métodos seleccionados.
    - Se desarrollará el algoritmo basado en el sistema de Rossler.
    - Se desarrollará la interfaz para poder ejecutar el algoritmo y obtener resultados.

2.7. Herramientas

2.7.1. MatLab

Matlab es la herramienta idónea para este tipo de trabajos ya que es un lenguaje orientado a las ciencias matemáticas y en sus librerías se encuentran numerosas funciones predefinidas que en cualquier otro tendríamos que desarrollar. Además, el poder trabajar con estructuras matriciales facilita mucho la implementación y ahorra mucho recurso computacional.

2.7.2. Gephi

Gephi es una plataforma interactiva de visualización y exploración de todo tipo de redes y sistemas complejos, grafos dinámicos y jerárquicos. En este proyecto lo hemos usado para visualizar los grafos y mostrarlos en este documento.

2.7.3. WinEdt

WinEdt es un editor de texto potente y versatil, muy enfocado a la creación de documentos en LaTex. Se ha utilizado para la creación y edición de la memoria.

2.8. Resto de la memoria

El capítulo 3 presenta un completo análisis de los métodos de generación de grafos selecionados para el proyecto. La dinámica de la red será presentada en el capítulo 4 así como una descripción detallada del algoritmo de detección de comunidades desarrollado. Posteriormente, en el capítulo 5, se mostrarán los resultados obtenidos y el análisis de los mismos. En el capítulo 6 se presentará el diseño de la interfaz desarrollada y se comentarán sus características y funcionalidades más importantes.

Para terminar, se darán las conclusiones que se han ido sacando durante la realización del proyecto.

3. CAPÍTULO

Generación de Grafos

La construcción de grafos a partir de datos es un primer paso en muchas tareas de las ciencias de la computación como pueden ser la *regresión* o el *spectral clustering*. La representación de estos datos mediante redes complejas es idónea para la resolución de problemas dadas las posibilidades que un grafo nos ofrece. En este apartado se analizan los métodos utilizados y las características de cada uno de ellos.

3.1. Conceptos generales

Formalmente un grafo [8] es una tupla $G = (V, E, W)$, donde V es el conjunto de nodos ($|V| = n$), E es el conjunto de enlaces($|E| = m$) y W es la matriz $n \times m$ de los pesos de los enlaces donde el peso de un enlace (i, j) es dado por W_{ij}. En el grafo construído G, el nodo correspondiente al ejemplo x_i, se llamará x_i. El *grado* de un nodo se define como el nº de enlaces que tiene con el resto de nodos. Si no se dice lo contrario, se asumen las siguientes características topológicas:

- El grafo inducido es no-dirigido y todos los pesos de los enlaces son positivos, esto es, $W_{ij} = W_{ji}$ y $W_{ij} > 0$.
- $W_{ij} = 0$ indica que no hay enlace entre los nodos i y j.
- $W_{ii} = 0 \; \forall i$.

Los nodos(V) están establecidos, el problema real es aprender la matriz de pesos de los enlaces W sujeta a las restricciones arriba mencionados.

3.1.1. Matriz de Adyacencias

La matriz de adyacencias es una de las formas de representar un grafo, y podemos definirla de la siguiente manera: la matriz de adyacencias A_{ij} de un grafo finito G de n nodos, es una matriz de dimensiones $n \times n$ donde cada elemento de la matriz a_{ij} representa el nº de enlaces entre el nodo i y el nodo j. En nuestro caso, trabajaremos con grafos no dirigidos, por lo que la matriz de adyacecnias A_{ij} tiene las siguientes características:

- Es simétrica
- Es una matriz con ceros en su diagonal, esto es, $a_{ii} = 0$

La matriz puede ser binaria o ponderada. Si es binaria significa que el valor de a_{ij} será 1 cuando hay un enlace entre los nodos i y j, y 0 si no la hay. En el caso de la matriz ponderada, los valores varían en base al peso del enlace.

En este proyecto trabajaremos tanto con matrices binarias como ponderadas. Cuando tengamos una matriz ponderada, la normalizaremos.

3.2. Métodos de Generación de Grafos

Estos métodos reciben como parámetro de entrada un conjunto de datos en forma matricial, de dimensiones $N \times M$ donde N es el nº de ejemplos de la base de datos y M el nº de atributos de cada ejemplo. Procesan los datos y generan una matriz de connectividad W, cada uno basándose en su propia metodología, que será el grafo que necesitamos para trabajar.

3.2.1. *k*NN

Uno de los métodos más clásicos, el *k*NN o *k* Nearest Neighbor se basa en el siguiente concepto general: calcular la distancia/similitud de cada uno de los ejemplos con el resto y seleccionar los k vecinos más cercanos. El método se descompone en dos procesos independientes. Primero se calcula la matriz de adyacencias y con esto los *links* o

enlaces quedan establecidos. Luego, se calculan los pesos de los links,esto es, se calcula la matriz de afinidad ponderada. Estos pesos pueden ser calculados de diferentes maneras como pueden ser el *coseno*, la *inversa de la distancia euclídea* o el *kernel Gaussiano*. Se debe saber que intentar calcular los k vecinos más cercanos de manera ingénua requiere un $T-e$(Tiempo de ejecución) alto. Sobretodo con grandes conjuntos de datos. Aún y todo es un método muy conocido y que hoy en día tiene nuchas alternativas. En nuestro caso, usaremos varias variantes de este método. La diferencia entre una variante y otra está en la manera de calcular la distancia entre dos nodos.

- kNN, kernel Gaussiano.
- kNN, coseno.

3.2.2. *l1*,(Lasso) [4]

Este es un método de regresión penalizada que combina "shrinkagez selección de variables imponiendo una penalización sobre los coeficientes de regresión. En vez de separar los procesos de calculo de la matriz de adyacencias y la matriz de afinidad ponderada, trata de unificarlos en un único proceso. Para ello, cada uno de los ejemplos se codifica como una combinación lineal dispersa del resto de los ejemplos y, las contribuciones del resto de los ejemplos en representar el ejemplo actual, como los peso de los *links*.

Considérese un vector y de dimensión D como parámetro de entrada y la matriz de datos $X,(DxN)$, que contiene N ejemplos.En este método, y será un ejemplo x_i y la matriz X la base de datos en el cual se ha quitado el ejemplo y. El objetivo es representar el ejemplo y como la combinación linear dispersa de la matriz X. Esto, matemáticamente, se representa como

$$b = argminb||y - Xb||_2^2 + \lambda||b||_1$$

donde b es el vector de coeficientes.

Al resolver el problema de minimización, el vector b muestra la contribución de cada ejemplo, esto es, cada columna de X reconstruyendo el parámetro de entrada y. Si el vector b es disperso, muchos de sus elementos son ceros y sólo unos pocos son diferentes de cero. Ejemplos de la base de datos que están lejos de y tendrán coeficientes muy pequeños o iguales a cero. Cuanto más similar el ejemplo respecto a y, mayor será su coeficiente. Así, los vecinos y sus pesos se van calculando de manera simultánea. Una vez calculados los vectores b_i de todos los ejemplos, podemos representar el grafo W como el conjunto de

todos los b_i. El grafo se genera de la siguiente manera:

$$W_{ij} = |bi(j)|$$

3.2.3. Pearson [6]

Este método estábasado en los denominados coeficientes de Pearson, y se calcula de la siguiente manera: Sea p_{ij} el coeficiente de Pearson entre dos ejmplos tomados arbitrariamente, x_i y x_j, que corresponde a la evolución/diferencia en los diferentes atributos de los ejemplosx_i y x_j respectivamente. Estas coeficientes de Pearson se transforman en similitudes que pertenecen al rango $[0,1]$ de la siguiente manera:

$$S_{ij} = (p_{ij} - min(p_{ij}))/(1 - min(p_{ij}))$$

donde $min(p_{ij})$ denota el p_{ij} mínimo de entre las entradas de S_{ij}. Basándonos en estra matriz de similitud, podremos calcular la similitud media de cada ejemplo respecto al resto. Esto se representa por m_i.

Una vez se haya generado la matriz de similitud S_{ij}, se le aplica una pequeña modificación. La matriz S_{ij} se utiliza para calcular una matriz dinámica de adyacencias A_{ij} cumpliendo con el siguiente criterio:

$$A_{ij} = \begin{cases} 1 & \text{si } s_{ij} > m_i \text{ o } s_{ij} > m_j \\ 0 & \text{en el resto de los casos} \end{cases}$$

Esta pequeña modificación también se realiza en el método de generación de grafo kNN, coseno. Esto se hace para los conjuntos de datos más grandes. Con conjuntos de datos muy grandes corremos el peligro que el grafo generado sea muy denso y esto es problemático para la futura detección. Por ello, mediante este mecanismo conseguimos un grafo lo suficientemente disperso como para poder trabajar con ello.

4. CAPÍTULO

Algoritmo de Detección de Comunidades

4.1. Estructura de comunidad en redes complejas

En términos generales, una red es cualquier sistema que admite una representación matemática abstracta como un grafo, donde los nodos identifican los elementos del sistema, y las aristas la relación entre éstos elementos.

La noción de comunidad puede definirse como la presencia de un gran número de aristas conectando nodos dentro de grupos individuales, pero con baja concentración de aristas entre dichos grupos. Pero también, una comunidad puede ser un conjunto de vertices con propiedades similares (e.g. las mismas preferencias en establecer vecinos).

El mejor modo para establecer la estructura de comunidad de una red compleja es un problema todavía sin resolver. Durante los últimos años, se han propuesto muchos algoritmos que persiguen extraer la partición óptima de una red en comunidades. El clustering espectral, los algoritmos de particionamiento de grafos, los métodos basados en la modularidad o los algoritmos basados en la sincronización son sólo algunos de estos algoritmos de extracción de comunidades.

Independientemente de la presencia o no de una estructura de comunidad en una red compleja, los algoritimos de detección de comunidades generalmente devuelven una partición de la red en comunidades. Por lo tanto, es necesario determinar la bondad de la partición obtenida.

