UNIVERSITAT POLITÈCNICA DE VALÈNCIA

CONSEJO SUPERIOR DE INVESTIGACIONES CIENTÍFICAS

TESIS DOCTORAL:

Detección de comunidades en redes complejas

por

RODRIGO ALDECOA GARCÍA

para la obtención del grado de

DOCTOR EN INFORMÁTICA

Director: DR. IGNACIO MARÍN

Valencia, Abril 2013

Resumen

El uso de las redes para modelar sistemas complejos es creciente en multitud de ámbitos. Son extremadamente útiles para representar interacciones entre genes, relaciones sociales, intercambio de información en Internet o correlaciones entre precios de acciones bursátiles, por nombrar sólo algunos ejemplos. Analizando la estructura de estas redes, comprendiendo cómo interaccionan sus distintos elementos, podremos entender mejor cómo se comporta el sistema en su conjunto. A menudo, los nodos que conforman estas redes tienden a formar grupos altamente conectados. Esta propiedad es conocida como *estructura de comunidades* y esta tesis doctoral se ha centrado en el problema de cómo mejorar su detección y caracterización. Como primer objetivo de este trabajo, se encuentra la generación de métodos eficientes que permitan caracterizar las comunidades de una red y comprender su estructura. Segundo, pretendemos plantear una serie de pruebas donde testar dichos métodos. Por último, sugeriremos una medida estadística que pretende ser capaz de evaluar correctamente la calidad de la estructura de comunidades de una red. Para llevar a cabo dichos objetivos, en primer lugar, se generan una serie de algoritmos capaces de transformar una red en un árbol jerárquico y, a partir de ahí, determinar las comunidades que aparecen en ella. Por otro lado, se ha diseñado un nuevo tipo de *benchmarks* para testar estos y otros algoritmos de detección de comunidades de forma eficiente. Por último, y como parte más importante de este trabajo, se demuestra que la estructura de comunidades de una red puede ser correctamente evaluada utilizando una medida basada en una distribución hipergeométrica. Por tanto, la maximización de este índice, llamado *Surprise*, aparece como la estrategia idónea para obtener la partición en comunidades óptima de una red. *Surprise* ha mostrado un comportamiento excelente en todos los casos analizados, superando cualitativamente a cualquier otro método anterior. De esta manera, aparece como la mejor medida propuesta para este fin y los datos sugieren que podría ser una estrategia óptima para determinar la calidad de la estructura de comunidades en redes complejas.

Resum

L'ús de les xarxes per a modelar sistemes complexos és creixent en multitud d'àmbits. Són extremadament útils per a representar interaccions entre gens, relacions socials, intercanvi d'informació a Internet o correlacions entre preus d'accions borsàries, per anomenar només alguns exemples. Analitzant l'estructura d'aquestes xarxes, comprenent com interaccionen els diferents elements, podrem entendre millor com es comporta el sistema en el seu conjunt. Moltes vegades, els nodes que conformen aquestes xarxes tendeixen a formar grups altament connectats. Aquesta propietat és coneguda com a *estructura de comunitats* i aquesta tesi doctoral s'ha centrat en el problema de com millorar la seva detecció i caracterització. Com a primer objectiu d'aquest treball, tractarem de generar mètodes eficients que permeteixquen caraterizar les comunitats d'una xarxa i comprendre la seua estructura. Segon, pretenem plantejar una sèrie de proves on testar aquests mètodes. Finalment, suggerirem una mesura estadística que pretén ser capaç d'avaluar correctament la qualitat de l'estructura de comunitats d'una xarxa. Per dur a terme aquests objectius, en primer lloc, es generen una sèrie d'algoritmes capaços de transformar una xarxa en un arbre jeràrquic i, a partir d'açí, determinar les comunitats que hi apareixen. D'altra banda, s'ha dissenyat un nou tipus de *benchmarks* per testar aquests i altres algoritmes de detecció de comunitats de forma eficient. Finalment, i com a part més important d'aquest treball, es demostra que l'estructura de comunitats d'una xarxa pot ser correctament avaluada utilitzant una mesura basada en una distribució hipergeomètrica. Per tant, la maximització d'aquest índex, anomenat *Surprise*, apareix com l'estratègia idònia per obtenir la partició en comunitats òptima d'una xarxa. *Surprise* ha mostrat un comportament excel · lent en tots els casos analitzats, superant qualitativament a qualsevol altre mètode anterior. D'aquesta manera, apareix com la millor mesura proposada per a aquest fi i les dades suggereixen que podria ser una estratègia òptima per determinar la qualitat de l'estructura de comunitats en xarxes complexes.

Abstract

Networks have become a widely used tool for modeling complex systems in many different fields. This approach is extremely useful for representing interactions among genes, social relationships, Internet communications or correlations of prices within a stock market, to name just a few examples. By analyzing the structure of these networks and understanding how their different elements interact, we could improve our knowledge of the whole system. Usually, nodes that compose these networks tend to create tightly knit groups. This property, of high interest in many scientific fields, is called *community structure* and improving its detection and characterization is what this thesis is all about. The first objective of this work is the generation of efficient methods able to characterize the communities of a network and to understand its structure. Second, we will try to create a set of tests where such methods can be studied. Finally, we will suggest a statistical measure in order to be able to properly assess the quality of the community structure of a network. To accomplish these objectives, first, we generate a set of algorithms that can transform a network into a hierarchical tree and, from there, to determine their most relevant communities. Furthermore, we have developed a new type of benchmarks for effectively testing these and other community detection algorithms. Finally, and as the most important contribution of this work, it is shown that the community structure of a network can be accurately evaluated using a hypergeometric distribution-based index. Thus, the maximization of this measure, called *Surprise*, appears as the best proposed strategy for detecting the optimal partition into communities of a network. *Surprise* exhibits an excellent behavior in all networks analyzed, qualitatively outperforming any previous method. Thus, it appears as the best measure proposed to this end and the data suggests that it could be an optimal strategy to determine the quality of the community structure of complex networks.

Artículos

En esta tesis doctoral se incluyen los siguientes artículos:

I. Rodrigo Aldecoa & Ignacio Marín
Jerarca: efficient analysis of complex networks using hierarchical clustering.
PloS ONE **5**, e11585 (2010)

II. Rodrigo Aldecoa & Ignacio Marín
Deciphering network community structure by Surprise.
PloS ONE **6**, e24195 (2011)

III. Rodrigo Aldecoa & Ignacio Marín
Closed benchmarks for network community structure characterization.
Physical Review E **85**, 026109 (2012)

IV. Rodrigo Aldecoa & Ignacio Marín
Surprise maximization reveals the community structure of complex networks.
Scientific Reports **3**, 1060 (2013)

V. Rodrigo Aldecoa & Ignacio Marín
Exploring the limits of community detection strategies in complex networks.
Enviado

Agradecimientos

A Ignacio, por enseñarme lo que es la Ciencia, con mayúscula. Por hacer tan fácil y a la vez tan productivo el largo camino de esta tesis. Pocos doctores podrán estar tan agradecidos y orgullosos de su padre científico como lo estoy yo de él.

A mi madre, mi mayor fan. Por leerse e intentar entender cada uno de mis artículos con más ahínco que todos los *referees* juntos.

A mi padre, por ser un ejemplo a seguir. Para que esta tesis le anime a continuar con la suya y ser el próximo doctor de la familia.

A mi hermana Elena, porque la vida es para unos pocos. Y ella sabe estar entre ellos.

A mis padres y a Elena, porque gracias a ellos he llegado hasta aquí. Por estar tan cerca estando tan lejos. Y porque se me están cayendo las lágrimas pensando en lo orgulloso que estoy de los tres.

A Amalia, por confiar tanto en mí y ser mi compañera de viaje en esta nueva etapa que nos espera. Pero sobre todo porque, sin ella darse cuenta, estando a su lado he aprendido algunas de las cosas más importantes de la vida.

Índice general

Capítulo 1

INTRODUCCIÓN

La realidad está formada por sistemas complejos. El Universo, el Sistema Solar, la Tierra, nuestro país, nuestra ciudad, nuestra familia, incluso nosotros mismos contenemos y a la vez formamos parte de sistemas. Cada uno de ellos está compuesto por diferentes elementos que, a su vez, constituyen otros sistemas complejos. Y es la interacción entre estos elementos lo que hace del sistema algo más que la suma de sus partes. Por tanto, para entender nuestra realidad, necesitamos entender cómo la información se codifica y entrelaza para organizar todo tipo de sistemas complejos.

En este contexto, las redes aparecen como un excelente modelo para el estudio y análisis de sistemas biológicos, sociales, económicos o políticos. Es posible representar en forma de red cualquier conjunto de elementos que interaccionan. Formalmente, a estos elementos se les conoce como nodos o vértices y a las interacciones entre ellos como aristas o arcos. Si buscamos detenidamente a nuestro alrededor, es fácil encontrar centenares de este tipo de estructuras. Ciudades conectadas por carreteras, empresas de un mismo sector intercambiando información, individuos compartiendo fotos y mensajes *online*, proteínas de un organismo que interaccionan para dar lugar a nuevas funciones, la interconexión de servidores, *routers* y ordenadores personales que conforman Internet...

La mayoría de estas redes que representan sistemas reales, como veremos más tarde, comparten ciertos patrones y propiedades comunes. Este hecho ha atraído la atención de la comunidad científica, ya que estudiando la información que contienen las redes seremos capaces de aumentar nuestro conocimiento sobre el mundo que nos rodea. Tal interés ha provocado la creación de una disciplina científica propia: el análisis de redes.

1.1. Teoría de grafos

1.1.1. Historia

Aunque hoy en día parece obvio el uso de una red para modelar ciertos tipos de datos, no fue hasta 1736 cuando se planteó y resolvió el primer problema mediante análisis de redes. En aquella época la actual ciudad rusa de Kalilingrado, conocida como Köningsberg hasta su conquista por las tropas soviéticas en 1945, fue testigo y parte del nacimiento de la Teoría de Grafos. El estuario del río Pregel, sobre el que está construida la ciudad, separa sus distintos barrios de forma que sólo se puede acceder de uno a otro mediante puentes. Esto hizo que entre los eruditos de la época surgiese, a modo de juego intelectual, el conocido posteriormente como *Problema de los puentes de Köningsberg*. El planteamiento del problema dice así: *"¿Es posible recorrer todos los barrios de la ciudad pasando una sola vez por cada uno de los puentes volviendo de nuevo al punto de partida?"* El matemático suizo Leonhard Euler consiguió responder negativamente a esta cuestión [Euler 1741]. Para ello recurrió a una abstracción del mapa, representando cada barrio como un nodo y a los puentes que los unen como conexiones entre estos nodos (Figura 1.1).

A partir de este momento clave, el estudio de los grafos y sus propiedades ha sido creciente dentro del ámbito matemático [Bollobás 1998]. Sin embargo, no es hasta principios del S. XX en sociología y hasta más de medio siglo después con la difusión de la informática, cuando el análisis de redes ha pasado a ser un elemento imprescindible en la mayoría de campos científicos. Áreas como la física [Albert y Barabási 2002], la informática [Borge-Holthoefer y Arenas 2010], la biología [Barabási y Oltvai 2004] o la sociología [Wasserman y Faust 1994] han sido beneficiadas por el uso de este nuevo enfoque.

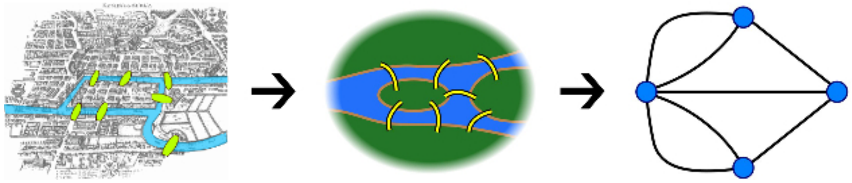


Figura 1.1: Transformación del mapa de Köningsberg en un grafo

1.1.2. Representación matemática

Una red, en su forma más simple, aparece como un grupo de puntos, de los cuales algunos pares se encuentran unidos mediante líneas (Figura 1.2). Dentro del ámbito matemático, este conjunto de nodos y conexiones recibe el nombre de grafo, y se representa formalmente como una tupla $G = (V, E)$, donde V corresponde al conjunto de vértices (nodos) y E es el conjunto de aristas (o conexiones) que relacionan estos nodos.

Una arista conecta dos nodos u y v, y se representa como: $\{u, v\}$. El número de aristas que conectan a un nodo con sus vecinos se conoce como grado del nodo. En el caso de un grafo no dirigido, como el de la Figura 1.2, las aristas son simétricas. Es decir, la arista que conecta u con v es la misma que conecta v con u ($\{u, v\} = \{v, u\}$).

La red social *Facebook* es un claro ejemplo de grafo no dirigido. Una relación de amistad significa que el individuo A es amigo del individuo B y viceversa. Es imposible que A sea amigo de B y B no lo sea de A. Sin embargo, hay redes donde el sentido de la conexión es verdaderamente importante, como ocurre por ejemplo en redes epidemiológicas. Si el individuo A es portador de una enfermedad puede contagiarla a B, que es un individuo sano. En este caso específico, el contagio de B a A no es posible. Este tipo de redes se denominan dirigidas y sus conexiones se representan mediante flechas del nodo origen al nodo destino.

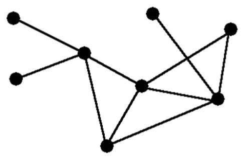


Figura 1.2: Ejemplo de red no dirigida con 8 nodos y 10 aristas.

En la red de la Figura 1.2 todas las conexiones son equivalentes, dos nodos pueden ser o no ser vecinos (presencia o ausencia de una conexión entre ellos). Esta relación puede ser más compleja, por ejemplo, si cada una de las conexiones tiene asociado un valor (o peso), creando lo que se conoce como red ponderada. En una red de transporte, una conexión entre dos nodos refleja que existe una carretera entre esas dos ciudades. Pero además, esa conexión puede contener más información como, por ejemplo, la distancia en kilómetros que separa a una de la otra. En este caso, el número de kilómetros será el peso de esa conexión.

1.1.3. Redes complejas

En 1959, Paul Erdős y Alfréd Rényi [Erdős y Rényi 1959] realizan el primer gran trabajo matemático sobre grafos. En él, proponen un generador de grafos aleatorios en el cual cada posible conexión de la red aparece con una probabilidad fija p, independientemente de la configuración del resto de conexiones. Esto es lo mismo que decir que cada par de nodos tiene una probabilidad de conexión p. Siguiendo este principio de generación, los grados de los nodos del grafo resultante siguen una distribución de Poisson. Sin embargo, cuando se empezaron a representar los primeros sistemas reales en forma de red, los científicos pronto se dieron cuenta de que la estructura de estas redes distaba mucho de la esperada para un grafo aleatorio [Watts y Strogatz 1998, Barabási y Albert 1999, Strogatz 2001]. Por ello, a estas redes reales se las conoce como *"complejas"*. En concreto, la distribución de grados de los nodos a menudo no se asemeja a una Poisson, sino que muestra una larga cola característica de distribuciones de ley de potencia, Zipf o Pareto [Barabási y Albert 1999, Newman 2005]. Esto quiere decir que, mientras unos pocos nodos de la red están altamente conectados con otros nodos, la mayoría de ellos no tienen apenas conexiones. A medida que se profundizaba más en el estudio de las redes complejas, se fueron encontrando interesantes propiedades. Entre ellas destacan su alto coeficiente de *clustering* [Strogatz 2001], alta asortatividad (los nodos tienden a estar conectados a otros nodos de grado similar) [Croft et al. 2005] y, muchas veces, estructura jerárquica [Ravasz y Barabási 2003]. Pero además, en este tipo de redes aparecen grupos de nodos muy conectados entre ellos (Figura 1.3). A esta propiedad se la conoce como estructura de comunidades y es el objeto de estudio de esta tesis doctoral.