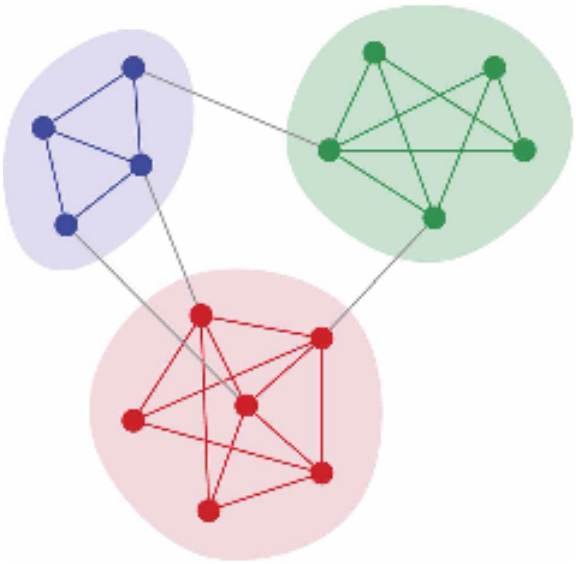

Figura 4.1: Estructura de comunidades en un grafo.

4.1.1. Noción de Modularidad

La modularidad es una de las funciones de calidad más populares generalmente utilizadas para identificar la mejor partición del grafo en comunidades. Fue propuesta por Newman y Girvan en 2004 [2]. La noción de modularidad se basa en la idea de que una distribución por comunidades no es lo que se espera por azar en un grafo, por lo que compara la densidad de links dentro de cada comunidad con la densidad esperada de links en un grafo aleatorio. Está medida nos permite de alguna manera saber cuán acertada ha sido la partición calculada. Valores altos de la modularidad indican buenas particiones con muchas más conexiones internas que las esperadas en uno aleatorio.

Está medida nos permite de alguna manera saber cuán acertada ha sido la partición calculada. Valores altos de la modularidad indican buenas particiones con muchas más conexiones internas que las esperadas aleatoriamente.

4.1.2. La Sincronización y la formación de dinámicas colectivas en redes

La sincronización es el proceso mediante el cual uno o varios sistemas acoplados ajustan sus dinámicas para alcanzar un comportamiento común. Se espera que los osciladores que pertenecen a una misma comunidad tengan la misma dinámica. Inicialmente, para una fuerza de acoplamiento nula, cada nodo del grafo oscila según su propia frecuen-

cia natural. Según vamos incrementando la fuerza de interacción, los osciladores inician un proceso hacia un régimen sincronizado pasando por regímenes de sincronización parcial.

Los estudios llevados a cabo en este campo muestran que para una red compleja con un patrón de conectividad no trivial, aquellas unidades (o sociladores) altamente interconectadas y que forman módulos locales, entran en un régimen de sincronización antes que las demás. Este proceso ocurre a diferentes escalas temporales cuando existe una clara estructura de comunidad en la red.

Por lo tanto, monitorizar el proceso de sincronización de osciladores acoplados permite revelar la estructura de comunidad presente en la red.

4.2. Análisis, Diseño y Desarrollo

En este proyecto tratamos de dotar a los nodos de un grafo, la dinámica necesaria para que el algoritmo desarrollado detecte las comunidades del grafo gracias a la sincronización generada. Se debe subrayar que los resultados de este tipo de algoritmos basados en la sincronización son dependientes tanto de la situación inicial como de la dinámica aplicada al grafo. El algoritmo está basado en uno realizado por el tutor de este proyecto y que se expone en el paper de la siguiente referencia[6].

Para modelar la dinámica de una red compleja dada, se ha usado el conjunto de sistemas acoplados caóticos de Rössler, cada uno de ellos caratcerizado por una frecuencia natural definida. Se eligió este modelo porque ha sido muy estudiado, especialmente en este campo de las redes complejas.

Nuestra propuesta afronta el problema de la estructura de comunidades concentrándose en dos aspectos. El primero es considerar un sistema de osciladores acoplados donde la fuerza de acoplamiento crece linealmente en el tiempo simulando una sociedad donde la interacción entre dos miembros mejora en el tiempo contribuyendo a estabilizar el caracter plural de las diferentes comunidades. Los valores de la fuerza de acoplamiento van desde 0 hasta un valor máximo definido por λ_{max}/λ_2, que representa un estado intermedio de un fuerte acoplamiento entre nodos del mismo cluster. λ_2 y λ_{max} son, respectivamente, el primer autovalor distinto de 0 y el máximo autovalor de la matriz de Laplace. La matriz de Laplace de un grafo es definida por: $L = D - A$, donde D es la matriz diagonal de grados y A es la matriz de adyacencias del grafo G.

El segundo aspecto está relacionado con la influencia que los vecinos ejercen sobre un agente(oscilador) para cambiar su frecuencia natural(opinión), y de ahí, la importancia de una estructura modular en una red, donde cada modulo correspondrá a osciladores con una frecuencia media común. En un pricipio, cuando el acoplamiento es 0, cada oscilador evoluciona de acuerdo a su frecuencia natural, dando pie a estructuras sin correlación. Cuando el acoplamiento va aumentando, van surgiendo nuevos estados de correlaciones correspondientes a osciladores que evolucionan hacia la misma frecuencia media.

4.2.1. Dinámica de la Red

La dinámica de una red G con N osciladores acoplados se puede definir de la siguiente manera:

$$\dot{\xi}_i(t) \quad = \quad F(\xi_i) + \sum_j K_{ij} A_{ij} (\xi_j - \xi_i), \tag{4.1}$$

donde $i = 1, ..., N$. $\dot{\xi}_i(t) = F(\xi_i)$, $\xi_i \in \Re^n$ es la dinámica local de cada oscilador autónomo. Cada oscilador puede ser caracterizada por una frecuencia natural definida. Los osciladores se acoplan con los coeficientes de acoplamiento K_{ij} sobre una topología predefinida caracterizada por una matriz de adyacencias A_{ij}. K_{ij} indica la fuerza de acoplamiento entre los osciladores i y j. Si los osciladores i y j están conectados, tendrán una fuerza de acoplamiento $K_{ij} = K$; si no, la fuerza de acoplamiento es $K_{ij} = 0$.

En este proyecto, investigamos diferentes sistemas acoplados Rössler a través de la componente y:

$$\begin{aligned} \dot{x}_i(t) &= -\omega_i y_i - z_i, \\ \dot{y}_i(t) &= \omega_i x_i + a y_i + \frac{K}{k_i} \sum_j^N A_{ij} (y_j - y_i), \\ \dot{z}_i(t) &= b + (x_i - c) z_i. \end{aligned} \tag{4.2}$$

Todos los osciladores Rössler son diferentes, y el parámtero $\omega_i = \omega_0 - \Delta\omega_i$ se selecciona aleatoriamente desde una distribución uniforme correspondiente a la frecuencia natural de un oscilador individual [Grigory et al. 1997]. $\Delta\omega_i$ es la diferencia de frecuencia entre osciladores vecinos.

Los parámetros iniciales los hemos establecido de la siguiente manera: $a = 0{,}2$, $b =$

0,2, $c = 5{,}7$ y $\omega_0 = 0{,}9$. k_i es el grado del oscilador i, K es la fuerza de acoplamiento y A_{ij} son los valores de la matriz de adyacencia del grafo G.

4.2.2. Adaptación de las frecuencias y Detección de Comunidades

Dada la fuerza de acoplamiento $K = 0$, cada oscilador evoluciona acorde a su propia dinámica dando como resultado espacios de estado no-correlacionados. En el momento en que la fuerza de acoplamiento crece, puede ser que muchos osciladores sincronicen y oscilen en una una frecuencia media común, mientras sus vecinos tienen sus propias, diferentes, frecuencias. Para asegurar la estabilidad de estos estados emergentes, se propone un mecanismo de adaptación dinámico, independiente de parámetros, que modifica la frecuencia característica de los osciladores.

Cada oscilador modifica su frecuencia repetidamente por la frecuencia media de los vecinos, esto es, osciladores cuyas frecuencias se han sincronizado con la suya. Estos vecinos, componen una comunidad. Para definir estas comunidades, el algoritmo hace lo siguiente: durante el proceso de sincronización, se monitoriza la evolución temporal de la componente y de cada oscilador en un intervalo de tiempo predefinido. Luego, calculamos la matriz de similitud S_{ij} de la serie temporal resultante mediante uno de los siguientes métodos de generación de grafo: kNN o *Lasso*. Estos métodos, son dos de los métodos de generación de grafo seleccionados para este proyecto por lo que ya han sido descritos y analizados en este documento.

Calculada la matriz de similitud S_{ij}, que es nuestro grafo W_{ij}, se define P_{ij} como la similitud esperada dentro de cada comunidad, de acuerdo al modelo nulo de Newman (2006)

$$P_{ij} = d_i d_j / m$$

donde d_i denota la similitud total entre el nodo i y el resto de nodos y m es la similitud total de la matriz de similitud S_{ij}. Se calcula B_{ij}, que denota la diferencia entre la similitud del grafo W_{ij} y la similitud esperada P_{ij}:

$$B_{ij} = W_{ij} - P_{ij}$$

Se calculan los autovalores λ_i, y los autovectores v_i de B_{ij} para todos los nodos n. Se recogen los x autovectores (v_i) de B_{ij} correspondientes a los x autovalores(λ_i) positivos. El nº de autovectores(x) que se han recogido determina el nº máximo de clusters que se

van a poder detectar.

Se hace una iteración $j = 1...x$ que abarca el rango entero de posibles grupos desde 2 hasta $x+1$. Se ejecuta el procedimiento de clustering K-means sobre los autovectores recogidos, empezando por $K = j+1$ para encontrar $j+1$ clusters potenciales. Se calcula la modularidad(Q) para ese clustering resultante en cada iteración. La máxima modularidad de entre todas las calculadas determina el mejor clustering[5].

Para terminar,se definen los vecinos,$v(i)$ del ejemplo i, como el conjunto de los ejemplos adyacentes de acuerdo a $S_i j$. Así, la frecuencia del oscilador i es reemplazada por la mediana de las frecuencias de los vecinos. El hecho de usar la mediana en vez de la frecuencia media hace que el método sea más robusto para los *outliers*, los ejemplos que no son vecino de ningún ejemplo, y así se consigue una estructura más clara.

Este mecanismo de adpatación de frecuencias se inicia cuando los osciladores acoplados llegan a cierto grado de sincronización marcado por valores altos de la similitud media m_i. Al principio se considera una primera fase en el cual los osciladores acoplados son forzados a sincronizarse mediante el crecimiento de la fuerza de acoplamiento y acorde a la topología subyacente de la red. Cuando la fuerza de acoplamiento K llega al valor apropiado, [Pastur et al. 2004] por el cual la sincronización total no está establecida y nuevas sincronizaciones pueden surgir, queda fijado. Entonces es cuando empieza el mecanismo de adaptación de frecuencias arriba descrito. Este mecanismo se aplica cada muy poco tiempo(paso de adaptación), durante un intervalo temporal específico. Tanto el tiempo de simulación como el de la adaptación determinan la duración de la serie temporal usada para el cálculo de la matriz de similitud. El nº de veces que se adapatan las frecuencias de los osciladores se define por el ratio entre el intervalo temporal y el paso de adaptación. Esto debería ser limitado para evitar un vector de frecuencias homogeneo.

Este procedimiento de adaptación devuelve un vector de frecuencias unidimensional con cierta estructura de clustering capaz de detectar los módulos subyacentes presentes en la red dada. En cada paso de adaptación se obtiene un nuevo vector de frecuencias y emergen nuevas subdivisiones de comunidad. Para determinar el nº óptimo de comunidades, hemos adoptado el criterio de la modularidad máxima [Newman 2004; Newman et al., 2004], descrita en este documento. La prinicpal razón por la que nos decantamos por este criterio es que es el más ampliamente utilizado en la literatura.

5. CAPÍTULO

Selección de redes reales

En este capítulo se presentan los conjuntos de datos que se han usado en el proyecto. Estos conjuntos se dividen en dos tipos:

- Grafos
- Bases de datos

Sobre los grafos se ha ejecutado directamente el algoritmo de detección de comunidades desarrollado, mientras que con las bases de datos ha habido una fase previa de "preparación de datos"para convertirlos en grafos y así poder acometer la detección de comunidades. Todos los conjuntos de datos, tanto los de tipo grafo como los de tipo base de datos, son redes reales publicados y ampliamente conocidos por la comunidad de investigadores.

5.1. Grafos[7]

5.1.1. zachary

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

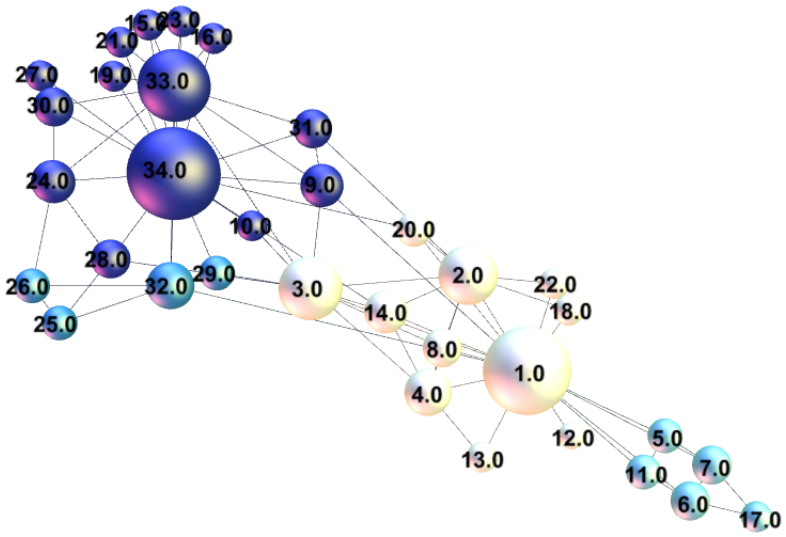

Figura 5.1: Representación del grafo Zachary

5.1.2. dolphins

Esta red fue construída tras un periodo de observación de una comunidad de 62 delfines mulares o de nariz de botella. La relación entre los miembros establece una frecuente asociación, estadísticamente relevante, de los miembros, que fue constatada en la observación. Estas comunidades de delfines mulares han sido descritas como sociedades de fisión-fusión y es por esto que los individuos pueden tomar la decisión de unirse al grupo o dejarlo. La estructura de comunidades de esta red revela que hay 2 comunidades mayoritarias. Una con 21 delfines, y otra que se divide en tres sub-comunidades que suman el resto de delfines.

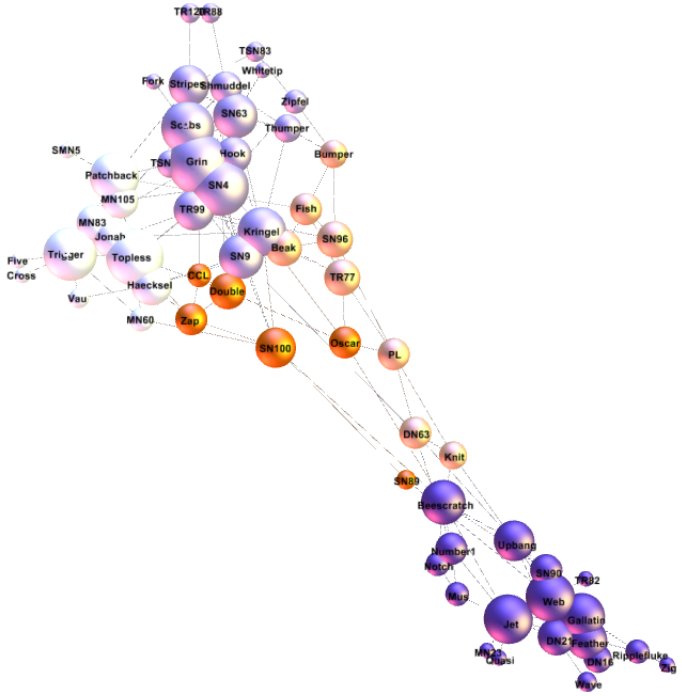

Figura 5.2: Representación del grafo Dolphins

5.1.3. jazz

La red 'jazz' fue creada entre los años 1912 y 1942 y contiene 198 bandas de jazz que actuaron en ese periodo. Dos bandas están conectadas si comparten algún miembro

de la banda. La estructura de comunidades muestra 2 grandes comunidades, de las que una se divide a su vez en otras dos sub-comunidades.

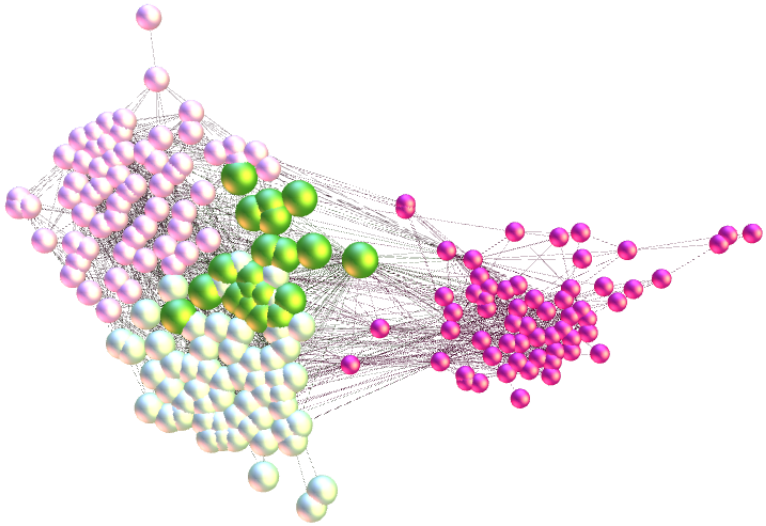

Figura 5.3: Representación del grafo Jazz

5.1.4. football

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

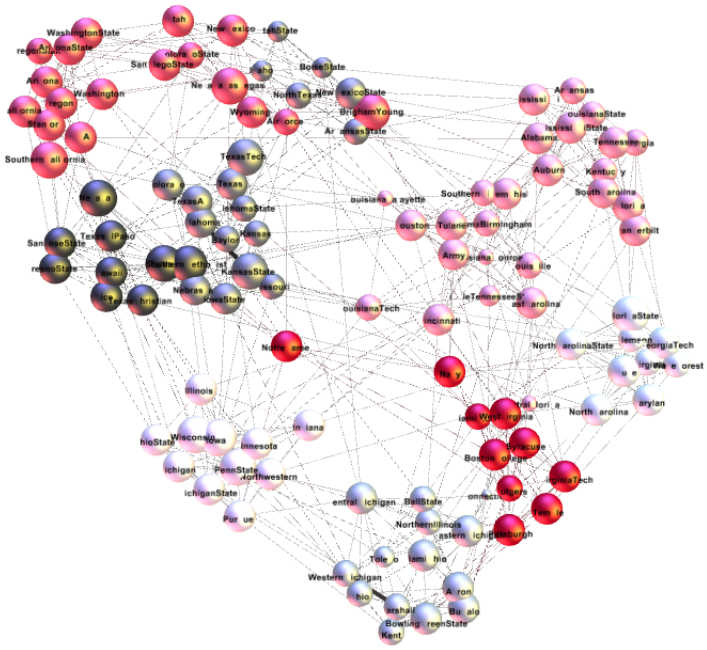

Figura 5.4: Representación del grafo Football

5.2. Bases de Datos[1]

5.2.1. breast tissue

Se utiliza para predecir posibles enfermedades del tejido mamarias. Contiene 106 mediciones de impedencia a diferentes frecuencias de tejido mamarial recientemente extraído. Estas medidicones reflejadas en el plano real/imajinario constituyen el espectro de impedancia en el cual se van a calcular las características del tejido mamarial en cuestión. Originalmente se estructura en 6 clusters aunque se pueden fusionarse tres de ellos para tener una estructura resultante de 4. Esta fusión no resulta relevante en la predicción por lo que no suele hacerse. Cada medición está definida por 9 atributos.