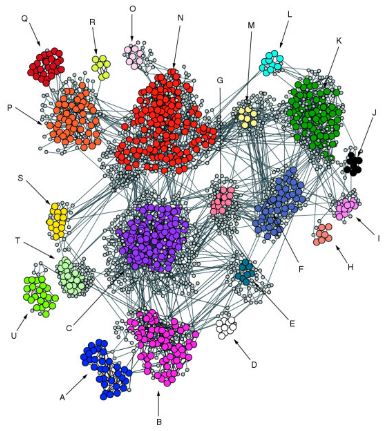


Figura 1.3: Red de co-expresión génica de *Saccharomyces cerevisiae*. Dos genes aparecen conectados si sus patrones de expresión son similares. Coloreando los nodos de la red según la función conocida de cada gen, las zonas densamente conectadas corresponden a grupos funcionales de genes [Magwene et al. 2004].

1.2. Estructura de comunidades

En redes complejas, los nodos relacionados tienden a formar grupos densamente conectados. Además, dichos grupos se encuentran poco conectados con los demás grupos que conforman el resto de la red. A estas formaciones se les denomina comunidades y su caracterización y análisis es de gran interés en diferentes campos científicos [Fortunato 2010]. Dependiendo del tipo de datos, las comunidades pueden representar complejos proteicos en redes de interacción proteína-proteína [Spirin y Mirny 2003], círculos de amigos en redes sociales [Traud et al. 2011], zonas con mayor riesgo de terremotos [Abe y Suzuki 2012] o, como en la Figura 1.3, grupos de genes relacionados funcionalmente.

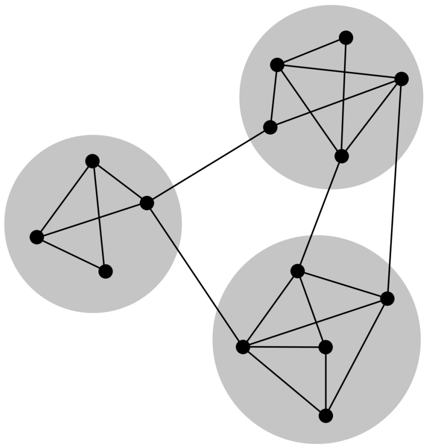


Figura 1.4: Partición de una red en tres comunidades. Los nodos de cada comunidad están más densamente conectados entre ellos que con nodos de otras comunidades (Fuente: Wikipedia Commons).

Una partición es la división de una red en comunidades, de modo que todo nodo pertenece a alguna de ellas (Figura 1.4). Cómo encontrar la partición óptima de una red es uno de los problemas abiertos más importantes en el campo de la Teoría de Redes. El principal obstáculo para obtener dicha partición es la falta de una definición formal de lo que es una comunidad [Radicchi et al. 2004, Fortunato 2010].

1.2.1. *Clustering* jerárquico iterativo

Se han propuesto innumerables métodos para detectar la partición óptima de una red, cada uno con su propio concepto de qué es una comunidad. Las estrategias utilizadas para optimizar las funciones objetivo de los distintos algoritmos son numerosas: *clustering jerárquico* [Arnau et al. 2005, Blondel et al. 2008, Aldecoa y Marín 2010], *simulated annealing* [Duch y Arenas 2005], compresión de la información de la red [Rosvall y Bergstrom 2008], heurísticas multiresolución [Reichardt y Bornholdt 2006, Ronhovde y Nussinov 2009, Traag et al. 2011], *random walks* [Pons y Latapy 2005, Weinan et al. 2008], métodos espectrales [Shen y Cheng 2010] o algoritmos genéticos [Shi et al. 2009].

De entre las diferentes estrategias cabe destacar, ya que será la base de la primera parte de esta tesis, el *clustering jerárquico iterativo* [Arnau et al. 2005]. El objetivo de este método desarrollado por nuestro grupo es resolver un problema habitual en *clustering*, el cual aparece al intentar agrupar elementos entre cuyas distancias existen un gran número de empates, como ocurre por ejemplo en grafos no ponderados. En este tipo de redes, dado que todas las conexiones son idénticas, el rango de valores distintos de las distancias mínimas entre nodos (si contamos cada conexión como un paso entre dos de ellos) es muy reducido. Esto produce que, al intentar agrupar jerárquicamente estos nodos, nos encontremos con un gran número de alternativas posibles en cada paso. De este modo, no existe manera de saber cuál de ellas será más efectiva para construir el árbol que más se acerca a la topología real de la red.

El proceso de *clustering jerárquico iterativo* consiste en realizar un gran número de veces un simple y rápido algoritmo de *clustering* sobre la red a estudiar para generar multitud de soluciones. Posteriormente, a partir de todas esas soluciones simples, se calcula una matriz de distancias promedio entre cada par de nodos. De este modo pasamos de una matriz de adyacencia, donde sólo existen valores 0 y 1, a una matriz con muy pocos empates o ninguno. Una vez obtenida dicha matriz, se aplica un algoritmo de *clustering* jerárquico convencional para producir un dendrograma que representa la topología de la red. Cada nivel de este árbol representa una partición en comunidades distinta y de entre ellas se selecciona como óptima aquella que obtiene mejor puntuación al ser evaluada mediante una función de calidad (la cual discutiremos más adelante) [Arnau et al. 2005]. Esta estrategia fue implementada en un programa llamado UVCluster y aplicada con éxito al análisis de redes de proteínas en *Saccharomyces cerevisiae* [Marco y Marín 2007, Marco y Marín 2009] y dominios proteicos en humanos [Lucas et al. 2006].

1.2.2. Comparación de algoritmos en *benchmarks* sintéticos

Cada uno de los algoritmos de detección de comunidades contiene, implícitamente, su propio concepto de qué es una comunidad. Al no perseguir el mismo objetivo, estos métodos intentan maximizar o minimizar diferentes parámetros y devuelven distintas soluciones, haciendo complicado el comparar su eficiencia directamente.

Habitualmente, esta comparación entre algoritmos se realiza en bancos de pruebas (de aquí en adelante llamados *benchmarks*) sintéticos [Danon et al. 2005, Lancichinetti y Fortunato 2009, Orman y Labatut 2009]. Estos tests comienzan con una red cuya estructura de comunidades está bien definida y es conocida *a priori*. A continuación, las conexiones de la red se van aleatorizando, de modo que las comunidades iniciales son cada vez más difíciles de reconocer por los distintos algoritmos. El primer *benchmark* utilizado en detección de comunidades fue propuesto por Girvan y Newman (GN) [Girvan y Newman 2002]. Está compuesto por cuatro comunidades de 32 nodos cada una y cada nodo se encuentra conectado con 16 vecinos, al principio todos ellos de su misma comunidad. Esta red inicial se va modificando de modo que las conexiones van desapareciendo entre nodos de una misma comunidad y aparecen conectando nodos de distintas comunidades. De esta manera la densidad intracomunitaria de la red inicial cada vez es menor y las comunidades acaban por desaparecer.

Sin embargo, estos *benchmarks* no reflejan apropiadamente la mayoría de propiedades que exhiben las redes del mundo real. En un intento de emular mejor estas redes reales, Lancichinetti, Fortunato y Radicchi (LFR) propusieron un nuevo tipo de *benchmark* donde tanto el tamaño de las comunidades como la distribución de grados de los nodos pueden ajustarse a una ley de potencia [Lancichinetti et al. 2008]. Además, podemos variar un parámetro μ, que indica la fracción de conexiones que unen cada nodo con nodos de otras comunidades, de modo que conforme crece μ las comunidades van degradándose progresivamente.

La generación de redes LFR es útil para comparar algoritmos debido a las diversas variables que el usuario puede controlar (distribución de grados de los nodos, de los tamaños de comunidad, grado mínimo, máximo y medio de los nodos, tamaño máximo y mínimo de las comunidades). A pesar de ello, en ciertas ocasiones, no es posible satisfacer los valores de todos estos parámetros al mismo tiempo, ya que cada uno de ellos impone ciertas constricciones a la red. Por ejemplo, no se pueden generar redes con tamaños de comunidades extremos. Tampoco es posible, en el caso de que se desee, elegir *a priori* el tamaño concreto de cada comunidad.

Algunas de estas limitaciones se pueden solucionar utilizando un *benchmark* similar conocido como *Relaxed Caveman* (RC) [Watts 2003], el cual comienza con *cliques* aislados (subgrafos completos sin conexiones entre ellos) que representan las comunidades iniciales. Los tamaños de estas comunidades son definidos por el usuario y al progresar el *benchmark* las conexiones dejan de unir nodos de una misma comunidad y pasan a conectar comunidades distintas. Así, se consigue un proceso de degradación similar al de los *benchmarks* GN y LFR. La principal ventaja de un *benchmark* RC es que podemos decidir exactamente el número de nodos de cada comunidad y, por tanto, generar redes donde coexistan comunidades de tamaños muy distintos.

Los *benchmarks* descritos han sido ampliamente utilizados para comparar el comportamiento de distintos algoritmos. Sin embargo, sabemos que a partir de una determinada degradación de las comunidades, la partición inicial dejará de ser la óptima, debido a que la mayoría de las conexiones que unían nodos de una misma comunidad, ahora se encuentran uniendo nodos de distintas comunidades. Por tanto, en casos donde la degradación de la red sea alta y un algoritmo devuelva una solución que no se corresponde con la partición inicial, no podemos saber si es porque el algoritmo no es capaz de reconocerla o porque sí que está detectando la partición óptima pero ésta ya no se corresponde con la inicial. La causa principal de esta limitación es que el barajeo de las conexiones es completamente aleatorio y, por tanto, la red evoluciona hacia una estructura final desconocida, tienen un final "abierto". Debido a ello hemos decidido denominar *open* a este tipo de *benchmarks*. Por contraposición, se pueden generar *benchmarks* cuyo final se define *a priori*. A estos *benchmarks* donde las estructuras inicial y final son conocidas, los hemos llamado cerrados (*closed*).

1.2.3. Evaluación de la estructura de comunidades

Aunque los análisis en *benchmarks* sintéticos son útiles para comparar la calidad relativa de los algoritmos, cuando tratamos con redes reales no tenemos información de cómo o cuántas son las comunidades que las componen. Por tanto, no tenemos ninguna referencia con la que comparar los resultados que obtenemos por parte de cada método. Idealmente, desearíamos tener una medida independiente que, simplemente teniendo en cuenta la configuración de la red, asigne un valor a una partición en función de lo bien definidas que se encuentran las comunidades observadas.

En 2004, Newman y Girvan propusieron un índice para evaluar la calidad de la estructura de comunidades de una red, al que llamaron *Modularity* Q [Newman y Girvan 2004]. Q compara la densidad de conexiones de una comunidad con la que esperaríamos dada la distribución observada de grados de los nodos. De este modo pretende calcular la calidad de la comunidad, es decir, cómo de bien definida se encuentra. A continuación, se suman las calidades de todas las comunidades y se obtiene el valor global Q. Dada una partición en comunidades de una red, su valor de *Modularity* se puede calcular de la siguiente manera:

$$Q = \frac{1}{2m} \sum_{ij} (A_{ij} - \frac{k_i k_j}{2m})\, \delta(c_i, c_j) \tag{1.1}$$

donde A_{ij} vale 1 si i y j están conectados, siendo 0 en caso contrario. $(k_i k_j)/(2m)$ es el número de conexiones esperadas por azar teniendo en cuenta los grados k de los nodos i y j. Y, por último, $\delta(c_i, c_j)$ vale 1 si los nodos i y j pertenecen a la misma comunidad y 0 en caso contrario.

Esta medida ha sido utilizada en centenares de artículos científicos y es muy popular. Sin embargo, ya en 2007 se demostró que, bajo ciertas circunstancias, siguiendo esta estrategia no se pueden detectar comunidades más pequeñas de un cierto umbral, el cual viene determinado por el tamaño de la red y su patrón de conexiones [Fortunato y Barthelemy 2007, Kumpula et al. 2007]. Se ha tratado de solventar este problema de varias maneras, principalmente con algoritmos *multiresolución*. Estos métodos, basados en modelos de Potts (que pueden ser vistos como una generalización de Q [Reichardt y Bornholdt 2006]), utilizan un parámetro para explorar la red a distintos niveles de resolución, buscando tanto las comunidades pequeñas como las de gran tamaño [Reichardt y Bornholdt 2006, Ronhovde y Nussinov 2009, Traag et al. 2011]. Sin embargo, trabajos recientes han demostrado que estas estrategias son incapaces de detectar, en una misma partición, comunidades de tamaños muy distintos [Lancichinetti y Fortunato 2011, Xiang y Hu 2012].

Poco después de la aparición de Q, en el artículo de nuestro grupo comentado anteriormente [Arnau et al. 2005], se propone una forma alternativa de evaluar la calidad de una partición en comunidades. El problema a resolver en ese trabajo era cómo elegir el nivel de un dendrograma obtenido mediante *clustering* jerárquico que indicase la mejor partición, aunque era obvio que, si dicho criterio funciona, puede extenderse a otro tipo de particiones. La idea que se exploró se basa en comparar, de forma global, el número de conexiones observadas dentro y fuera de las comunidades con las esperadas si las conexiones apareciesen en la red de forma completamente aleatoria. La medida asume como modelo nulo subyacente una generación del grafo aleatoria Erdős-Rényi [Erdős y Rényi 1959], en la cual la probabilidad de aparición de una conexión es la misma para cada par de nodos. La probabilidad de esa distribución de conexiones observada puede ser exactamente calculada utilizando una distribución hipergeométrica [Arnau et al. 2005]:

$$I = \sum_{j=p}^{min(M,n)} \frac{\binom{M}{j}\binom{F-M}{n-j}}{\binom{F}{n}} \tag{1.2}$$

donde n es el número de conexiones observadas, de un máximo posible F ($F = \frac{k(k-1)}{2}$), siendo k el número de nodos de la red. El número máximo de conexiones posibles en la partición observada es M. Y, de ellas, sólo existen p (número de conexiones intracomunitarias observadas).

Como veremos, esta medida de la calidad de una partición, o más bien su ligera modificación $S = -\log I$, donde S es el parámetro que llamaremos *Surprise*, ha sido explorada en este trabajo.

1.3. Objetivos

El objetivo último de esta tesis doctoral es el diseño de procedimientos para mejorar la caracterización de la estructura de comunidades en redes complejas. Los objetivos son tres:

1. Generar una serie de algoritmos que permitan, de manera eficiente, extraer de una red las diferentes comunidades que la componen. Para ello nos basaremos en el concepto de *clustering jerárquico iterativo* citado anteriormente [Arnau et al. 2005]. Además de extraer la estructura de comunidades, esta estrategia permite transformar el grafo en un árbol jerárquico, obteniendo valiosa información sobre la topología de la red.

2. Por otro lado, dada la cantidad de algoritmos de detección de comunidades que existen en la literatura, es necesario disponer de una serie de pruebas estándar o *benchmarks* que nos permitan compararlos y clasificarlos para seleccionar los mejores de ellos. Dado que los análisis en los *benchmarks open* tradicionalmente utilizados, como se ha comentado previamente, nos impiden conocer si una solución dada es óptima o no, desarrollaremos un nuevo tipo de tests que intente eliminar esta y otras limitaciones.

3. Finalmente, se pretende encontrar un índice matemático que permita evaluar de forma correcta la calidad de la estructura de comunidades de una red. Exploraremos la idea presentada en [Arnau et al. 2005] de evaluar la

calidad de una partición mediante una distribución hipergeométrica, a fin de establecer si es una medida fiable de la calidad de una partición y, por tanto, si su maximización sería una buena estrategia para encontrar la estructura de comunidades óptima de una red.

Capítulo 2

ARTÍCULOS

Jerarca: Efficient analysis of complex networks using hierarchical clustering

Rodrigo Aldecoa and Ignacio Marín
Instituto de Biomedicina de Valencia. Consejo Superior de Investigaciones Científicas (IBV-CSIC) Calle Jaime Roig 11. Valencia, Spain

PLoS ONE 5, e11585 (2010)

Background: How to extract useful information from complex biological networks is a major goal in many fields, especially in genomics and proteomics. We have shown in several works that iterative hierarchical clustering, as implemented in the UVCluster program, is a powerful tool to analyze many of those networks. However, the amount of computation time required to perform UVCluster analyses imposed significant limitations to its use.