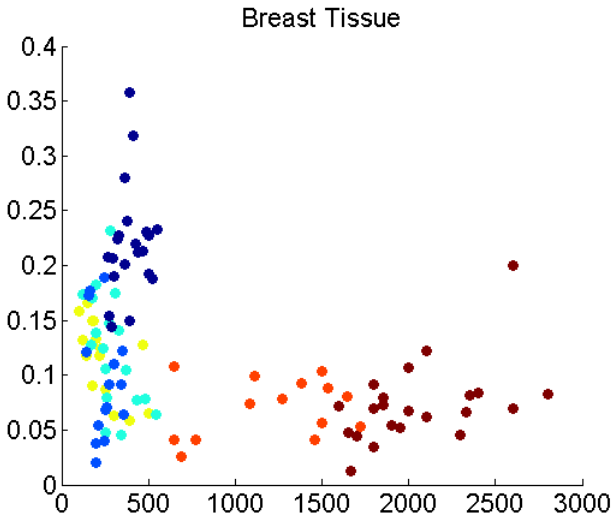

Figura 5.5: Representación de la base de datos Breast Tissue

5.2.2. glass

El desarrollo de esta base de datos fue motivada por una investigación criminológica, ya que unas gafas en la escena del crimen podrían ser usadas como prueba. Clasifica diferentes tipos de cristal para gafas. Contiene 214 ejemplos y están estructuradas en 7 clases. Cada ejemplo se define por 9 atributos.

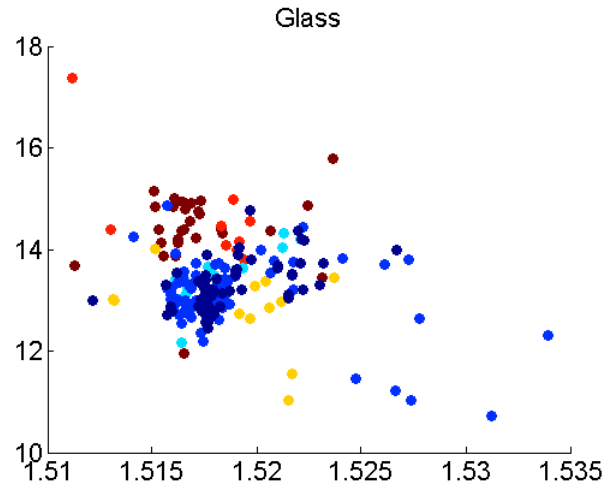

Figura 5.6: Representación de la base de datos Glass

5.2.3. iris

Posiblemente la base de datos más conocida en la literatura de reconocimiento de patrones, un clásico referenciado frecuentemente. Clasifica 3 tipos diferentes de la planta iris. Para ello, contiene 150 ejemplos, cada una de ellas definida por 4 atributos. Una de las comunidades es linealmente separable de las otras dos. Con estas dos últimas no pasa lo mismo.

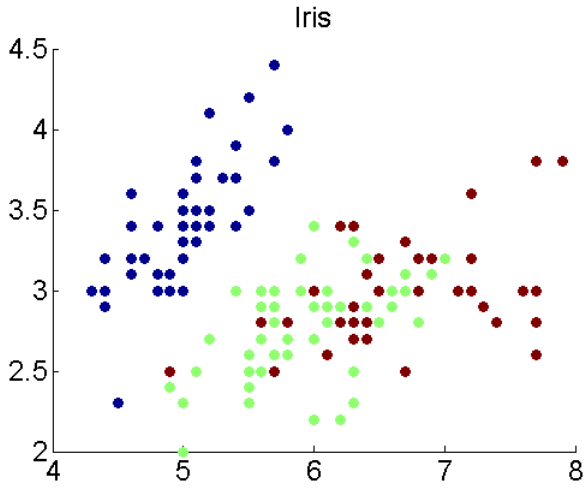

Figura 5.7: Representación de la base de datos Iris

5.2.4. libras movement

Contiene 360 ejemplos distribuidos en 15 comunidades, 24 en cada una de ellas. Esta base de datos clasifica 15 tipos diferentes de movimientos de mano. En el pre-procesamiento de video, la normalización del tiempo se lleva a cabo seleccionando 45 frames por segundo. En cada frame, se buscan los pixeles centrales de la mano, los cuales componen la versión discreta de la curva F con 45 puntos. Todas las curvas se normalizan en el espacio unitario.

Para poder analizar estos movimientos mediante algoritmos se mapeó cada curva F en una representación(ejemplo) de 90 atributos, los cuales representan las coordenadas del movimiento.

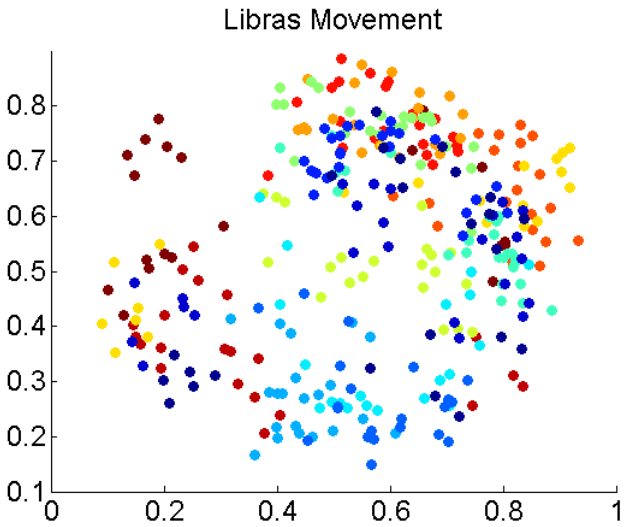

Figura 5.8: Representación de la base de datos Libras Movement

5.2.5. seeds

Contiene 210 ejemplos de semillas de trigo distribuidas en 3 clusters. Cada cluster representa un tipo diferente de trigo: Kama, Rosa y Canadiense. Esta clasificación se desarrollo haciendo uso de rayos-X suaves, el cual es una técnica más sofisticada y menos costosa que otras técnicas de visualización como el escaneo microscópico o la tecnología láser. Esta técnica permite realizar visualizaciones de gran calidad de la estructura interna de la semilla. Cada ejemplo está representada por 7 atributos.

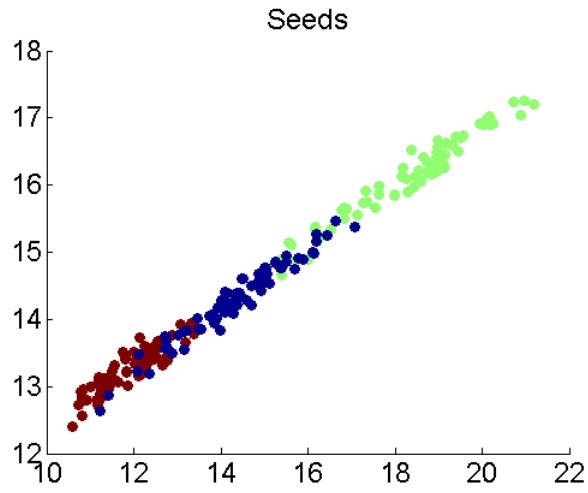

Figura 5.9: Representación de la base de datos Seeds

6. CAPÍTULO

Resultados

En este capítulo se muestran los resultados obtenidos al aplicar el algoritmo de detección desarrollado en redes complejas reales. Cada red compleja está representada bien sea por la matriz de adyacencias A_{ij} o bien por la matriz de similitud S_{ij} y una frecuencia ω_i que representa cada oscilador. Los valores de ω_i se asignan aleatoriamente desde una distribución uniforme por lo que los osciladores Rössler evolucionarán de una fase caótica a un regimen coherente.

Los resultados se han obtenido simulando el sistema durante 225 unidades temporales. La simulación se divide en dos partes. En el primero, en cada unidad de tiempo la fuerza de acoplamiento(K) se aumenta de foma lineal desde un valor inicial variable. El valor de K nunca debe ser lo suficientemente grande como para inducir el sistema a un régimen cercano a la sincronización total. En esta fase, los osciladores acoplados son forzados a una sincronización mutua acorde a la topología de la red y a la fuerza de interacción.

En la segunda fase, K se estabiliza y se comienza con la adaptación de frecuencias. Tal como con la fuerza de acoplamiento, en cada unidad de tiempo se adaptan las frecuencias de los osciladores mediante el mecanismo de adaptación y se guardan los valores de la similitud media (m_i) de cada oscilador para representar la coherencia de la red en ese momento, y el valor de la modularidad (Q) que cuantifica lacalidad de la partición de la red. La partición óptima se determina a partir del valor máximo de Q obtenido en de entre todas iteraciones en la simulación.

En esta sección se hace una comparación de los valores de la modularidad obtenidos

Network	Q_N (N_C)	Q_{max}(N_C)
Zachary	0.4090 **(4)**; 0.4010 (5)	0.4198 **(4)**
Dolphins	0.4580 (4); 0.5190 **(5)**	0.511 **(4)**;0.5099 (4)
Jazz	0.4379 (4); 0.4452 **(5)**	0.4422 (3); 0.4393 (3)
Football	0.5470 (6); 0.5980 **(8)**	0.5595 (6); 0.5605 **(7)**

Tabla 6.1: Comparación de modularidades obtenidas por nuestro algoritmo y el algoritmo de Girvan and Newman [3]. N_C es el nº de comunidades encontrada. Q_N es la modularidad obtenida por el algoritmo de Girvan and Newman. Q_{max} es lamodularidad máxima correspondiente a la partición óptima obtenida con nuestro algoritmo. El nº de comunidades con los valores máximos aparecen destacados.

por nuestro algoritmo con los obtenidos por el ampliamente conocido y usado algoritmo de Girvan and Newman [3]. Esta comparación se hace con las redes complejas con las que hay resultados publicados con dicho algoritmo de Girvan and Newman, esto es, las redes en formato grafo Zachary, Dolphins, Jazz y Football.

En la Tabla1 se muestran los valores de la modularidad correspondientes a las particiones óptimas obtenidas por nuestro algoritmo comparados con los obtenidos con el algoritmo de Girvan and Newman [3]. Por cada red se muestran dos modularidades correspondientes a particiones con diferente nº de comunidades.

A continuación, se analizan, para cada red compleja considerada en este proyecto, los resultados obtenidos por el algoritmo desarrollado. Así mismo, se muestra la partición óptima para cada uno de las redes complejas.

6.1. Redes complejas reales

A-Red compleja Zachary

En la imagen 5.2 se muestra la distribución de la partición óptima de la red compleja Zachary obtenida por nuestro algoritmo. La modularidad máxima obtenida es $Q_{max} =$ 0,4198, más alta que la obtenida por el algoritmo de Girvan and Newman [3]. La partición resultante muestra una división de las dos comunidades iniciales en 4 sub-comunidades. El valor inicial de K se ha establecido en 0.2 y el método de generación de grafo utilizado en la detección ha sido *kNN*.

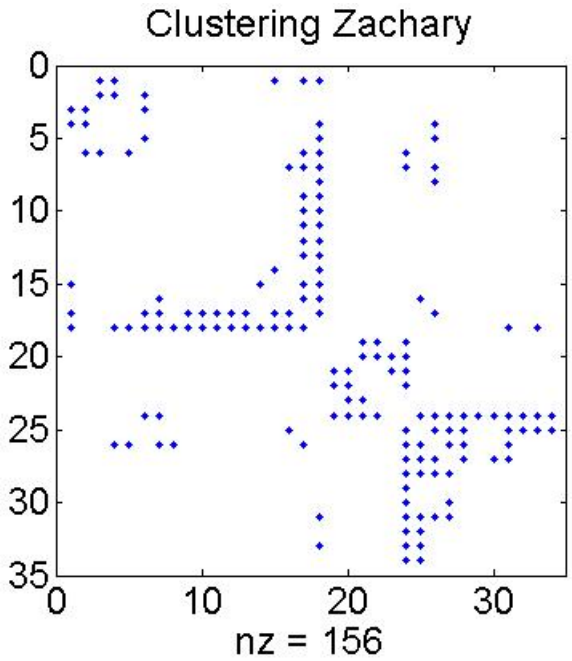

Figura 6.1: Red compleja Zachary. Representación de la partición óptima obtenida con nuestro algoritmo. $Q_{max} = 0{,}4198$

B- Red compleja Dolphins

En las siguientes imagenes se pueden var las dos distribuciones de la particiones óptimas de la red compleja Dolphins obtenidas por nuestro algoritmo. Las modularidades máximas obtenidas son $Q_{max} = 0{,}511$ y $Q_{max} = 0{,}5099$ respectivamente. Ambas particiones representan una distribución de 4 comunidades. Si se comparan con los resultados obtenidos por el algoritmo de Girvan and Newman [3], se puede reslatar dos aspectos. Por una parte, nuestro algoritmo, como reultados óptimos, ha dado dos particiones de 4 comunidades,mientras que en el caso delalgoritmo de Girvan and Newman una partición es de 4 comunidades y la otra es de 5. Por otra parte, en cuanto a las modularidades obtenidas, nuestra algoritmo ha dado dos modularidades mayores que una de las de Girvan and Newman, concretamente la modularidad de la partición de 4 comunidades $Q_z = 0{,}458$, pero inferiores a la modularidad de la partición de 5 comunidades, $Q_z = 0{,}5190$. El valor inicial de K se ha establecido en 0.2 y el método de generación de grafo utilizado en la detección ha sido *Lasso*, con $\lambda = 0{,}2$.

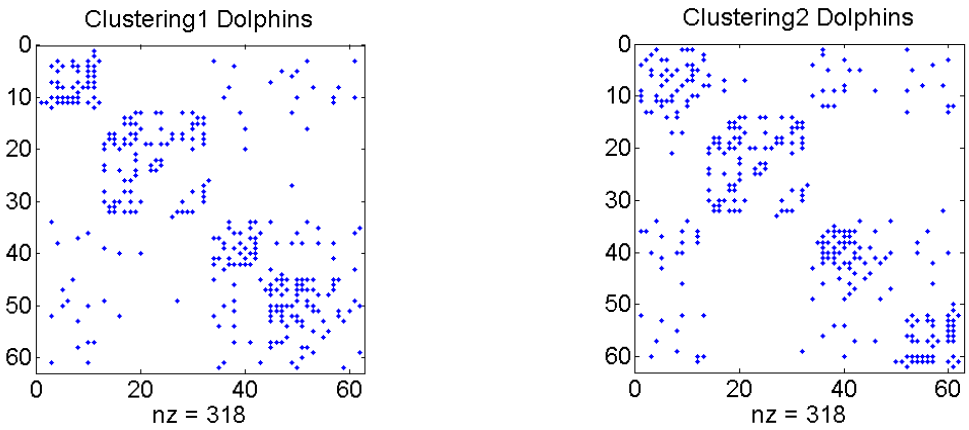

Figura 6.2: Red compleja Dolphins. Representación de las particiones óptimas obtenidas con nuestro algoritmo. En la primera partición $Q_{max} = 0{,}511$; en la segunda $Q_{max} = 0{,}5099$

.

C- Red compleja Jazz

En el caso de la red Jazz, se han obtenido dos partciciones óptimas. Tal y como se puede observar, ambas particiones muestran una estructura de 3 comunidades, ya que una de las dos comunidades iniciales se ha dividio en dos sub-comunidades. Las modularidades máximas obtenidas son $Q_{max} = 0{,}4422$ y $Q_{max} = 0{,}4393$ respectivamente. Mientras las dos particiones obtenidas por nuestro algoritmo son de 3 comunidades, el algoritmo de Girvan and Newman da como resultado dos particiones distintas, una de 4 y otra de 5 comunidades. En cuanto a las modularidades, las obtenidas por nuestro algoritmo son mayores que una de las obtenidas por el algoritmo de Girvan and Newman, en concreto, que la modularidad de la partición de 4 comunidades, que es $Q_z = 0{,}4379$. Sin embargo,son inferiores a la modularidad de 5 comunidades obtenida por el algoritmo de Girvan and Newman, que es $Q_z = 0{,}4452$. El valor inicial de K se ha establecido en 0.6 y el método de generación de grafo utilizado en la detección ha sido *kNN*.

D- Red compleja Football

En el caso de la red Football, las distribuciones de las particiones óptimas obtenidas por nuestro algoritmo muestran unas modularidaes máximas de $Q_{max} = 0{,}5595$ y $Q_{max} = 0{,}5605$ respectivamente. La primera es para una distribución de 6 comunidades mientras que la segunda muestra una estructura de 7 comunidades. En comparación con los resultados obtenidos por el algoritmo de Girvan and Newman, tenemos dos lectu-

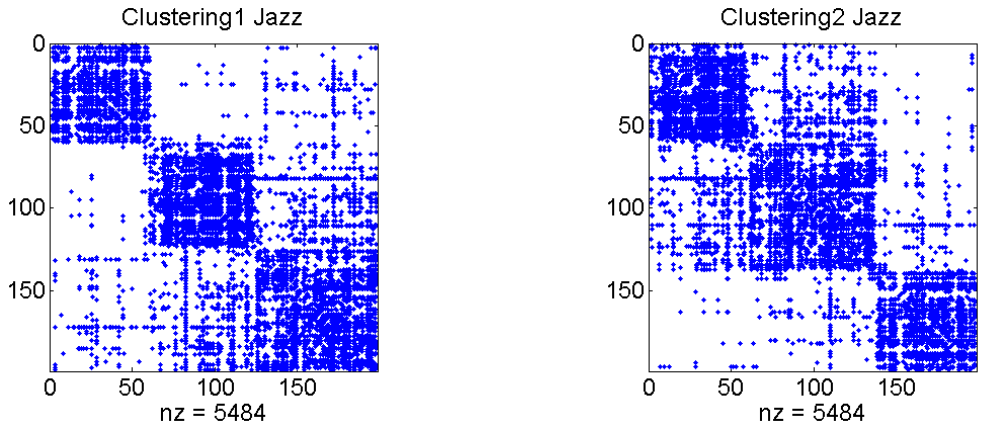

Figura 6.3: Red compleja Jazz. Representación de las particiones óptimas obtenidas con nuestro algoritmo. En la primera partición $Q_{max} = 0{,}4422$; en la segunda $Q_{max} = 0{,}4393$

.

ras. Por una parte, los dos algoritmos dan como uno de los resultados una partición de 6 comunidades, de los cuales el obtenido con el nuestro es mayor que el obtenido por el de Girvan and Newman, $Q_z = 0{,}5470$. Por otra parte, la segunda partición óptima de nustro algoritmo muestra una estructura de 7 comunidades con $Q_{max} = 0{,}5605$, mientras que elalgoritmo de Girvan and Newman obtiene una partición de 8 comunidades con $Q_z = 0{,}5980$. El valor inicial de K se ha establecido en 0.3 y el método de generación de grafo utilizado en la detección ha sido *Lasso*, con $\lambda = 0{,}4$.