Methodology/Principal Findings: We describe the suite Jerarca, designed to efficiently convert networks of interacting units into dendrograms by means of iterative hierarchical clustering. Jerarca is divided into three main sections. First, weighted distances among units are computed using up to three different approaches: a more efficient version of UVCluster and two new, related algorithms called RCluster and SCluster. Second, Jerarca builds dendrograms based on those distances, using well-known phylogenetic algorithms, such as UPGMA or Neighbor-Joining. Finally, Jerarca provides optimal partitions of the trees using statistical criteria based on the distribution of intra- and intercluster connections. Outputs compatible with the phylogenetic software MEGA and the Cytoscape package are generated, allowing the results to be easily visualized.

Conclusions/Significance: The four main advantages of Jerarca respect to UVCluster are: 1) Improved speed of a novel UVCluster algorithm; 2) Additional, alternative strategies to perform iterative hierarchical clustering; 3) Automatic evaluation of the hierarchical trees to obtain optimal partitions; and, 4) Outputs compatible with popular software such as MEGA and Cytoscape.

Introduction

There are many types of data, both biological and non-biological, which can be represented as undirected graphs. Examples in biology are networks based on protein-protein interaction data, those based on shared protein domains, genetic interaction networks or coexpression networks. Developing heuristic strategies to extract useful information from them is an active field of research (reviewed in [1–3]). A typical problem is how to generate partitions of a network in order to establish clusters, groups of tightly connected units. There are two basic general strategies to perform such a task. One option is to search for densely connected modules, for instance using a local evaluation function that measures when adding or eliminating units leads to a significant decrease of the average density of connections within a group (see e. g. refs. [4–9]). A second possibility is to generate complete partitions of the graph, assigning each unit to a cluster. This requires global parameters to evaluate the quality of the alternative partitions [10–12]. Although both methods have advantages and drawbacks, the latter should be considered preferable on theoretical grounds, given that it allows classifying all the units of the network.

To classify data, hierarchical clustering has several advantages over other procedures. First, it is a fully unsupervised method. In the case of networks, this allows to cluster all units without having to specify a priori the number of clusters present. In addition, the generation of a hierarchical tree provides not only partitions of the network (either by how units are grouped in agglomerative clustering, or by how the units are divided into groups, in divisive clustering), but also allows to visualize how the basic, first-order clusters are combined into higher-level groups. However, the development of hierarchical clustering strategies to analyze networks is problematic. Particularly, clustering unweighted undirected graphs (e. g. networks of interacting units) is seriously hampered by the "ties in proximity"problem (discussed in [12]). In this type of networks, the distance between two units is defined as the minimal number of edges that must be walked to connect them. Then, in typical biological networks – large and with small-world properties – the number of tied distances is astronomical. This makes it impossible to directly obtain a reasonable hierarchical tree based on the distances among units. The problem caused by the ties is that in each step of the clustering process a large number of alternative agglomerations (or divisions) are possible. Several authors attempted to solve this problem by using measures of proximity among units different from their distances [13–15]. However, to justify the usage of any of these alternative parameters is difficult. A few years ago, we devised a valid strategy to solve the ties in proximity problem [12]. The first step consists in generating a large number of alternative, mathematically equivalent partitions of the network using the distances among the units (*primary distances*, according to our nomenclature) and conventional (e. g. average linkage) hierarchical clustering. The results are then averaged to obtain a weighted distance measure for each pair of units (*secondary distance*). This distance corresponds to the fraction of alternative partitions in which two units are assigned to different clusters. Finally, a dendrogram is obtained from the matrix of secondary distances. This strategy, which we called iterative cluster analysis, has already empirically demonstrated its usefulness. High-quality dendrograms have been obtained from complex networks derived from different types of biological data [16–18]. However, performing iterative hierarchical clustering has been so far hampered by the intrinsic slowness of obtaining a representative set of partitions. For example, our original program, UVCluster [12], runs in $O(n^3)$ time, n being the number of nodes. For this reason, the largest analysis published so far corresponds to a network with just 632 units [18].

In this work, we describe a suite of programs called Jerarca (Spanish for hierarch), which contains new, efficient algorithms to perform iterative hierarchical cluster analyses. One of them is basically a faster implementation of the UVCluster program. The other two, RCluster and SCluster, provide alternative ways to obtain the matrices of secondary distances from a graph. In addition, for the conversion of the matrix of distances into a dendrogram, two well-known phylogenetic algorithms, UPGMA and Neighbor-Joining [19–21] have been included in Jerarca. Finally, Jerarca also includes two different mathematical criteria to determine the best partition of the dendrogram into clusters. The first one is a parameter called modularity (Q) [10], which has been extensively used to measure community structure in networks. As an alternative, we include a modification of a hypergeometric distribution-based index suggested in one of our previous works [12]. Several output files, useful to edit and visualize the results, are generated by the program. All these options make Jerarca much more efficient and versatile than our original UVCluster program.

Methods

The Jerarca suite has been written in *C++*. Both the source code and compiled versions for Windows and Linux platforms are freely available at `http://jerarca.sourceforge.net`. Figure 2.1 details the control flow structure of the code. To perform a round of analyses, the user must execute the program from a command window, writing four parameters in the following order: 1) the name of a text file that describes the list of edges of the graph. The names of two linked nodes, separated by a tab or space, must be written in each line of the file; 2) the algorithm(s) chosen to iteratively calculate the matrix(ces) of secondary distances; 3) the algorithm(s) that will be used to obtain the dendrogram; and, 4) the number of iterations to be performed. Therefore, a typical Jerarca input has the following structure (parameters are indicated in brackets):

jerarca [file_name] [iterative_algorithm] [tree_algorithm] [n_iterations]

For the iterative algorithm, four options are valid: *uv* (UVCluster), *r* (RCluster), *s* (SCluster) and *all*. This last option will produce three parallel solutions, one

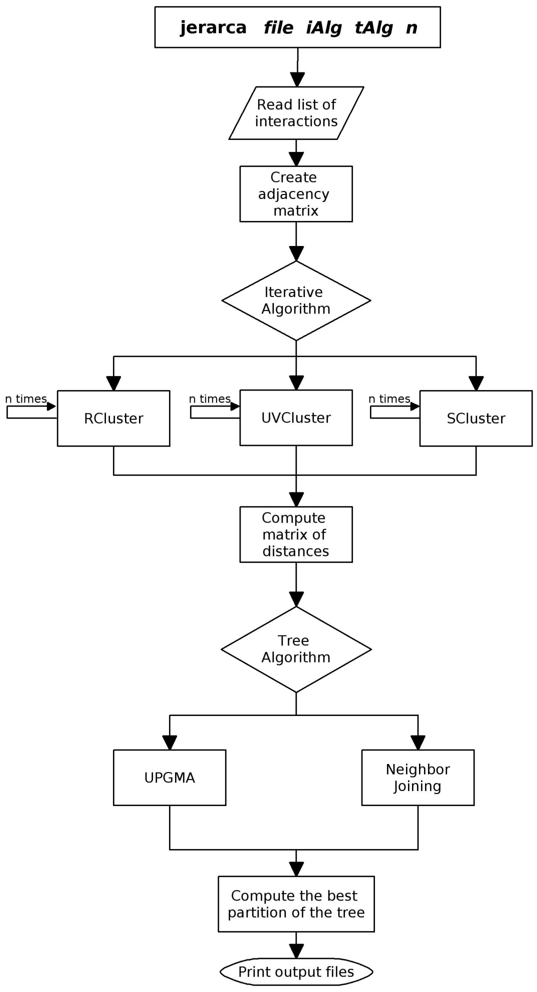
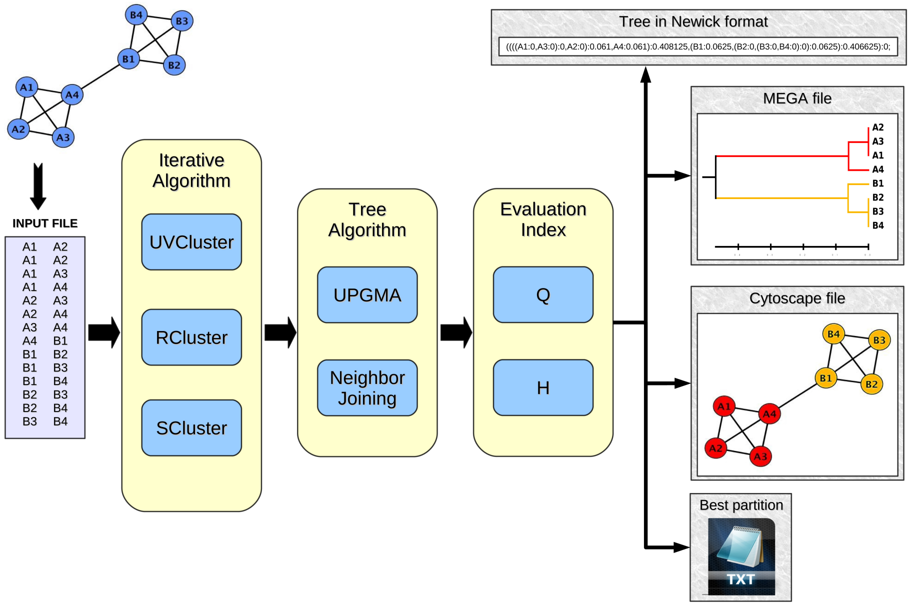


Figure 2.1: Control flowchart of Jerarca. The four input parameters are *file* (list of interactions that represent the edges of the network), *iAlg* (iterative algorithm to use), *tAlg* (tree algorithm to use) and *n* (number of iterations to perform).

for each available algorithm. For the tree algorithm, three options are valid: *u* (UPGMA), *nj* (Neighbor-joining) and *all*. This last option again will produce two solutions, one for each algorithm.

A typical Jerarca analysis is shown in Figure 2.2. In summary, the program reads the input file and creates the adjacency matrix A of the graph: $A_{ij} = 1$ if vertices i and j are connected and $A_{ij} = 0$ otherwise. Then, it applies the iterative algorithm(s) selected as many times as the number of iterations specified. To calculate the matrix of secondary distances, the algorithm saves, for each pair of nodes, the number of iterations in which they have been clustered separately, and the secondary distances between each two units are calculated by dividing those values by the number of iterations. After creating the matrix of secondary distances, the program uses the phylogenetic algorithm(s) chosen to build a dendrogram. The program finally evaluates, using the two indices implemented, each level of the dendrogram and saves the optimal partition of the tree for each index (see below). Several convenient output files (described also in detail below) are generated.



```
A1   A2
A1   A2
A1   A3
A1   A4
A2   A3
A2   A4
A3   A4
A4   B1
B1   B2
B1   B3
B1   B4
B2   B3
B2   B4
B3   B4
```

```
((((A1:0,A3:0):0,A2:0):0.061,A4:0.061):0.408125,(B1:0.0625,(B2:0,(B3:0,B4:0):0):0.0625):0.406625):0;
```

Figure 2.2: A typical analysis with Jerarca. The user specifies the input file where the graph is represented. It is analyzed by the program through diverse algorithms returning four different outputs: the tree in Newick format, a MEGA-compatible file, a file with attributes for Cytoscape and a text file containing the optimal partition of the tree.

Details of the iterative algorithms

We recently developed several novel ideas that are the basis of Jerarca. We first thought a way to notably improve the speed of the UVCluster program. UVCluster contained a parameter called *Affinity Coefficient (AC)*, which sets how permissive the clustering process is, in such a way that the lower the AC value, the larger the average distances among clustered units can be (see [12] for a detailed explanation). The maximum value of $AC = 100$ implies that only units that are directly connected in the graph are clustered together. Very significantly, this value was the only used in all our subsequent works [16–18]. Not a single useful application for other values has ever been found. This has an important consequence, given that, if we fix $AC = 100$, UVCluster-based iterative hierarchical clustering can be performed using the adjacency matrix of the network instead of the matrix of primary distances. This avoids computing the primary distances among all units using Floyd's algorithm, whose time complexity is $O(n^3)$. Once noticed that important point, we decided to generate a new version of UVCluster implementing this new approach. It turns out that this improved version is qualitatively faster than our former program, running in $O(n^2)$ time.

Two new algorithms, called RCluster and SCluster, described here for the first time, provide alternative ways to establish the matrix of secondary distances, following strategies related to the one implemented in the new version of UVCluster. These programs use alternative methods to select the units to be merged. Figure 2.3 shows a compact, technical description of their differences. However, we think that the reader may benefit from the following verbal summary of how the three programs work. The differences in the clustering process are as follows:

1. To select which units to merge, UVCluster generates in each iteration a list in which the units are randomly ordered and then proceeds to generate a cluster taking the first unit in that list and searching for all the units that can be merged to that one, according to the provided AC parameter. If $AC = 100$ (fixed value in the new version of the program) this means that UVCluster establishes cliques, i. e. groups in which each unit is connected with all the rest of units in the group. Once the largest clique that can be formed from the first selected unit is found, the units of that clique are set apart (i. e. they are considered to form a cluster) and the next unit still available in the list is used to start again the same process. This is a greedy algorithm, which tends to favor finding compact clusters.

2. Our second algorithm, RCluster (R meaning random), also establishes cliques but, instead of using a starting unit and greedily making a particular cluster to grow from it, RCluster in each step randomly merges two clusters, provided that all their units are connected (i. e. they form, after being merged, a clique). The program follows a hybrid strategy to select the clusters. To start with, the program simply randomly picks up two clusters, establishes whether they can be merged or not and, if indeed it is possible to merge

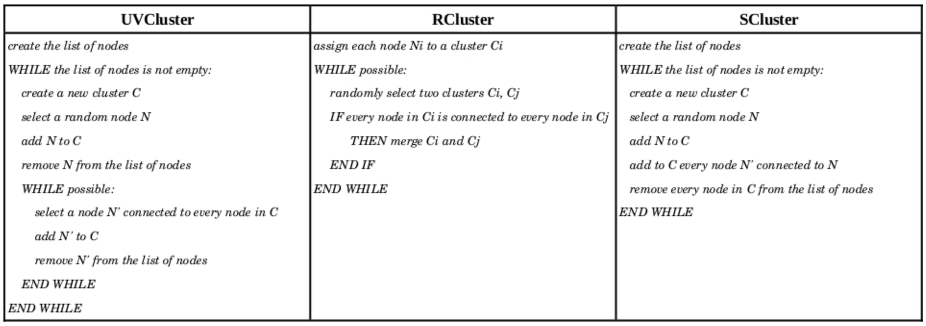
them, puts all the units together into a single, new cluster. While there are many clusters that can be merged, this simple strategy is very efficient and it has the big advantage of not requiring to recalculate the adjacency matrix in each merging step, something that is very time consuming for large graphs. However, as the merging process progresses, the likelihood of finding mergeable clusters just by randomly picking up two of them gets smaller. It is then convenient to shift to a second strategy, which is indeed based on generating in each step of the merging process an adjacency matrix, in which a value $A_{ij} = 1$ means that the units of the two clusters (i, j) form a clique. This second strategy is implemented in two steps: 1) The program generates an adjacency matrix and then randomly searches for a $A_{ij} = 1$ value in that matrix to merge two clusters; 2) It recalculates the adjacency matrix. Logically, for the newly formed cluster, it assigns a value of "1" with another cluster only when all the units in both clusters are connected. These two processes are repeated until no clusters can be merged. The transition from the first to the second strategy occurs when n random picks, n being the number of nodes of the network, have failed to find two mergeable clusters. Empirical analyses have shown this to be a convenient cutoff. Notice that, in RCluster, and differently from what occurs in UVCluster, multiple clusters grow at the same time. However, the process of choosing a random pair of clusters to merge in each iteration makes the program slower than the current version of UVCluster. We found that it runs in $O(n^2 \log n)$ time.

3. Finally, the third alternative is our novel SCluster algorithm (S stands for simple), which is both our greediest and our fastest algorithm, running in $O(n \log n)$ time. SCluster just picks up a unit by random and then collapses in a cluster that unit with all the units directly connected to it. These units are removed from the graph and then another unit is randomly chosen and the process is repeated until no further units remain. Notice the difference with UVCluster and RCluster: the units collapsed in a cluster do not have to be all connected among them (forming cliques) but just linked to the initial unit.

Dendrogram algorithms and evaluation of the partitions

Using any/all the algorithms described above, a matrix of secondary distances is obtained from which dendrograms can be generated. Jerarca implements two well-known phylogenetic algorithms for this task, UPGMA and Neighbor-Joining. The user may run one or both algorithms.

From the dendrogram, partitions of the units into clusters can be obtained. Jerarca establishes partitions by scanning the dendrogram from the root to the external leaves. Starting from the root, each dichotomy in the tree (that increases the number of clusters) generates an alternative partition that can be evaluated. Given that the neighbor-joining method generates unrooted trees, the middle


UVCluster

```
create the list of nodes
WHILE the list of nodes is not empty:
    create a new cluster C
    select a random node N
    add N to C
    remove N from the list of nodes
    WHILE possible:
        select a node N' connected to every node in C
        add N' to C
        remove N' from the list of nodes
    END WHILE
END WHILE
```

RCluster

```
assign each node Ni to a cluster Ci
WHILE possible:
    randomly select two clusters Ci, Cj
    IF every node in Ci is connected to every node in Cj
        THEN merge Ci and Cj
    END IF
END WHILE
```

SCluster

```
create the list of nodes
WHILE the list of nodes is not empty:
    create a new cluster C
    select a random node N
    add N to C
    add to C every node N' connected to N
    remove every node in C from the list of nodes
END WHILE
```

Figure 2.3: Main loop of the three iterative clustering algorithms implemented in Jerarca. An iteration defines a partition of the network by assigning the nodes to clusters. These loops are repeated as many times as iterations are specified by the user.

point of the tree is used as root [22]. Jerarca implements two mathematically independent criteria in order to evaluate the community structure of a given partition. The first index is the well-known and broadly used modularity (Q) [10], which measures the distribution of within and between communities links in a certain partition compared to the expected number of connections that should exist given a specific degree distribution [23]. The second index (called H) is based on the cumulative hypergeometric distribution of links, and derives from an index proposed in the paper that described UVCluster [12]. The definition of H is as follows:

$$H = -\log \sum_{j=p}^{min(M,n)} \frac{\binom{M}{j}\binom{F-M}{n-j}}{\binom{F}{n}} \tag{2.1}$$

where F is the maximum possible number of direct interactions in the whole network (for a network of k elements, $F = k(k-1)/2$), n is the number of direct interactions actually observed among the k elements of the network, M is the maximum possible number of intracluster direct interactions in a given partition and p is the total number of direct intracluster interactions actually detected in that partition. The parameter H measures the probability of obtaining by chance a given partition assuming a random distribution of intracluster and intercluster connections. The larger the value of H, the better ("more unexpected") the partition of the tree.

Output files

Jerarca produces four types of output files (Figure 2.2). Their names, automatically generated, include a reference to the algorithms and the evaluation criterion used (e. g. a typical name would be"Filename_partitionH_SCluster_Upgma.txt"). Moreover, the extension of a file specifies the content of the output:

1. Files with ".meg" extension contain the matrix of distances among units and the clusters obtained in the optimal partition of the dendrogram, according to either Q or H. This file can be directly imported into the software MEGA 4 [24] for further analyses.

2. Files with ".att" extension contain the assignment of nodes to clusters in the best partition. These files are designed to be imported into Cytoscape (version 2.x) [25] as attributes of the nodes (from the main Cytoscape menu: File - Import - Node attributes).

3. Files with a ".txt" extension save the best partition of the dendrogram obtained in text format. They include a description of the optimal partition: number of clusters, value of the index used and the assignment of nodes to each cluster.

4. Finally, the files with ".nwk" extension describe the dendrogram structure in standard Newick format, which can be read by virtually all programs that analyze trees, such as MEGA.

Results

The speed of the programs has been tested in several benchmarks. Here we describe the results for three of them, consisting in an artificial and two real networks:

Benchmark A
We prepared a synthetic graph of known community structure, in which 512 units were divided into 16 clusters of equal size. Within each cluster all units were initially fully connected (for a total of $(k^2 - k)/2$ edges, being k the number of units in a cluster). Then, we progressively "degraded"that structure by removing a certain percentage of edges and then randomly shuffling a number of edges among the units. The networks generated are a variation of the *connected-caveman* graphs defined by Watts [26].

Benchmark B
The proteins (nodes) that constitute 408 different protein complexes described in the yeast *Saccharomyces cerevisiae* were obtained from the CYC2008 database (http://wodaklab.org/cyc2008; [27]). We then downloaded from the BioGRID database [28] the protein-protein interactions (edges) characterized so far for all these proteins. The final graph contained 1604 nodes and 14171 edges.

Benchmark C

The complete set of protein-protein interactions (interactome) of *S. cerevisiae* was obtained from BioGRID. These data generated a network formed by 5735 nodes (proteins) and 51134 edges (protein-protein interactions).

Benchmark A was specifically created for testing the quality of the optimal partitions computed by the algorithms implemented in Jerarca. We generated networks with progressive percentages of degradation. In this context, a percentage of degradation of, say, 10 %, means that first, 10 % of links were eliminated and, from the rest, 10 % shuffled among units. The shuffling process involves the random removal of an edge of the graph and the later addition of a new edge between two nodes, chosen also randomly. We previously suggested using a number of iterations equal to 10 times the number of units [12]. Thus, for each of those networks, we ran 5000 iterations of Jerarca with the parameter *all* for both the iterative and the tree algorithms. This means that 12 analyses (= 3 iterative algorithms x2 tree algorithms x2 partition criteria) were performed for each network. With 0-30 % degradation, all algorithms recovered the original community structure of the network without errors. However, starting at 40 % degradation, slight errors in recovering the original community structure of the graph began to emerge, so we focused on this case. For each of the six dendrograms constructed by using the three iterative and the two tree algorithms, the optimal partitions given by the two evaluation indexes implemented in Jerarca (Q and H) were exactly the same. In all cases but one, a single unit of the network, different for each combination of programs, was misclassified. Only the combination of SCluster and UPGMA recovered the exact community structure of the original network. Significantly, this particular combination also obtained the highest Q and H values. This example shows that all the programs efficiently recover the original structure, even when it is quite cryptic (40 % degradation means that just about a third of the original links remain). On the other hand, it also shows the advantage of using when possible all the programs together, given that some may perform better than others.

We performed speed tests in a PC-compatible computer with an Intel Core 2 Quad Q8200 at 2.33 GHz and 4 GB of RAM, running Linux. The analyses of benchmark A were very fast. The 12 analyses per network described in the previous paragraph (5000 iterations/analysis) required just between 30 and 75 seconds. The least degraded (= more compact) graphs, allow for the fastest analyses. To test the speed of the program in real networks of larger sizes, we used benchmarks B and C. For benchmark B (1604 nodes), 16000 iterations took about 3.25 hours when using the RCluster algorithm, while for UVCluster and SCluster the cost was 2 minutes and less than a minute respectively. This large difference is due to the fact that this network contains densely connected modules (each protein complex was much more tightly connected internally than with the rest of the network), a feature that favors the greedy strategies implemented in UVCluster and SCluster. For benchmark C (5735 nodes), 60000 iterations took

40 minutes with SCluster and about 3 hours with UVCluster. For RCluster, we estimated the analysis to require around 300 hours, so it was not performed in full.

In summary, the new algorithms implemented in Jerarca make possible to analyze large networks. As the times just detailed demonstrate, a single computer may easily cope with problems involving several thousands of units in a reasonable time, using both UVCluster and SCluster. Also, for networks with up to 1000 nodes, the user can test the three programs together, obtaining the results in minutes to a few hours.

Discussion

As the amount of biological information is rapidly increasing, one of the main goals in bioinformatics is the generation of fast programs able to deal with large datasets. For network analyses, the bottleneck of the iterative hierarchical clustering strategy is precisely that the clustering algorithm must be repeatedly used to generate a sufficiently large set of iterations as to be representative of the underlying structure of the graph. The second part of the analysis, the construction of the tree applying a phylogenetic algorithm is performed just once and therefore has little effect in the time complexity of the program. As already indicated in the Introduction, the applications of our UVCluster program were limited by the high amount of time needed for analyzing large networks. An optimization of the iterative clustering method implemented in that program was therefore mandatory. By setting certain restrictions (fixed AC), we have qualitatively reduced the time complexity of the UVCluster algorithm. Traditionally limited to analyses below 1000 units, the current algorithm can cope with networks of several thousand units in a few hours. This allows analyzing some very interesting datasets, such as the whole interactome of the eukaryotic species *Saccharomyces cerevisiae* (see benchmark C above).

A second significant advantage of Jerarca is that it also includes two novel algorithms, RCluster and SCluster, which provide alternative ways of computing the secondary distances between the nodes of the graph. RCluster randomly grows multiple clusters at the same time, avoiding the greedy agglomerative process implemented in UVcluster. However, the randomization process required makes the program slower than UVCluster. SCluster is just the opposite: it is the fastest and greediest of the three algorithms. In spite of its simplicity, its performance is also appropriate (See results for benchmark A above). Since Jerarca allows to execute several parallel analyses, we recommend to use the three iterative algorithms for networks with up to 1000 nodes. A complete analysis of such networks may require less than two hours (see Results). With larger networks, up to 10000 units, both UVCluster and SCluster can be used, the analyses with both programs requiring just a few hours. The inclusion of SCluster, which runs in $O(n \log n)$ time, allows for the analyses of even larger networks. This may be of interest in fields such as the analysis of coexpression or gene interaction networks, in which the number of nodes (in those cases, corresponding to genes) may be in the tens of thousands. All these considerations obviously refer to analyses using a single computer. However, it is important to take into account that the programs can be very easily parallelized, given that the iterations can be divided into multiple processors and the results added together at the end of the computation.

In addition to UPGMA, already included in the original version of UVCluster, Jerarca also allows the alternative of building the trees using the neighbor-joining algorithm, probably the most frequently used algorithm to generate trees from a distance matrix. We suggest to obtain both trees (which is almost instantaneous), in order to evaluate the congruence of the results. An additional advantage of Jerarca respect to UVCluster refers to the determination of the optimal partitions of the graph according to two statistical parameters (Q and H). We added these options considering that the users may be often not only interested in obtaining a hierarchical representation of the network, but also in how the network can be divided into clusters or communities (see Introduction). The strategy used to obtain the partitions is in fact quite simple, given that the tree is just scanned from root to leaves. Therefore, the number of partitions examined is quite reduced (equal to the number of nodes n). More complex methods can be easily envisaged. For example, partitions could be generated at different distances from the root in different sections of the tree. However, although this option may potentially improve the likelihood of obtaining a better partition of the network, it is computationally much more expensive. We plan to explore this possibility in future versions of the suite. A final advantage of Jerarca is the set of outputs that it generates, which is much more complete than the one provided by our original UVCluster program. The possibility to directly export the data to powerful packages such as MEGA and Cytoscape will allow the users both to perform additional analyses that may complement those generated by Jerarca and to obtain sophisticated graphical representations of the results. All these advantages clearly make Jerarca a better tool to perform iterative clustering analyses of network data than our original UVCluster program.

The program, along with the source code is freely available under the *GNU General Public License v3* at http://jerarca.sourceforge.net. The modular code structure of Jerarca permits easily including new features to the program. New algorithms, both iterative and for building the trees, as well as new indexes for extracting the optimal partition of the tree, can be easily added.

Acknowledgments

The authors would like to thank Vicente Arnau for his contributions in the development of the original UVCluster program, and Antonio Marco for his suggestions about how to implement the calculation of the H parameter.

References

[1] A.L. Barabási and Z.N. Oltvai. Network biology: understanding the cell's functional organization. *Nature Reviews Genetics*, 5:101–113, 2004.

[2] T. Aittokallio and B. Schwikowski. Graph-based methods for analysing networks in cell biology. *Briefings in bioinformatics*, 7:243–255, 2006.

[3] R. Sharan, I. Ulitsky, and R. Shamir. Network-based prediction of protein function. *Molecular systems biology*, 3:88, 2007.

[4] V. Spirin and L.A. Mirny. Protein complexes and functional modules in molecular networks. *Proceedings of the National Academy of Sciences*, 100:12123–12128, 2003.

[5] G.D. Bader and C.W.V. Hogue. An automated method for finding molecular complexes in large protein interaction networks. *BMC bioinformatics*, 4:2, 2003.

[6] C. Brun, C. Herrmann, and A. Guénoche. Clustering proteins from interaction networks for the prediction of cellular functions. *BMC bioinformatics*, 5:95, 2004.

[7] J.B. Pereira-Leal, A.J. Enright, and C.A. Ouzounis. Detection of functional modules from protein interaction networks. *Proteins: Structure, Function, and Bioinformatics*, 54:49–57, 2003.

[8] N. Pržulj, DA Wigle, and I. Jurisica. Functional topology in a network of protein interactions. *Bioinformatics*, 20:340–348, 2004.

[9] M. Altaf-Ul-Amin, Y. Shinbo, K. Mihara, K. Kurokawa, and S. Kanaya. Development and implementation of an algorithm for detection of protein complexes in large interaction networks. *BMC bioinformatics*, 7:207, 2006.

[10] M.E.J. Newman and M. Girvan. Finding and evaluating community structure in networks. *Physical review E*, 69:026113, 2004.

[11] AD King, N. Pržulj, and I. Jurisica. Protein complex prediction via cost-based clustering. *Bioinformatics*, 20:3013–3020, 2004.

[12] V. Arnau, S. Mars, and I. Marín. Iterative cluster analysis of protein interaction data. *Bioinformatics*, 21:364–378, 2005.

[13] A.W. Rives and T. Galitski. Modular organization of cellular networks. *Proceedings of the National Academy of Sciences*, 100:1128–1133, 2003.

[14] H. Lu, X. Zhu, H. Liu, G. Skogerbø, J. Zhang, Y. Zhang, L. Cai, Y. Zhao, S. Sun, J. Xu, et al. The interactome as a tree—an attempt to visualize the protein–protein interaction network in yeast. *Nucleic acids research*, 32:4804–4811, 2004.

[15] A.M. Yip and S. Horvath. Gene network interconnectedness and the generalized topological overlap measure. *BMC bioinformatics*, 8:22, 2007.

[16] J.I. Lucas, V. Arnau, and I. Marín. Comparative genomics and protein domain graph analyses link ubiquitination and rna metabolism. *Journal of molecular biology*, 357:9–17, 2006.

[17] A. Marco and I. Marín. A general strategy to determine the congruence between a hierarchical and a non-hierarchical classification. *BMC bioinformatics*, 8:442, 2007.

[18] A. Marco and I. Marín. Interactome and gene ontology provide congruent yet subtly different views of a eukaryotic cell. *BMC systems biology*, 3:69, 2009.

[19] R.R. Sokal. A statistical method for evaluating systematic relationships. *Univ Kans Sci Bull*, 38:1409–1438, 1958.

[20] N. Saitou and M. Nei. The neighbor-joining method: a new method for reconstructing phylogenetic trees. *Molecular biology and evolution*, 4:406–425, 1987.

[21] M. Nei and S. Kumar. *Molecular evolution and phylogenetics.* Oxford University Press, USA, 2000.

[22] J.S. Farris. Estimating phylogenetic trees from distance matrices. *American Naturalist*, 106:645–668, 1972.

[23] A. Clauset. Finding local community structure in networks. *Physical Review E*, 72:026132, 2005.

[24] K. Tamura, J. Dudley, M. Nei, and S. Kumar. Mega4: molecular evolutionary genetics analysis (mega) software version 4.0. *Molecular biology and evolution*, 24:1596–1599, 2007.

[25] M.S. Cline, M. Smoot, E. Cerami, A. Kuchinsky, N. Landys, C. Workman, R. Christmas, I. Avila-Campilo, M. Creech, B. Gross, et al. Integration of biological networks and gene expression data using cytoscape. *Nature protocols*, 2:2366–2382, 2007.

[26] D.J. Watts. *Small worlds: the dynamics of networks between order and randomness.* Princeton university press, 2003.

[27] S. Pu, J. Wong, B. Turner, E. Cho, and S.J. Wodak. Up-to-date catalogues of yeast protein complexes. *Nucleic acids research*, 37:825–831, 2009.