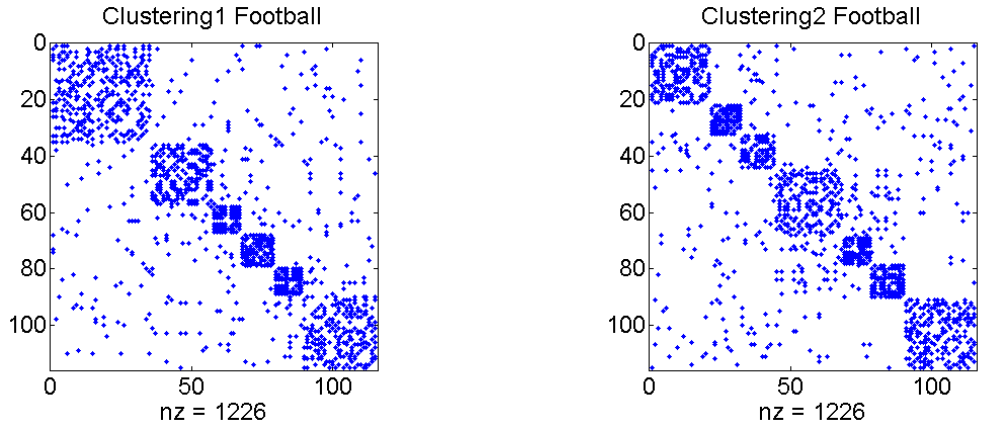

Figura 6.4: Red compleja Football. Representación de las particiones óptimas obtenidas con nuestro algoritmo. En la primera partición $Q_{max} = 0{,}5595$; en la segunda $Q_{max} = 0{,}5605$

.

6.2. Grafos obtenidos de datos en formato registro

A la hora de analizar los resultados que nuestro algoritmo ha obtenido con las redes complejas en formato registro, tenemos que tener en cuenta que primero se debe generar un grafo que represente la red compleja en cuestión. Además, no hemos encontrado resultados publicados de otros algoritmos.

Por todo ello, con el objetivo de valorar de alguna manera la calidad de la partición obtenida por el algoritmo, se ha seguido el siguiente criterio: la modularidad máxima Q_{max} de la partición óptima obtenida debe superar la modularidad del grafo generado Q_G. A este criterio hay que añadir el índice Rand. El índice Rand *RI*, Es un criterio supervisado basado en la similitud. Se define como $(a+b)/M$ donde *a*representa los pares de nodos que pertenecen a la misma clase real y al mismo cluster, *b* representa los pares de nodos que pertenecen a diferentes clases reales y a diferentes clusters y *M* representa el total de los pares de nodos. En caso de que no se cumpla uno de los dos criterios, se da prioridad a maximizar al valor del índice Rand *RI*.

A- Red compleja Breast Tissue

Con la red Breast Tissue, se han obtenido unos resultados óptimos que reflejan unos índices $RI = 0{,}7844$ y $RI = 0{,}77233$ con unas modularidades máximas $Q_{max} = 0{,}7903$ y $Q_{max} = 0{,}78645$ respectivamente. La modularidad del grafo generado es $Q_G = 0{,}251$, por lo que se ha conseguido un espectacular aumento en la modularidad de 212 % y 207 % respectivamente. Ambas particiones muestran una estructura de 7 comunidades. Para generar el grafo, se ha utilizado el método *kNN* con $k = 5$. En cuanto a la configuración del algoritmo de detección, el valor inicial de *K* se ha establecido en 0.2 en ambas detecciones. Una de ellas se ha efectuado con *kNN* y la otra se ha configurado con *Lasso*, $\lambda = 0{,}4$.

B- Red compleja Glass

En el caso de la red Glass, los resultados obtenidos no son muy buenos en cuanto al índice rand(*RI*), pero son muy buenos en cuanto a la modularidad máxima obtenida(Q_{max}). Así, los resultados óptimos reflejan unos índices $RI = 0{,}73095$ y $RI = 0{,}73345$ con unas modularidades máximas $Q_{max} = 0{,}53886$ y $Q_{max} = 0{,}53058$ respectivamente. La modu-

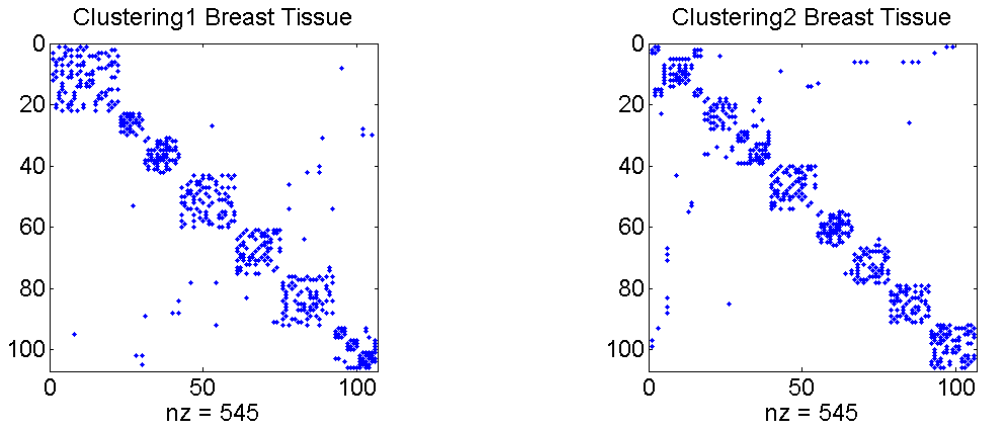

Figura 6.5: Red compleja Breast Tissue. Representación de las particiones que reflejan un mayor índice *RI*. En la primera partición $RI = 0{,}7844$ y $Q_{max} = 0{,}7903$; en la segunda $RI = 0{,}77233$ y $Q_{max} = 0{,}78645$

.

laridad del grafo generado es $Q_G = 0{,}2305$, por lo que se ha conseguido un aumento en la modularidad de 133 % y 130 % respectivamente. La primera partición muestra una estructura de 6 comunidades, mientras que la segunda divide el grafo en 7 counidades. Para generar el grafo, se ha utilizado el método *kNN* con $k = 20$. En cuanto a la configuración del algoritmo de detección, el valor inicial de K se ha establecido en 0.2 en ambas detecciones. Una de ellas se ha efectuado con *kNN* y la otra se ha configurado con *Lasso*, $\lambda = 0{,}4$.

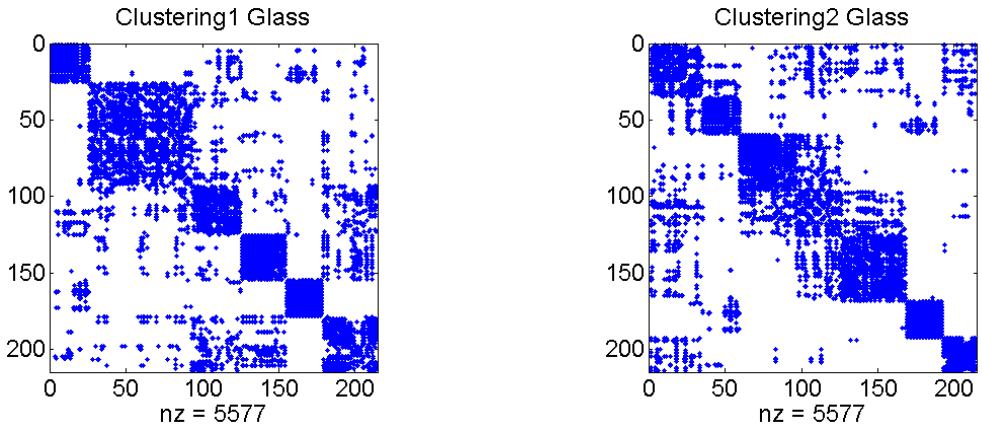

Figura 6.6: Red compleja Glass. Representación de las particiones que reflejan un mayor índice *RI*. En la primera partición $RI = 0{,}73095$ y $Q_{max} = ,53886$; en la segunda $RI = 0{,}73345$ y $Q_{max} = 0{,}53058$

.

C- Red compleja Iris

En cuanto a la red Iris, se han obtenido unos valores del índice Rand(RI) muy altos. Así, los resultados óptimos reflejan unos índices $RI = 0{,}95749$ y $RI = 0{,}96564$ con unas modularidades máximas $Q_{max} = 0{,}62848$ y $Q_{max} = 0{,}62588$ respectivamente. La modularidad del grafo generado es $Q_G = 0{,}60426$, por lo que se ha conseguido un aumento en la modularidad de 4 % y 3,5 % respectivamente. Ambas particiones muestran una estructura de 3 comunidades. Para generar el grafo, se ha utilizado el método *kNN* con $k = 15$. En cuanto a la configuración del algoritmo de detección, el valor inicial de K se ha establecido en 0.2 en ambas detecciones. Una de ellas se ha efectuado con *kNN* y la otra se ha configurado con *Lasso*, $\lambda = 0{,}4$.

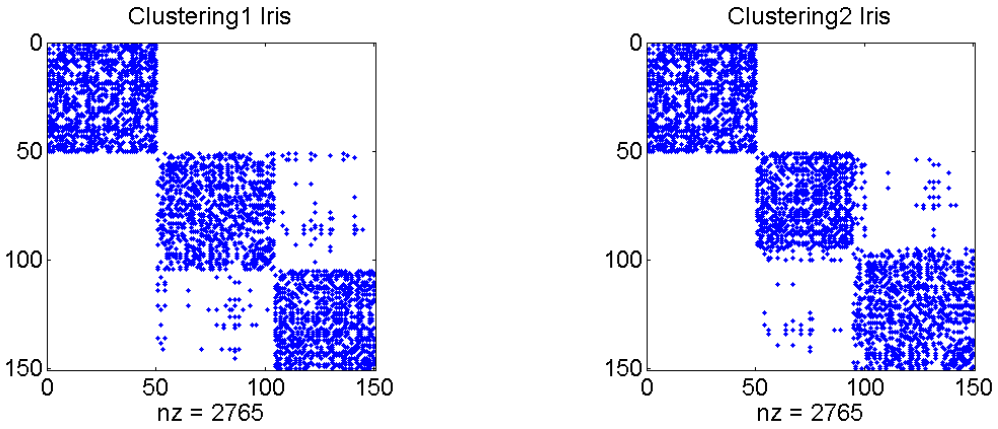

Figura 6.7: Red compleja Iris. Representación de las particiones que reflejan un mayor índice RI. En la primera partición $RI = 0{,}95749$ y $Q_{max} = 0{,}62848$; en la segunda $RI = 0{,}96564$ y $Q_{max} = 0{,}62588$

.