[28] B.J. Breitkreutz, C. Stark, T. Reguly, L. Boucher, A. Breitkreutz, M. Livstone, R. Oughtred, D.H. Lackner, J. Bähler, V. Wood, et al. The biogrid interaction database: 2008 update. *Nucleic acids research*, 36(suppl 1):D637–D640, 2008.

Deciphering network community structure by Surprise

Rodrigo Aldecoa and Ignacio Marín
Instituto de Biomedicina de Valencia. Consejo Superior de Investigaciones Científicas (IBV-CSIC) Calle Jaime Roig 11. Valencia, Spain

PLoS ONE 6, e24195 (2011)

The analysis of complex networks permeates all sciences, from biology to sociology. A fundamental, unsolved problem is how to characterize the community structure of a network. Here, using both standard and novel benchmarks, we show that maximization of a simple global parameter, which we call Surprise (S), leads to a very efficient characterization of the community structure of complex synthetic networks. Particularly, S qualitatively outperforms the most commonly used criterion to define communities, Newman and Girvan's modularity (Q). Applying S maximization to real networks often provides natural, well-supported partitions, but also sometimes counterintuitive solutions that expose the limitations of our previous knowledge. These results indicate that it is possible to define an effective global criterion for community structure and open new routes for the understanding of complex networks.

Introduction

A network of interacting units is often the best abstract representation of real-life situations or experimental data. This has led to a growing interest in developing methods for network analysis in scientific fields as diverse as mathematics, physics, sociology and, most especially, biology, both to study organismic (e. g. populational, ecological) and cellular (metabolic, genomic) networks [1–5]. A significant step to understand the properties of a network consists in determining its communities, compact clusters of densely linked, related units. However, the best way to establish the community structure of a network is still disputed. Many strategies have been used (reviewed in [6]), the most popular being the maximization of Newman and Girvan's modularity (Q) [7]. However, Q has the drawback of being affected by a resolution limit: its maximization fails to detect communities smaller than a threshold size that depends on the total size of the network and the pattern of connections [8]. Since this finding, no other global parameters have been proposed to substitute Q. Alternative strategies (searching for local structural determinants, multilevel optimization of Q) have been suggested, but none of them has achieved general acceptance [6].

Some years ago, we suggested determining the community structure of a network by evaluating the distributions of intra- and inter-community links with a cumulative hypergeometric distribution [9]. Accordingly, to find the optimal community structure of a network of symmetrically connected units (undirected graph) is equivalent to maximize the following parameter:

$$S = -\log \sum_{j=p}^{min(M,n)} \frac{\binom{M}{j}\binom{F-M}{n-j}}{\binom{F}{n}} \tag{2.2}$$

Where F is the maximum possible number of links in a network (i. e. $[k^2 - k]/2$, being k the number of units), n is the observed number of links, M is the maximum possible number of intracommunity links for a given partition, and p is the total number of intracommunity links actually observed in that partition. The parameter S, which stands for Surprise, indeed measures the "surprise" (improbability) of finding by chance a partition with the observed enrichment of intracommunity links in a random graph.

In this work, we show that S has features that make it the parameter of choice for global estimation of community structure. By using standard and novel benchmarks and a set of high-quality algorithms for community detection, we show that maximizing S often provides optimal characterizations of the existing communities. When this method is applied to real networks, we obtained some expected, logical solutions - some of them much better than those provided by Q maximization - but also unexpected partitions that demonstrate the limitations that the usage of inefficient tools has hitherto cast over the field.

Results

Testing the performance of a global parameter to determine community structure requires both a set of efficient algorithms for community detection and a set of standard benchmarks, consisting in synthetic networks of known structure. In this study, six selected algorithms (see Methods) were tested in two types of benchmarks, which will be called LFR and RC throughout the text. LFR (Lancichinetti-Fortunato-Radicchi) benchmarks are characterized by providing networks in which both the degrees of the nodes and the sizes of the communities follow power laws [10]. RC (Relaxed Caveman) benchmarks start with networks in which all the nodes in a community are connected. Then, this structure is relaxed by generating intercommunity links [11]. We further divided LFR and RC benchmarks into "open" and "closed". Open benchmarks have been commonly used in the past (e.g. [10, 12, 13]). In them, sets of similar networks with different proportions of intercommunity links are tested. With many intercommunity links, the networks approach randomness. In closed benchmarks, a starting community structure is progressively transformed into a second, final structure which is exactly known.

For each benchmark, we estimated S and Q with the six algorithms. The maximum values of S and Q obtained (S_{max} and Q_{max}) provided the partitions used to compare with the known community structures. As in previous works [10, 14, 15], Normalized Mutual Information (NMI) was used to measure the congruence between the known and the estimated community structures. However, we also used the Variation of Information (VI) [16] in a particular case.

Open benchmarks

Figures 2.1a and 2.1b summarize the results obtained for four standard open LFR benchmarks that differ in number of units and community sizes [10] (see Methods). Figure 2.1a indicates that selecting the solution with a maximum S value leads to a perfect characterization of the network structure ($\text{NMI}_S = 1$) even when that structure is blurred by a large number of inter-community links, generated by increasing the mixing parameter μ up to 0.5-0.7 (see Methods for μ definition). If μ is further increased, the original partition is not chosen by any algorithm ($\text{NMI}_S < 1$). This suggests that the original community structure is not present anymore, which is in good agreement with the fact that $S_{max} \gg S_{orig}$, where S_{orig} is the S value obtained assuming that the original community structure is still present (Table S1). S maximization qualitatively improves over Q maximization (Figure 1b and Table S1): $\text{NMI}_S > \text{NMI}_Q$ in $2827/3600 = 78{,}5\,\%$ of the cases, $\text{NMI}_Q > \text{NMI}_S$ in just $4.1\,\%$ of them and the rest are ties. Interestingly, $\text{NMI}_Q \ll \text{NMI}_S$ in quasi-random and random networks (Figure 1b), suggesting that maximizing Q overimposes spurious community structures in those cases. It is significant that S maximization provided better average NMI scores than those obtained by any single algorithm in these same benchmarks [15]. Different

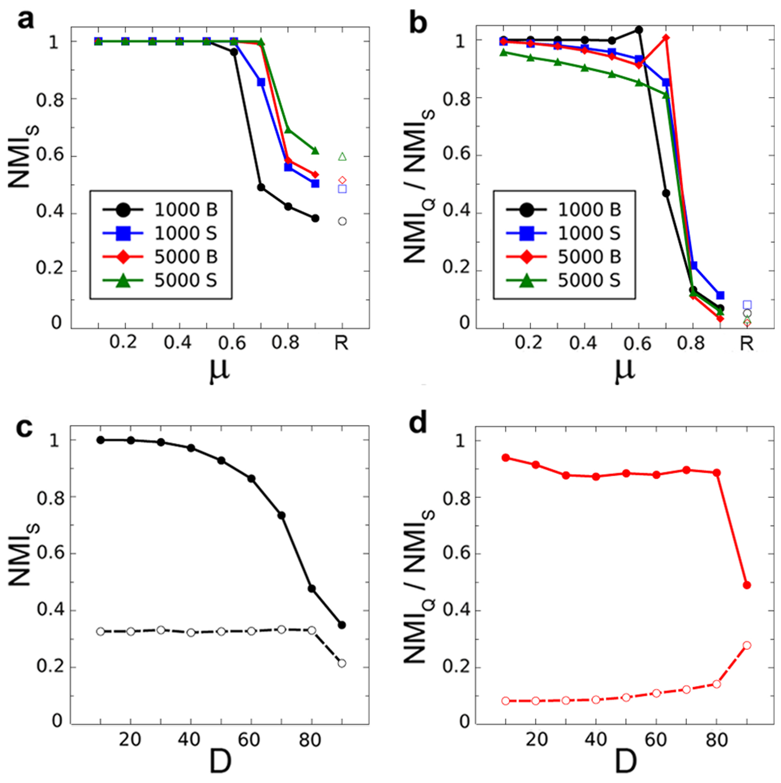


Figure 2.1: Results for open LFR and RC benchmarks. a) Results for the four standard LFR networks. B and S indicate big and small communities respectively and 1000 or 5000 the number of nodes. μ: mixing parameter. NMI measures the congruence between the known and the deduced community structures. Each point is based on 100 different networks; standard errors of the mean are too small to be visualized. Values for 100 random (R) networks with the same number of units and degree distributions are also shown. b) Comparison of S and Q maximizations in LFR benchmarks. The $\text{NMI}_Q/\text{NMI}_S$ ratios, which are almost always below 1, are shown. c) Results for the RC benchmark. The parameter Degradation (D) indicates the percentage of both deleted and shuffled links. Each black dot is based on 100 networks, again standard errors are so small that cannot be visualized at this scale. For each value of D, results for 100 random networks with the same number of links are also shown (open circles). d) Relative quality of the partitions generated by maximizing S and Q in RC benchmarks. As in panel b, $\text{NMI}_Q/\text{NMI}_S$ ratios are shown. White dots: results for random networks with different D values.

algorithms provided the top S scores, depending on the benchmark and μ value examined (Figure 2.2a and Figure S1).

The discovery of the resolution limit of Q showed that heterogeneous community sizes may greatly affect the ability of global parameters to detect structure [8]. However, by construction, community sizes in the standard LFR benchmarks are very similar. Pielou's evenness indexes (PI) [21] ranged from 0.96 to 0.98 in the four benchmarks used above, close to the maximum value of the index (PI = 1 for communities of identical size). Considering that it was critical to test S in more extreme situations, we built the RC benchmarks, which have PIs as low as 0.70 (as shown in Figure S2). Figures 2.1c and 2.1d summarize the results for open RC benchmarks, with progressive Degradation (D; see Methods) of the original structure. That structure is efficiently detected by S maximization, with a slow decrease in performance when D increases (Figure 1c; see also Table S2, Figure S2). Again, S maximization clearly improves over Q maximization in these benchmarks (Figure 2.1d; $\text{NMI}_S > \text{NMI}_Q$ in $848/900 = 94{,}2\,\%$ of the cases, while $\text{NMI}_Q > \text{NMI}_S$ in just 3.3 % of the cases). As occurred for the LFR benchmarks, none of the algorithms obtained the best results in all networks (Figure 2.2b).

Closed benchmarks

The results just shown indicate that using S_{max} to detect community structure has obvious advantages over maximizing Q. However, they do not allow to evaluate how optimal is that criterion, given that the potential maximum NMIs are unknown. To solve this limitation, we generated closed LFR and RC benchmarks, in which we had an *a priori* expectation of the maximum NMI values. Results are shown in Figures 2.3 (LFR) and 2.4 (RC). In all cases in which S_{max} was used, an almost perfectly symmetrical dynamics was observed. In the process of converting the original structure into the final one (by increasing the Conversion parameter; see Methods), NMI losses for the first structure are compensated by increases for the second. The average of both NMIs is thus approximately constant, and it has a value identical or very close to $(1+\text{NMI}_{IF})/2$, where NMI_{IF} is obtained comparing the initial and final structures (Figures 2.3a-d; Figures 2.4a-c; Figures S3, S4). This is exactly the result expected for an optimal parameter (see theoretical details in Methods). On the contrary, maximizing Q shows a poor performance except when community sizes are very similar/identical (Figures 2.3e, 2.4d; Figures S3, S4). The same results were obtained using a second measure of congruence, Variation of Information (VI) (Figures S5, S6). Finally, in the LFR benchmarks, S_{max} was always identical or higher than S_{orig} (Figure 2.3f). However, this does not happen for the RC benchmarks (Figure 2.4e). Therefore, these algorithms sometimes fail to obtain the highest possible S values. This fact may explain the slight departures from NMI symmetry observed in some RC benchmarks (blue diamonds in Figures 2.4b, 2.4c).

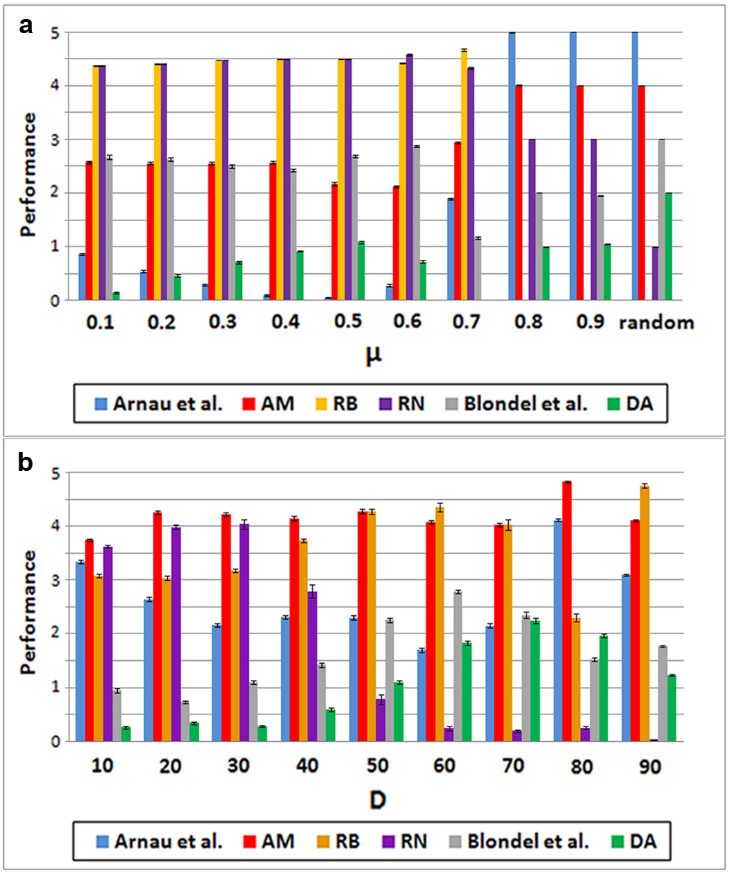


Figure 2.2: Average performance of the algorithms in the open LFR and RC benchmarks. The algorithms used were described by Arnau et al. [9], Aldecoa and Marín (AM) [13], Rosvall and Bergstrom (RB) [17], Ronhovde and Nussinov (RN) [18], Blondel et al. [19] and Duch and Arenas (DA) [20]. a) Typical example of the results obtained in LFR benchmarks, here with 5000 units and big communities (see Figure S1 for all of them). After ordering the algorithms from best to worst performance, their ranks were added for the 100 different networks. Performance was defined as $P = 6 - average\_rank$. Therefore, the maximum value $P = 5$ means that an algorithm was the best in all networks tested, while $P = 0$ means that it was always the worst. As it can be observed, none of the algorithms achieved optimal results in all cases. b) Results obtained in the RC benchmark with different Degradation (D) values. Performance evaluated as in panel a).

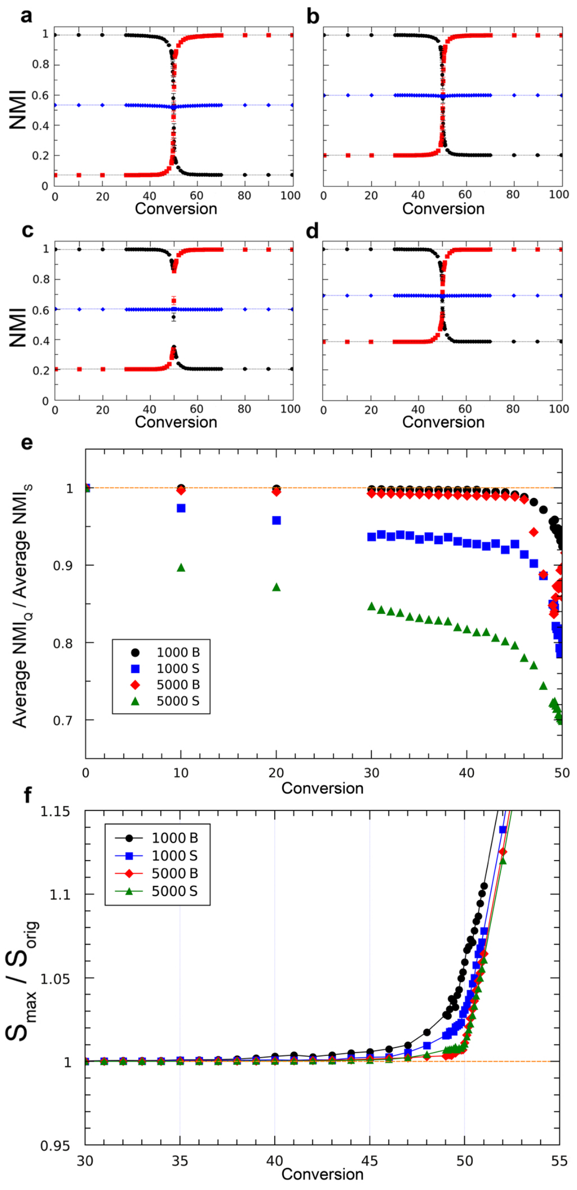
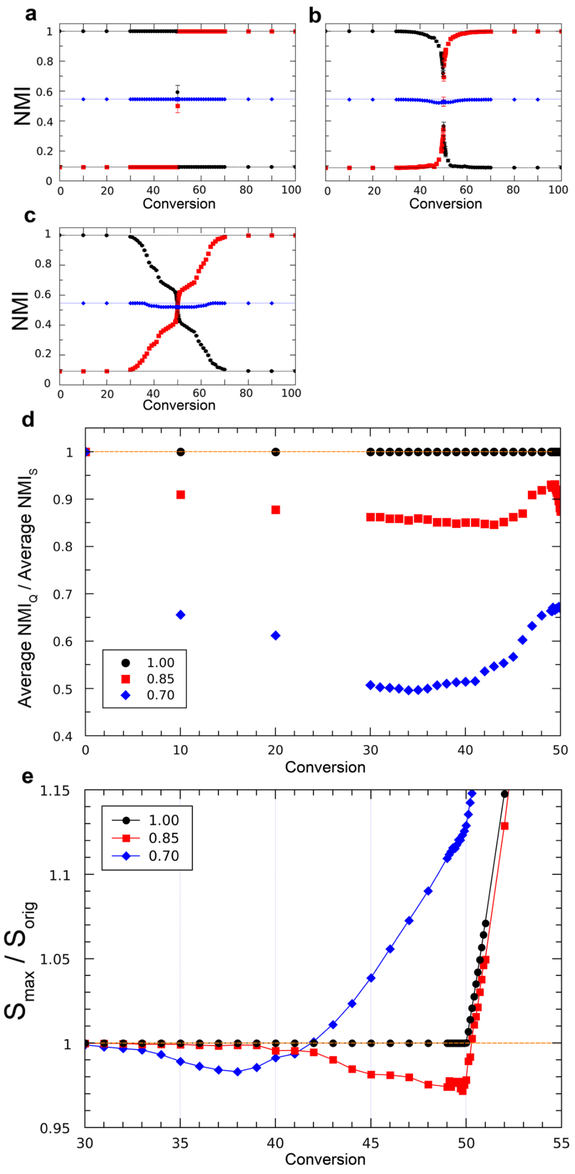


Figure 2.3: Results for closed LFR benchmarks. a) LFR benchmark with 1000 units and big communities. For each Conversion (C) value, NMIs comparing the S_{max} partition with the initial (black dots) or final (red squares) community structures were obtained. The symmetrical results led to NMI averages (blue diamonds) that, with great precision, fell in a straight line of value $(1+\text{NMI}_{IF})/2$. Dots are based on 100 independent analyses. b–d) LFR benchmarks with, respectively, 1000 units, small communities (b), 5000 units, big communities (c) and 5000 units, small communities (d). Results are very similar to those in panel a). e) Average NMI values for partitions obtained maximizing Q are worse than those obtained maximizing S, especially as we move towards $C = 50$, in which the real community structure is more difficult to establish. This effect is exacerbated by large number of units and small community sizes, due to the resolution limit of Q. Results for $C > 50$ are symmetrical to the ones shown here. See also Figure S3. f) S_{max}/S_{orig} ratio ≥ 1, i. e. either the original structure or a different one with higher S is found. These results are compatible with the algorithms used being able to detect the true structure present with great accuracy.

Figure 2.4: Results for closed RC benchmarks. Three networks with different heterogeneity in community sizes (Pielou's indexes equal to 0.70, 0.85 and 1.00 respectively) were used as examples. a) $PI = 1$; b) $PI = 0{,}85$; c) $PI = 0{,}70$. Results similar to those in Figure 2, except that the figures are not so perfectly symmetrical in the most heterogeneous networks (panels b and c; blue diamonds slightly deviate from the straight line). d) Average NMI values are much worse when Q is used, provided that community sizes are heterogeneous. See also Figure S4. e) $S_{max}/S_{orig} < 1$ with heterogeneous community sizes. The algorithms used did not detect in those cases the maximum possible S, which still may correspond to the initial structure. This may contribute to the departures from symmetry shown in panel a). The fact that $S_{max}/S_{orig} \gg 1$ with $C < 0{,}50$ and $PI = 0{,}70$ (blue diamonds) implies that the algorithms are detecting structures different from the initial one.



Real networks

Figure 5 summarizes the S_{max} results for three real networks. The first example is based on the CYC2008 database, which compiles 1604 proteins that belong to 324 protein complexes [22]. The general agreement between communities detected using S_{max} and *a priori* defined protein complexes is almost perfect, $\text{NMI}_S = 0.91$. On Figure 2.5a, the 11 communities of size >20, out of the 313 detected, are detailed to show how fine-grained is the classification obtained. On the contrary, optimizing Q provides a very coarse classification into just 24 communities with $\text{NMI}_Q = 0.57$. The largest five communities alone almost cover the whole network (Figure 2.5b). These results indicate how excellent is S performance when there are many small, abundant communities, a typical situation in which Q, affected by its resolution limit, radically fails. Figure 2.5c shows, as a positive control, the results for a classical benchmark of well-known structure, the *College football network* [12]. The agreement with the expected communities is again very high ($\text{NMI}_S = 0.93$). Finally, Figure 2.5d shows the results for another well-known example, the *Zachary's Karate club* network [12, 23]. This social network supposedly contains two communities. However, S analyses surprisingly unearthed 19 communities, 12 of them singletons (Figure 2.5d).

Discussion

In this study, we have shown the potential of maximizing the global parameter Surprise (S) to determine the community structure present in complex networks. The results indicate that it has a qualitative better performance than the hitherto most commonly used global measure, Newman and Girvan's modularity (Q). The advantage of S over Q is maybe not that surprising, considering the different theoretical foundations of both measures. Newman and Girvan's Q is based on a simple definition of community, as a region of the network with an unexpectedly high density of links. However, the number of units within each community does not influence the value of Q [7]. On the contrary, S evaluates both the number of links and of units in each community (see 2.2). Therefore, S implicitly assumes a more complex definition of community: a precise number of units for which it is found a density of links which is statistically unexpected given the features of the network. In this context of comparison of both measures, it is also very significant that, while some of the algorithms used in this work were the best among those specifically designed to maximize Q, none was devised to maximize S. Therefore, our results actually underestimate the power of S maximization for community detection. A direct example of that underestimation is shown in Figure 2.4e: the maximum values of S were, in some cases, not found. The few exceptions found in which $\text{NMI}_Q > \text{NMI}_S$ (3-4 % of all the cases examined in the open benchmarks) could be also explained by an incomplete success in determining S_{max} with these algorithms.

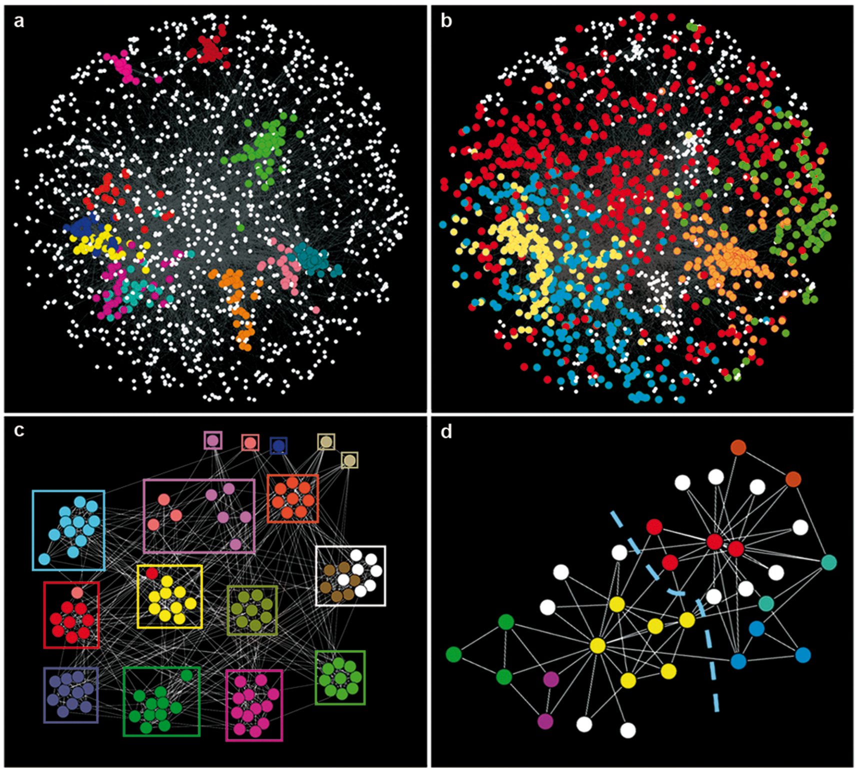


Figure 2.5: Community structure of the CYC2008 network (a, b), College football network (c) and Zachary's karate club network (d), according to S maximization (panels a, c, d) or Q maximization (panel b). In panel c, the known community structure is shown (squares). The broken lines in panel d divide the network into the two communities assumed to exist. That division of the network is not supported at all by S_max analyses. While $S_{(2communities)}$ = 13.61, the optimal division found has $S_{(19communities)}$ = 25.69. Twelve of these optimal communities are singletons (white dots).

The commonly used open benchmarks are useful for general evaluations of the performance of different algorithms, but they do not allow to establish how optimal are the results obtained. For that, we have devised novel closed benchmarks in which an initial known community structure is progressively transformed into a second, also known, community structure. Provided that both community structures are identical, it can be demonstrated that, at any point of the transformation from one to the other, the average of the NMIs of the solution found respect to the initial and final structures should approximate a constant value ([1+NMI$_{IF}$]/2), if that solution is optimal (see Methods). This feature allows establishing the intrinsic quality of the partitions obtained, with S maximization often providing optimal results. We conclude that S maximization establishes the community structure of complex networks with a high accuracy. Two promising lines of research are clear. First, generating novel, specific algorithms for S maximization, which may improve over the existing ones. Second, building a standard set of closed benchmarks to test any new algorithms for community detection. Our LFR and RC closed benchmarks may be a good starting point for that standard set.

When S maximization was applied to real networks, the results obtained are of two types. On one hand, for the CYC2008 and College football networks, the expectation was to find a clear community structure which should faithfully correspond to either the complexes to which the proteins examined are part (CYC2008 network) or to the conferences to which the teams belong (*College football network*), given that intracomplex or intraconference links are abundant (e. g. Figure 2.5c). These are exactly the results found using S_{max}. On the other hand, the structure of the *Zachary's karate* network is far from obvious (Figure 2.5d). Therefore, finding that, according to S_{max}, the network contains some small groups plus many singletons is, at least *a posteriori*, not so unexpected. A natural question is then why the scientific community has been so keen of exploring this particular network, often to establish whether an algorithm was able or not to detect the putative two communities (e. g. refs. [7, 12, 23, 24] among many others). This may reflect a psychological bias, to which the use of underperforming methods for community detection may have certainly contributed. It shows to which extent human prejudices may taint evaluations in this type of ill-defined problems.

Methods

Algorithms used to maximize S and Q

Six of the best available algorithms, selected either by their exceptional performance in artificial benchmarks or their success in previous analyses of real and simulated networks [9, 13–15, 25, 26], were used. They were the following: 1) UVCluster algorithm [9, 13]: It performs iterative hierarchical clustering, generating dendrograms. The best values of S and Q were obtained scanning these dendrograms from root to leaves. 2) SCluster algorithm [13]: also performs iterative hierarchical clustering, but using an alternative strategy which is faster and sometimes more accurate than the one implemented in UVCluster. 3) Dynamic algorithm by Rosvall and Bergstrom [17]: an algorithm based on expressing the characterization of communities as an information compression problem. 4) Potts model multiresolution algorithm [18]: works by minimizing the Hamiltonian of a Potts spin model at different resolution scales, i. e. searching for communities of different sizes. 5) Fast modularity optimization [19]: devised to maximize Q. It provides multiple solutions from which values for S and Q can be obtained, and the maximum ones were used in our analyses. 6) Extremal optimization algorithm [20]: A divisive algorithm also developed to maximize Q. Analyses were always performed with the default program settings.

Features of the benchmarks

First, the recently developed LFR benchmarks, specifically devised for testing alternative community detection strategies [10], were used. In particular, we chose four standard LFR benchmarks already explored by other authors [15]. The networks analyzed had either 1000 or 5000 units and were built according to two alternative ranges of community sizes (Big (B): 20-100 units/community; Small (S): 10-50 units/community). For each of the four conditions (1000 B, 1000 S, 5000 B, 5000 S), 100 different networks were generated for each value of a mixing parameter μ, which varied from 0.1 to 0.9 [15]. μ is the average percentage of links that connect a unit to those in other communities. Logically, increasing μ weakens the network community structure. When $\mu = 0.9$, the networks are quasi-random (see below).

Once found that these LFR benchmarks generated networks with communities of very similar sizes, we decided to implement RC benchmarks in which these sizes were more variable. All networks in these benchmarks had 512 units divided into 16 communities. One hundred networks with random community sizes, determined using a broken-stick model [27], were generated. This model provides highly heterogeneous community sizes. Progressive weakening of the community structure of the RC networks, similar to the effect of increasing μ in the LFR networks, was obtained as follows. Initially, all units of each community in the network were fully connected. Then, that obvious structure was progressively blurred, by first randomly removing a certain percentage of edges and then randomly shuffling the same percentage of links among the units. That common percentage, we have called Degradation (D). Thus, D = 10 % means that, first, 10 % of the links present were eliminated and then 10 % of the remaining edges were randomly shuffled among units. Shuffling involved first the random removal of an edge of the graph and then the addition of a new edge between two randomly chosen nodes.

In the LFR and RC benchmarks just described it was possible to compare networks having obvious community structures (generated with low μ or D parameters) with others that were increasingly random. This type of benchmarks, we have called open. We also generated closed LFR and RC benchmarks. In them, links were shifted in a directed way, in order to convert the original community structure of a network into a second, also predefined, structure. In this way, it is possible to monitor when the original structure is substituted by the final one according to the solutions provided by S_{max} or Q_{max}. In the LFR and RC closed benchmarks, the starting networks were the same described in the previous paragraphs, with $\mu = 0.1$ (LFR) or D = 0 (RC) respectively, and the final networks were obtained by randomly relabeling the nodes. Therefore, the initial and final networks had identical community structures but the nodes within each community were different. Conversion (C) is defined as the percentage of links exclusively present in the initial network that are substituted by links only present in the final one (i. e. C = 0: initial structure present; C = 100: final structure present).