D- Red compleja Libras Movement

En el caso de la red Libras Movement, los resultados obtenidos son buenos. Los resultados óptimos reflejan unos índices $RI = 0{,}86332$ y $RI = 0{,}8504$ con unas modularidades máximas $Q_{max} = 0{,}75172$ y $Q_{max} = 0{,}63149$ respectivamente. Ambas particiones muestran una estructura de 7 comunidades. Para generar el grafo, se ha utilizado el método *kNN* en ambos casos pero con distino valor de k. Esto hace que la modularidad del grafo generado varíe. Así, para el primer caso, con $k = 10$ la modularidad del grafo obtenido es $Q_G = 0{,}4361$, por lo que se ha conseguido un aumento en la modularidad del 72.3 %.

Para el segundo, con $k = 20$, la modularidad del grafo es $Q_G = 0{,}2818$ y el aumento en la modularidad, del 124 %. En cuanto a la configuración del algoritmo de detección, el valor inicial de K se ha establecido en 0.2 en ambas detecciones. Una de ellas se ha efectuado con *kNN* y la otra se ha configurado con *Lasso*, $\lambda = 0{,}4$.

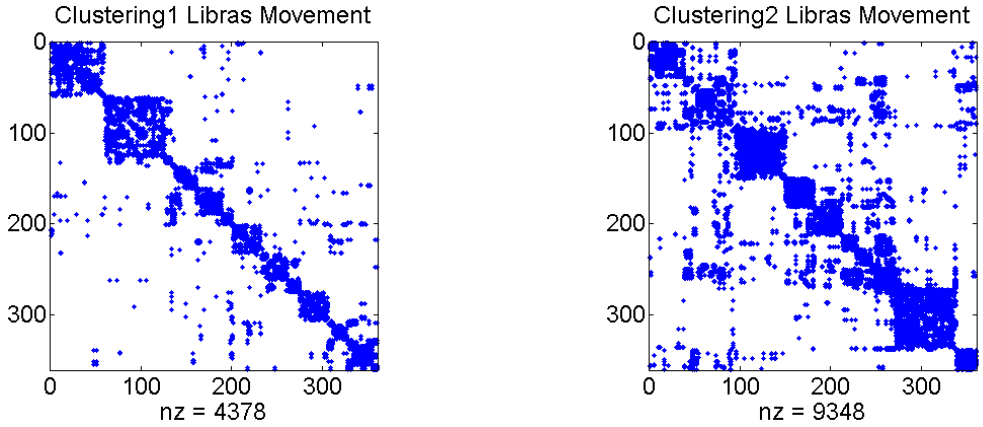

Figura 6.8: Red compleja Libras Movement. Representación de las particiones que reflejan un mayor índice *RI*. En la primera partición $RI = 0{,}86332$ y $Q_{max} = 0{,}75172$; en la segunda $RI = 0{,}8504$ y $Q_{max} = 0{,}63149$

.

E- Red compleja Seeds

En cuanto a la red Seeds, se han obtenido unos resultados óptimos que reflejan unos índices $RI = 0{,}84516$ y $RI = 0{,}81271$ con unas modularidades máximas $Q_{max} = 0{,}57659$ y $Q_{max} = 0{,}61562$ respectivamente. La modularidad del grafo generado es $Q_G = 0{,}5020$, por lo que se ha conseguido un aumento en la modularidad de 14.8 % y 22.6 % respectivamente. Para generar el grafo, se ha utilizado el método *kNN* con $k = 20$. Ambas particiones muestran una estructura de 5 comunidades. En cuanto a la configuración del algoritmo de detección, el valor inicial de K se ha establecido en 0.2 en ambas detecciones. Una de ellas se ha efectuado con *kNN* y la otra se ha configurado con *Lasso*, $\lambda = 0{,}4$.

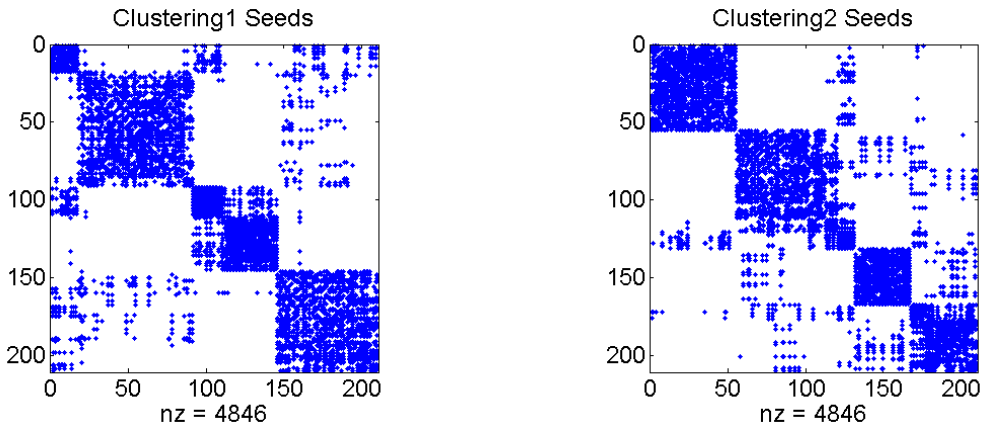

Figura 6.9: Red compleja Seeds. Representación de las particiones que reflejan un mayor índice *RI*. En la primera partición $RI = 0{,}84516$ y $Q_{max} = 0{,}57659$; en la segunda $RI = 0{,}81271$ y $Q_{max} = 0{,}61562$

.

7. CAPÍTULO

Interfaz

La interfaz desarrollada es simple y sencilla. Nos permitirá seleccionar un conjunto de datos, generar un grafo a partir de un conjunto de datos en formato de registros y ejecutar el algoritmo de detección desarrollado. Se podrá elegir entre varios métodos de generación de grafos si fuese necesario. Por otra parte, se podrán elegir varios parámetros del algoritmo de detección, para conseguir resultados a partir de diferentes configuraciones del algoritmo. Para terminar, se podrán visualizar los resultados.

7.1. Diseño

El diseño de la interfaz se divide, para empezar, en dos partes. La parte superior, que se podría denominar como 'preparación de datos' y la inferior, que sería la parte de 'detección'. La parte superior, a su vez, se divide en dos sub-partes. En la parte izquierda se muestra un desplegable donde el usuario selecciona la red que se quiere analizar. Si la red seleccionada está en formato registro, se muestra una vista preliminar de la red y se habilita un pequeño panel de parámetros. En este panel, el usuario selecciona el método de generación con el que quiere conseguir el grafo de la red en cuestión. Si el método requiriese algún parámetro de entrada, el usuario se lo asignaría en este mismo panel. Para generar el grafo, existe un botón llamado 'Generar Grafo'.

En la parte derecha de la preparación de datos, se muestra el grafo generado de la red seleccionada. Si la red en cuestión esta en formato grafo, se muestra aquí directamente al seleccionarlo en el desplegable inicial.

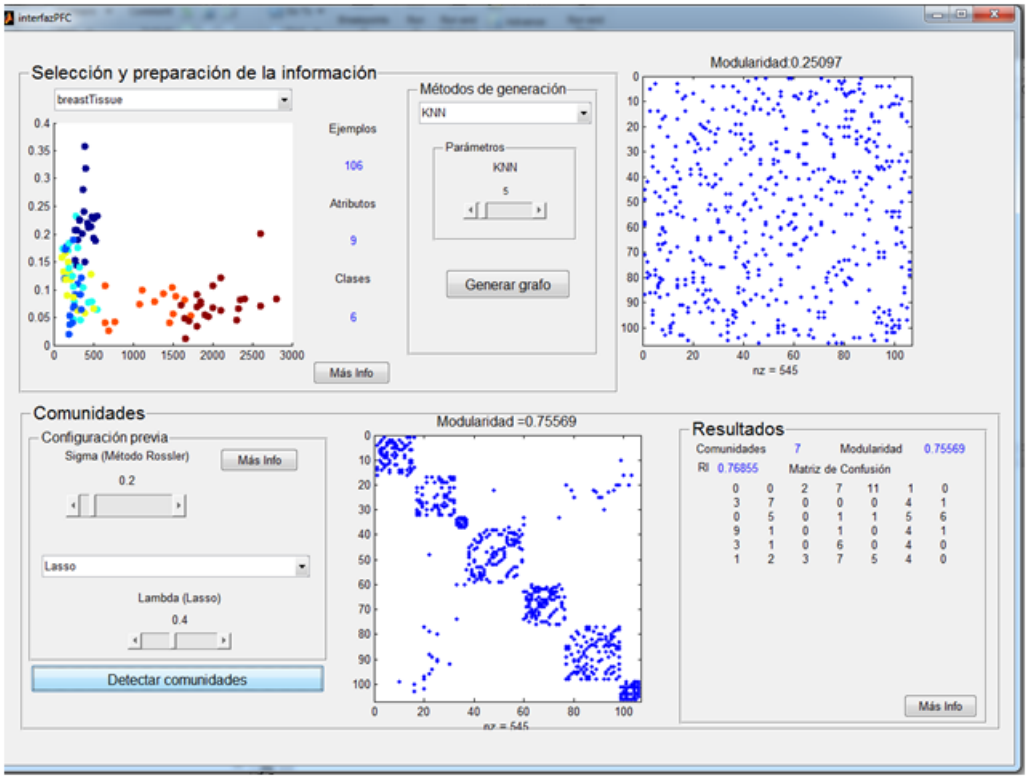

Figura 7.1: Vista general de la interfaz tras la ejecución del algoritmo de detección para 'Breast Tissue'