NMI symmetry as a measure of performance in closed benchmarks

In our closed benchmarks, a peculiar symmetrical behavior of NMI values respect to the initial and final partitions is expected. Imagine that a putative optimal partition is estimated according to a given criterion. Let us now consider the following triangle inequality:

$$\frac{NMI_{IE} + NMI_{EF}}{2} \leq \frac{1 + NMI_{IF}}{2} \tag{2.3}$$

where NMI$_{IE}$ is the normalized mutual information calculated for the initial structure (I) and the estimated partition (E), NMI$_{EF}$ is the normalized mutual information for the final structure (F) versus the estimated partition and NMI$_{IF}$ is the normalized mutual information for the comparison between the initial and final structures. Inequality 2.3 holds true if the structures of I, F and E are identical (i. e. both the number and sizes of the communities are the same, but not necessarily are the same the nodes within each community). This follows from the fact that

$$1 + NMI_{XY} \leq \frac{VI_{XY}}{H(X) + H(Y)} \tag{2.4}$$

Where VI_{XY} is the Variation of Information for both partitions [16] and H(X) and H(Y) are the entropies of the X and Y partitions, respectively. Given that VI is a metric [16], it satisfies the triangle inequality

$$VI_{AB} + VI_{BC} \geq VI_{AC} \tag{2.5}$$

If, as indicated, the structures of all partitions are identical, then all their entropies are also identical. In that case, the following inequality can be deduced from formulae 2.4 and 2.5

$$(1 - NMI_{AB}) + (1 - NMI_{BC}) \geq (1 - NMI_{AC}) \tag{2.6}$$

From this inequality, and substituting A, B and C with I, E and F, respectively, formula 2.3 can be deduced. Formula 2.3 therefore means that, provided that I, E and F have the same structure, the average of NMI$_{IE}$ and NMI$_{EF}$ may acquire a maximum value [(1+NMI$_{IF}$)/2]. Inequality 2.3 will also hold approximately true if the entropies of I, E and F are very similar (i. e. many identical communities). In our closed benchmarks the I and F structures are identical, and we progressively convert one into the other. It is thus expected that the optimal partition along this conversion is similar in structure to both I and F. Hence, deviations from the expected average value (1+NMI$_{IF}$)/2 are a cause of concern, as they probably mean that the optimal partition has not been found. On the other hand, finding values equal to (1+NMI$_{IF}$)/2 is a strong indication that the optimal partition has indeed been found.

It is worth noting that, although NMI has been commonly used in this field [10, 14, 15], using VI instead has clear advantages to analyze closed benchmarks: Formula 2.5 can be used instead of Formula 2.3, avoiding considering entropies at all. This is why we evaluated the closed benchmark results both using NMI and VI (see above).

Real networks

Two of the three networks explored, known as *College football* and *Zachary's karate* networks, have been frequently used in the past in the context of community detection [e. g. refs. [7, 12, 23, 24, 28]. The third network derived from the CYC2008 protein complexes database [22]. This database contains information for 408 protein complexes of the yeast Saccharomyces cerevisiae. The protein complex data were converted into 324 non-overlapping complexes by assigning each protein present in multiple complexes to the largest one. This was made to allow for NMI calculations. Once each protein (unit) was assigned to a non-overlapping cluster (community), we downloaded from the BioGRID database [29] the protein-protein interactions (edges) characterized so far for all these proteins. The final graph contained 1604 nodes and 14171 edges.

References

[1] A.L. Barabási and Z.N. Oltvai. Network biology: understanding the cell's functional organization. *Nature Reviews Genetics*, 5:101–113, 2004.

[2] S. Wasserman and K. Faust. *Social network analysis: Methods and applications*, volume 8. Cambridge university press, 1994.

[3] S.H. Strogatz. Exploring complex networks. *Nature*, 410:268–276, 2001.

[4] L.F. Costa, F.A. Rodrigues, G. Travieso, and P.R.V. Boas. Characterization of complex networks: A survey of measurements. *Advances in Physics*, 56:167–242, 2007.

[5] M. Newman. *Networks: an introduction.* Oxford University Press, Inc., 2010.

[6] S. Fortunato. Community detection in graphs. *Physics Reports*, 486:75–174, 2010.

[7] M.E.J. Newman and M. Girvan. Finding and evaluating community structure in networks. *Physical review E*, 69:026113, 2004.

[8] S. Fortunato and M. Barthelemy. Resolution limit in community detection. *Proceedings of the National Academy of Sciences*, 104:36–41, 2007.

[9] V. Arnau, S. Mars, and I. Marín. Iterative cluster analysis of protein interaction data. *Bioinformatics*, 21:364–378, 2005.

[10] A. Lancichinetti, S. Fortunato, and F. Radicchi. Benchmark graphs for testing community detection algorithms. *Physical Review E*, 78:046110, 2008.

[11] D.J. Watts. *Small worlds: the dynamics of networks between order and randomness.* Princeton university press, 2003.

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

[13] R. Aldecoa and I. Marín. Jerarca: Efficient analysis of complex networks using hierarchical clustering. *PloS one*, 5:e11585, 2010.

[14] L. Danon, A. Díaz-Guilera, J. Duch, and A. Arenas. Comparing community structure identification. *Journal of Statistical Mechanics: Theory and Experiment*, P09008, 2005.

[15] A. Lancichinetti and S. Fortunato. Community detection algorithms: a comparative analysis. *Physical Review E*, 80:056117, 2009.

[16] M. Meilă. Comparing clusterings—an information based distance. *Journal of Multivariate Analysis*, 98:873–895, 2007.

[17] M. Rosvall and C.T. Bergstrom. Maps of random walks on complex networks reveal community structure. *Proceedings of the National Academy of Sciences*, 105:1118–1123, 2008.

[18] P. Ronhovde and Z. Nussinov. Multiresolution community detection for megascale networks by information-based replica correlations. *Physical Review E*, 80:016109, 2009.

[19] V.D. Blondel, J.L. Guillaume, R. Lambiotte, and E. Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, P10008, 2008.

[20] J. Duch and A. Arenas. Community detection in complex networks using extremal optimization. *Physical review E*, 72:027104, 2005.

[21] ECJ Pielou. The measurement of diversity in different types of biological collections. *Journal of theoretical biology*, 13:131–144, 1966.

[22] S. Pu, J. Wong, B. Turner, E. Cho, and S.J. Wodak. Up-to-date catalogues of yeast protein complexes. *Nucleic acids research*, 37:825–831, 2009.

[23] W.W. Zachary. An information flow model for conflict and fission in small groups. *Journal of anthropological research*, 33:452–473, 1977.

[24] L.C. Freeman. Finding groups with a simple genetic algorithm*. *Journal of Mathematical Sociology*, 17:227–241, 1993.

[25] J.I. Lucas, V. Arnau, and I. Marín. Comparative genomics and protein domain graph analyses link ubiquitination and rna metabolism. *Journal of molecular biology*, 357:9–17, 2006.

[26] A. Marco and I. Marín. Interactome and gene ontology provide congruent yet subtly different views of a eukaryotic cell. *BMC systems biology*, 3:69, 2009.

[27] R.H. MacArthur. On the relative abundance of bird species. *Proceedings of the National Academy of Sciences of the United States of America*, 43:293, 1957.

[28] M.E.J. Newman. Modularity and community structure in networks. *Proceedings of the National Academy of Sciences*, 103:8577–8582, 2006.

Supplementary Information

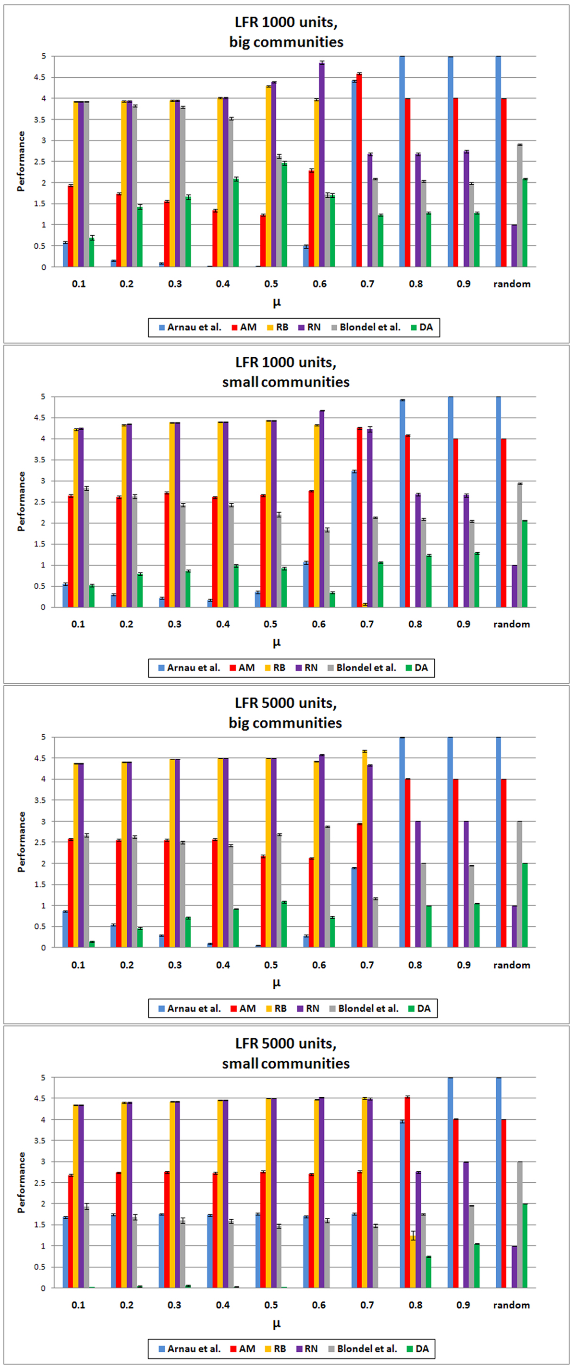


Figure S1: Average performances of the algorithms in the LFR benchmarks. With different network sizes (1000, 5000 units), community sizes (small: 10 to 50 units per community; big: 20-100 units per community) and values of mixing parameter (μ) and for random networks of the same size. After ordering the algorithms from best to worst performance, their ranges were added for the 100 different networks. Performance is defined as P = 6 - average range.

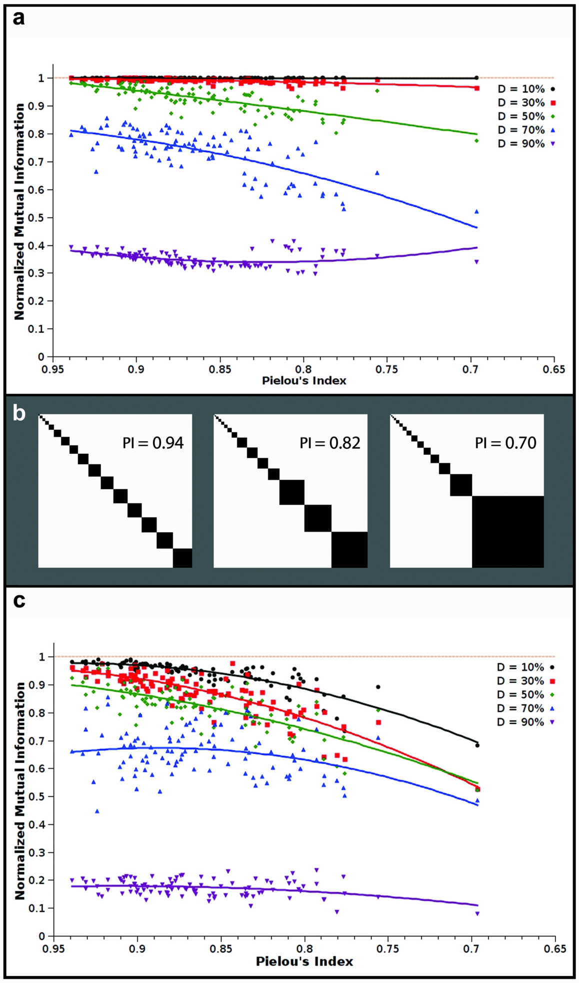


Figure S2: Details of the results for the RC benchmark. a) Normalized Mutual Information values for the 100 networks tested, obtained by S maximization. Given that both a low Pielou's index and high D may alter the original structure of the network, these results would tend to underestimate the real quality of the partition into communities obtained. Lines correspond to the second degree polynomials that best fit the results, which were found to be better than the first degree ones. b) Examples of the relative sizes of communities for different Pielou's indexes, to show the very different structures provided by generating the community sizes according to a broken stick model. c) Summary of the results in the RC benchmark with Q maximization. The results are much worse than those shown in panel a), due to the resolution limit that affects Q values when some communities are small (low Pielou's indexes). Lines again correspond to the best fits according to second degree polynomials.

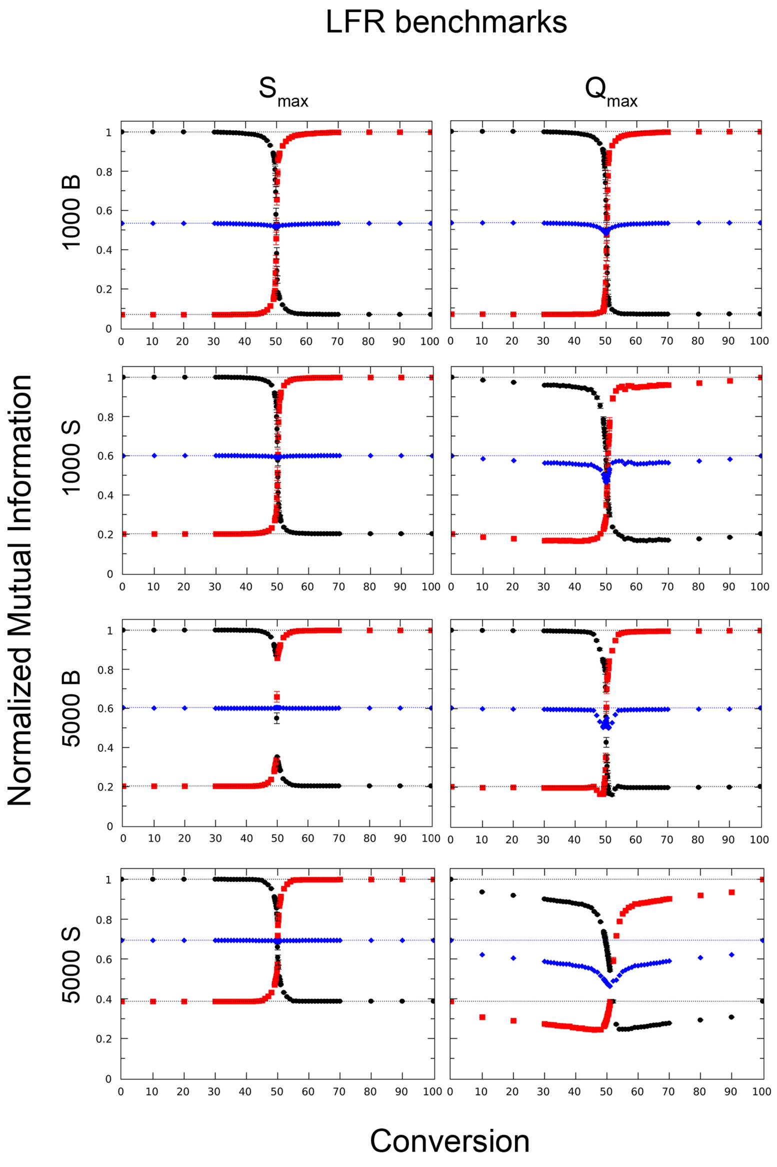


Figure S3: Behavior of S and Q maximization in closed LFR benchmarks. Notice the obvious decrease below $(1+\text{NMI}_{IF})/2$ when Q is maximized.