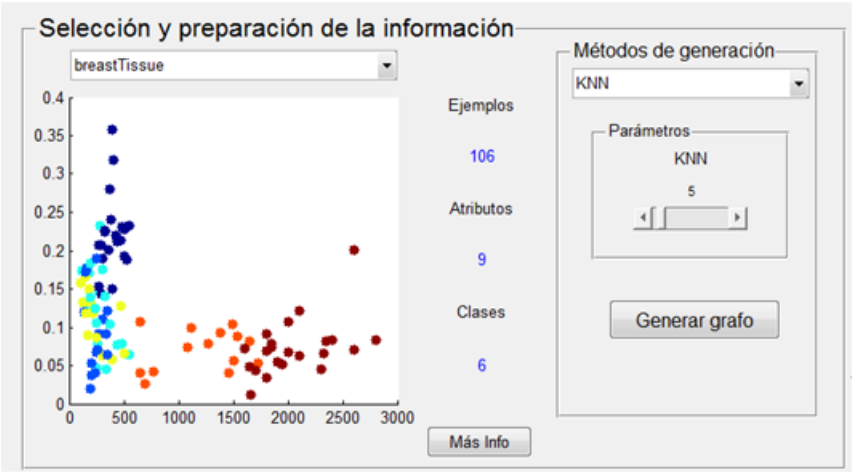

Figura 7.2: Vista de la parte 'preparación de datos' (parte superior-izquierda)de la interfaz

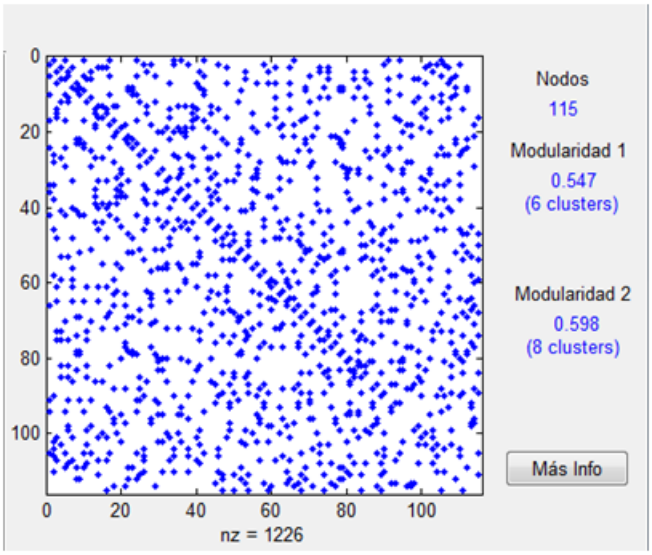

Figura 7.3: Grafo generado a partir de la red selecionada y el método de generación seleccionado

La parte inferior se divide en 2 sub-partes. En la parte izquierda se configuran los parámetros de entrada del algoritmo de detección desarrollado. En la parte central-derecha se muestran los resultados de la detección. Se muestra la partición resultante de la ejecución y los datos respectivos a la partición.

7.2. Funcionalidades

Eldesarrollo de nuestro proyecto se divide en dos tareas bien diferenciadas, por un lado está la traea de generar un grafo que, sólo se ejecuta en el caso de seleccionar una red compleja que está en formato registros. Y la segunda, y principal tarea de nuestro proyecto, es la detección de comunidades de un grafo. Esas dos tareas son las funcionalidades que nuestra interfaz permite ejecutar al usuario. En esta sección, pasamos a explicar qué nos permite hacer la interfaz desarrollada.

- Generación de Grafo: El usuario podrá seleccionar una red compleja desde nuestra base de datos de redes que se muestra en una lista desplegable y, generar un grafo partiendo de la red seleccionada. Para ello, una vez de seleccionar la red, el usuario deberá seleccionar el método que se quiere aplicar en la generación. A su vez, si el método seleccionado lo requiriese, se configurará el parámtero de entrada. Hay que destacar que esta funcionalidad será necesaria sólo en el caso de las redes en formato registros.

Figura 7.4: Panel de configuración de parámetros para la detección de comunidades

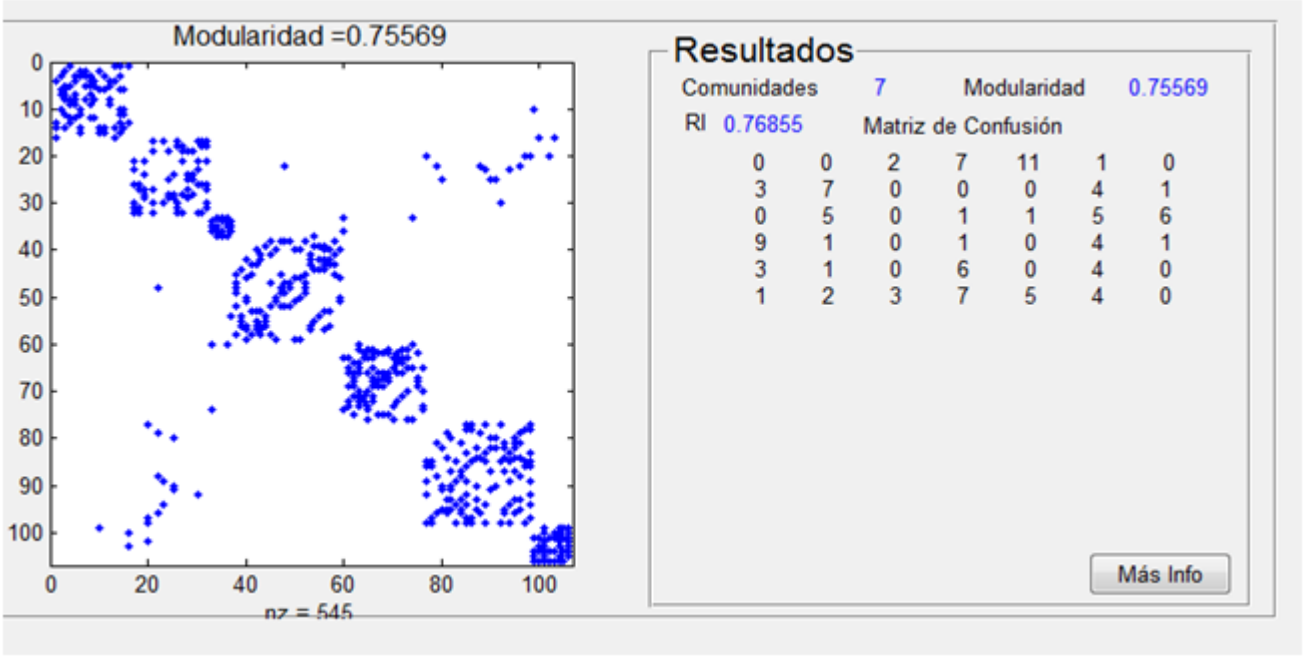

Figura 7.5: Visualización de la partición obtenida y sus datos

- Detección de Comunidades: Esta es ta funcionalidad principal de la interfaz, el objeto del algoritmo desarrollado. Una vez tengamos el grafo, bien porque se ha seleccionado una red en formato grafo, bien porque se ha generado uno a partir de una red en formato registros, la interfaz nos permite, de forma sencilla, ejecutar el algoritmo de detección y visualizar los resultados. Para ello, el usuario deberá configurar un par de parámetros de entrada. Por un lado, el valor inicial de la fuerza de acoplamiento del sistema Rössler (K). Por otro lado, se debe seleccionar el método de generación de grafo que se usará en la ejecución de la detección, que podrán ser *Lasso* y *kNN*, y configurar el valor del parámtero de entrada. Cuando la configuración esté completada, se pulsa el botón 'Detectar Comunidades' y el algoritmo de detección se ejecutará,mostrando los resultados obtenidos.

8. CAPÍTULO

Conclusiones

En este proyecto se ha considerado el problema de la estructura de comunidades tanto en redes complejas reales así como en grafos obtenidos de datos en formato registro, basado en la sincronización de osciladores caóticos. Se ha desarrollado un algoritmo que adapta, de manera dinámica, las frecuencias de los osciladores acoplados con el objetivo de mejorar la estabilidad de los grupos sincronizados emergentes. Este mecanismo de adaptación permite al algoritmo detectar estructuras de comunidades en una red compleja dada. La calidad de las particiones obtenidas en diferentes mediciones, se ha cuantificado mediante el criterio de modularidad máxima. Dichas modularidades obtenidas se han comparado, en el caso de redes reales, con los resultados obtenidos por el algoritmo de Girvan and Newman [3]. En el caso de grafos generados a partir de datos, hemos considerado el índice Rand como criterio para validar los resultados. Se ha desarrollado también, una sencilla interfaz que permite al usuario ejecutar nuestro algoritmo con las diferenetes redes complejas usadas y visualizar los resultados.

Bibliografía

[1] Center for Machine Learning and Intelligent Systems. Machine learning repository. http://archive.ics.uci.edu/ml/.

[2] Santo Fortunato. Community detection in graphs. *Physics Reports*, 486(3–5):75 – 174, 2010.

[3] M. Girvan and M. E. J. Newman. Community structure in social and biological networks. *Proceedings of the National Academy of Sciences*, 99(12):7821–7826, 2002.

[4] Xiaofei He and Partha Niyogi. Locality preserving projections. In S. Thrun, L.K. Saul, and B. Schölkopf, editors, *Advances in Neural Information Processing Systems 16*, pages 153–160. MIT Press, 2004.

[5] MD Humphries. Spike-train communities: finding groups of similar spike trains. *The Journal of neuroscience : the official journal of the Society for Neuroscience*, 31(6):2321—2336, February 2011.

[6] Abdelmalik Moujahid, Alicia d'Anjou, and Blanca Cases. Community structure in real-world networks from a non-parametrical synchronization-based dynamical approach. *Chaos, Solitons and Fractals*, 45(9–10):1171 – 1179, 2012.

[7] Mark Newman. Network data. http://www-personal.umich.edu/~mejn/netdata/.

[8] Partha Pratim Talukdar. Topics in graph construction for semi-supervised learning. *Technical Reports (CIS)*, page 22, 2009.