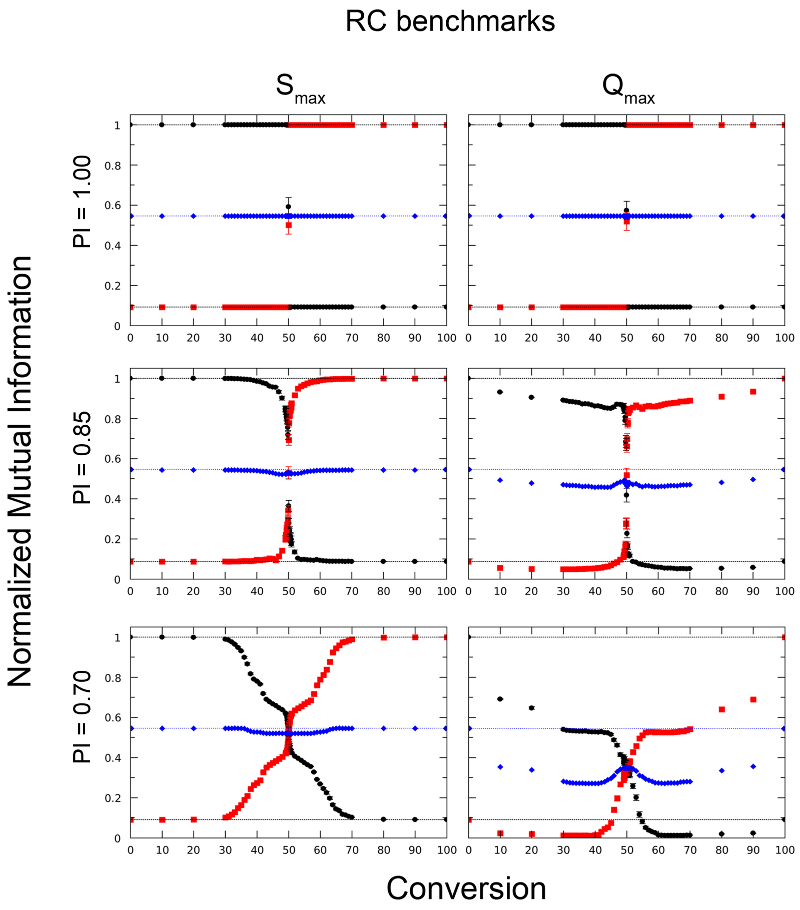


Figure S4: Results for S and Q maximization in the closed RC benchmarks. The behavior of S_{max} is again qualitatively better than the one of Q_{max}, except when all communities are identical.

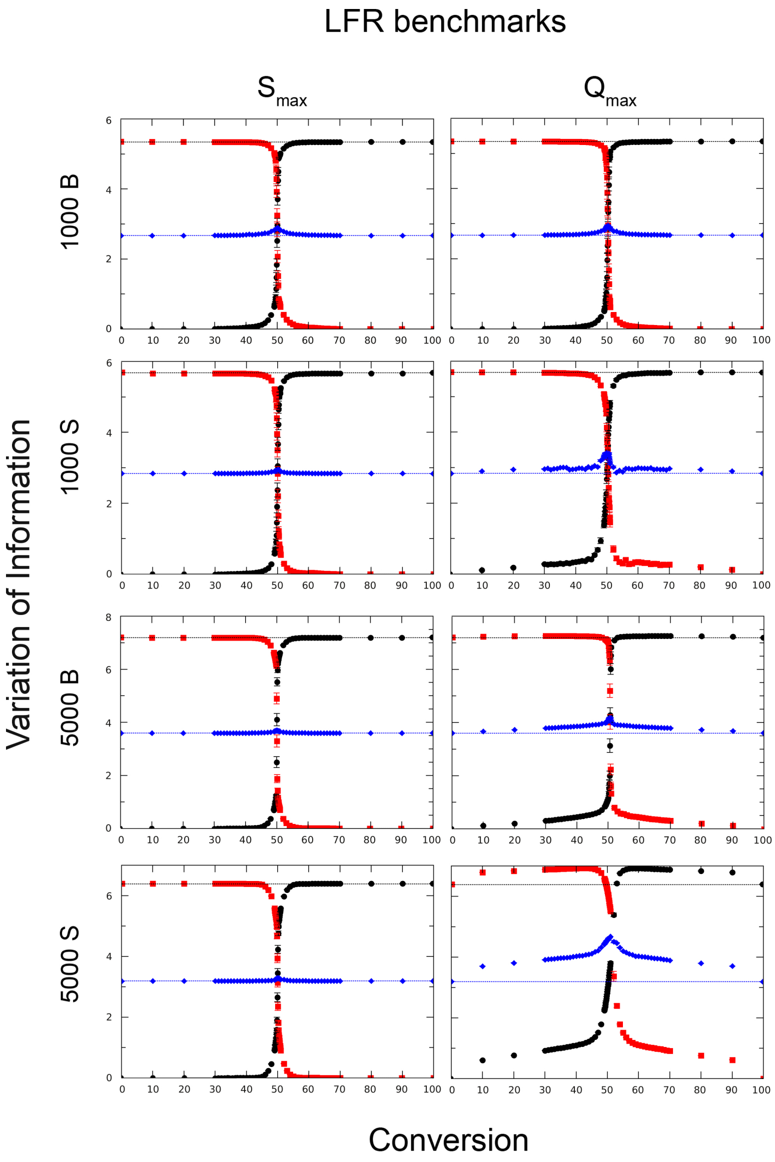


Figure S5: Behavior of S and Q maximization in closed LFR benchmarks using Variation of Information (VI) as a measure of congruence. As it can be deduced from Formula 2.5 in the main text, a good behavior of a parameter implies minimal deviations from the expected value $\text{VI}_{IF}/2$ (blue line). Results are almost identical to those shown in Figure S3 using NMI. S_{max} behavior is clearly better than Q_{max} behavior.

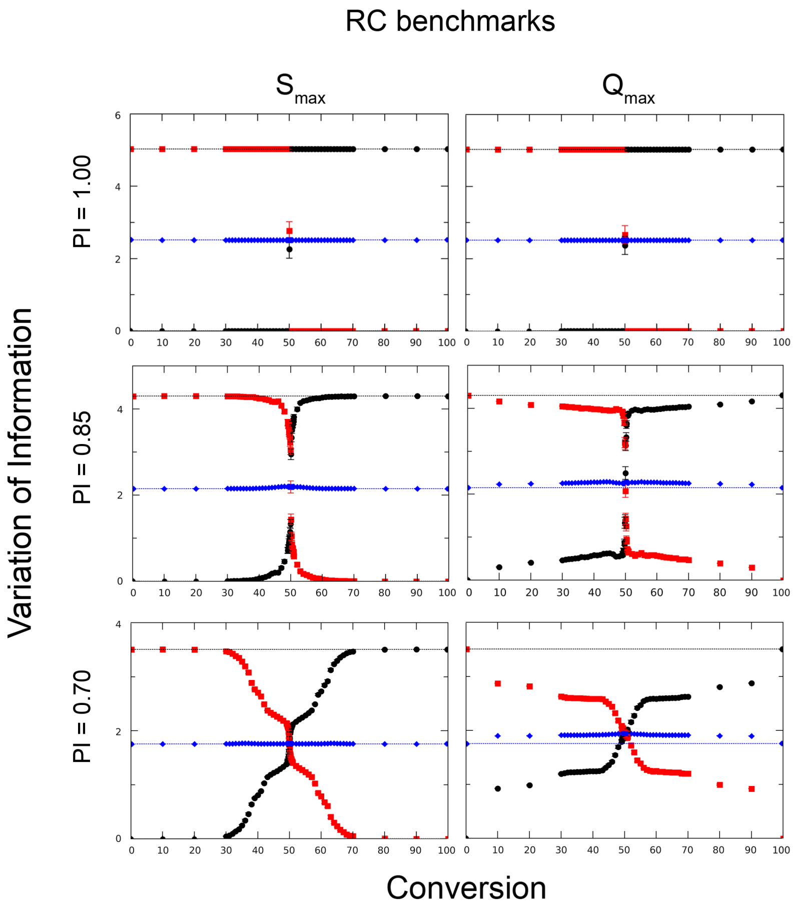


Figure S6: Results for S and Q maximization in the closed RC benchmarks, measured with VI. The behavior of S_{max} is again qualitatively better than the one of Q_{max}, confirming the results shown in Figure S5.

LFR Benchmark	μ	NMI ± s. e. m.		$\mathrm{NMI_S} = 1$ (%)	S_{max} ± s. e. m.	S_{orig} ± s. e. m.
		S	Q			
1000 units, big communities	0.1	1.000 ± 0.000	1.000 ± 0.000	100	9918 ± 52	9918 ± 52
	0.2	1.000 ± 0.000	1.000 ± 0.000	100	7815 ± 47	7815 ± 47
	0.3	1.000 ± 0.000	1.000 ± 0.000	100	6069 ± 36	6069 ± 36
	0.4	1.000 ± 0.000	1.000 ± 0.000	100	4601 ± 25	4601 ± 25
	0.5	1.000 ± 0.000	0.998 ± 0.000	90	3326 ± 23	3325 ± 23
	0.6	0.963 ± 0.003	0.997 ± 0.001	2	2200 ± 17	2133 ± 17
	0.7	0.492 ± 0.003	0.231 ± 0.001	0	1394 ± 4	1237 ± 13
	0.8	0.425 ± 0.002	0.057 ± 0.001	0	1301 ± 2	493 ± 8
	0.9	0.384 ± 0.002	0.027 ± 0.001	0	1298 ± 2	51 ± 2
	random	0.374 ± 0.002	0.020 ± 0.001	0	1291 ± 2	0.46 ± 0.05
1000 units, small communities	0.1	1.000 ± 0.000	0.995 ± 0.000	99	13433 ± 57	13433 ± 57
	0.2	1.000 ± 0.000	0.988 ± 0.001	99	11071 ± 40	11071 ± 40
	0.3	1.000 ± 0.000	0.981 ± 0.001	100	8795 ± 33	8795 ± 33
	0.4	1.000 ± 0.000	0.971 ± 0.001	100	6708 ± 29	6708 ± 29
	0.5	1.000 ± 0.000	0.958 ± 0.002	100	4982 ± 22	4982 ± 22
	0.6	0.999 ± 0.000	0.933 ± 0.003	56	3451 ± 14	3446 ± 14
	0.7	0.858 ± 0.001	0.732 ± 0.016	0	2114 ± 22	2105 ± 11
	0.8	0.562 ± 0.002	0.123 ± 0.002	0	1334 ± 2	1030 ± 6
	0.9	0.505 ± 0.002	0.058 ± 0.001	0	1293 ± 2	238 ± 2
	random	0.486 ± 0.002	0.040 ± 0.001	0	1286 ± 2	0.36 ± 0.03
5000 units, big communities	0.1	1.000 ± 0.000	0.994 ± 0.000	100	81118 ± 121	81118 ± 121
	0.2	1.000 ± 0.000	0.988 ± 0.000	100	67120 ± 100	67120 ± 100
	0.3	1.000 ± 0.000	0.978 ± 0.000	100	54934 ± 91	54934 ± 91
	0.4	1.000 ± 0.000	0.963 ± 0.001	100	43834 ± 77	43834 ± 77
	0.5	1.000 ± 0.000	0.942 ± 0.001	100	33410 ± 52	33410 ± 52
	0.6	1.000 ± 0.000	0.912 ± 0.001	85	24084 ± 41	24083 ± 41
	0.7	0.991 ± 0.001	0.999 ± 0.000	2	15805 ± 34	15777 ± 33
	0.8	0.586 ± 0.001	0.066 ± 0.003	0	8797 ± 4	8578 ± 20
	0.9	0.536 ± 0.001	0.018 ± 0.000	0	8461 ± 4	2777 ± 8
	random	0.517 ± 0.001	0.010 ± 0.000	0	8408 ± 4	0.50 ± 0.05
5000 units, small communities	0.1	1.000 ± 0.000	0.958 ± 0.000	100	99066 ± 112	99066 ± 112
	0.2	1.000 ± 0.000	0.939 ± 0.000	100	82631 ± 94	82631 ± 94
	0.3	1.000 ± 0.000	0.924 ± 0.001	100	67847 ± 91	67847 ± 91
	0.4	1.000 ± 0.000	0.904 ± 0.000	100	54354 ± 77	54354 ± 77
	0.5	1.000 ± 0.000	0.882 ± 0.001	100	41991 ± 49	41991 ± 49
	0.6	1.000 ± 0.000	0.853 ± 0.001	92	30807 ± 40	30807 ± 40
	0.7	0.999 ± 0.000	0.810 ± 0.001	14	20572 ± 27	20563 ± 27
	0.8	0.693 ± 0.002	0.087 ± 0.004	0	10012 ± 66	11599 ± 18
	0.9	0.620 ± 0.001	0.037 ± 0.000	0	8475 ± 4	4205 ± 8
	random	0.600 ± 0.001	0.020 ± 0.000	0	8407 ± 4	0.51 ± 0.04

Table S1: Detailed results obtained for the LFR benchmarks. The values of NMI when S and Q are maximized are indicated, together with the percentage of cases in which NMI = 1 and the values of S_{max} and S_{orig} (i.e. the S value obtained assuming that the original structure is present). Notice that when μ = 0.6-0.7, $S_{max} > S_{orig}$, meaning that the original structure is not the one present anymore. In those cases, NMIs are expected to rapidly decrease, as indeed is observed.

	Degradation	NMI ± s. e. m.		NMIs = 1 (%)	S_{max} ± s. e. m.	S_{orig} ± s. e. m.
		S	Q			
RELAXED CAVEMAN	10	1.000 ± 0.000	0.940 ± 0.005	79	12642 ± 146	12642 ± 146
	20	0.999 ± 0.000	0.914 ± 0.007	43	8613 ± 86	8612 ± 86
	30	0.992 ± 0.000	0.871 ± 0.008	8	4824 ± 39	4823 ± 39
	40	0.972 ± 0.002	0.849 ± 0.008	0	3317 ± 24	3323 ± 24
	50	0.928 ± 0.004	0.821 ± 0.008	0	2197 ± 12	2224 ± 13
	60	0.864 ± 0.006	0.760 ± 0.008	0	1412 ± 6	1447 ± 8
	70	0.735 ± 0.008	0.659 ± 0.008	0	866 ± 4	884 ± 4
	80	0.478 ± 0.003	0.424 ± 0.006	0	653 ± 2	487 ± 3
	90	0.350 ± 0.003	0.172 ± 0.003	0	645 ± 2	201 ± 1
	Degradation	**NMI ± s. e. m.**		**NMIs = 1 (%)**	**S_{max} ± s. e. m.**	**S_{orig} ± s. e. m.**
		S	**Q**			
RANDOM (ERDOS-RENYI)	10	0.327 ± 0.003	0.027 ± 0.001	0	552 ± 3	0.47 ± 0.04
	20	0.327 ± 0.003	0.027 ± 0.001	0	556 ± 2	0.45 ± 0.05
	30	0.332 ± 0.003	0.028 ± 0.001	0	564 ± 2	0.46 ± 0.04
	40	0.323 ± 0.003	0.028 ± 0.001	0	564 ± 2	0.44 ± 0.05
	50	0.327 ± 0.003	0.031 ± 0.001	0	575 ± 2	0.41 ± 0.04
	60	0.328 ± 0.002	0.036 ± 0.001	0	592 ± 2	0.42 ± 0.04
	70	0.334 ± 0.003	0.041 ± 0.001	0	613 ± 2	0.41 ± 0.04
	80	0.331 ± 0.003	0.047 ± 0.001	0	621 ± 1	0.46 ± 0.04
	90	0.215 ± 0.003	0.060 ± 0.001	0	642 ± 2	0.39 ± 0.04

Table S2: Details of the RC benchmark results. Same data as in Table S1, but with variations in the Degradation (D) parameter. Data for random networks of the same size are also included.

Closed benchmarks for network community structure characterization

Rodrigo Aldecoa and Ignacio Marín
Instituto de Biomedicina de Valencia. Consejo Superior de Investigaciones Científicas (IBV-CSIC) Calle Jaime Roig 11. Valencia, Spain

Physical Review E 85, 026109 (2012)

Characterizing the community structure of complex networks is a key challenge in many scientific fields. Very diverse algorithms and methods have been proposed to this end, many working reasonably well in specific situations. However, no consensus has emerged on which of these methods is the best to use in practice. In part, this is due to the fact that testing their performance requires the generation of a comprehensive, standard set of synthetic benchmarks, a goal not yet fully achieved. Here, we present a type of benchmark that we call "closed", in which an initial network of known community structure is progressively converted into a second network whose communities are also known. This approach differs from all previously published ones, in which networks evolve toward randomness. The use of this type of benchmark allows us to monitor the transformation of the community structure of a network. Moreover, we can predict the optimal behavior of the variation of information, a measure of the quality of the partitions obtained, at any moment of the process. This enables us in many cases to determine the best partition among those suggested by different algorithms. Also, since any network can be used as a starting point, extensive studies and comparisons can be performed using a heterogeneous set of structures, including random ones. These properties make our benchmarks a general standard for comparing community detection algorithms.

Introduction

Network analysis offers a powerful approach to solve problems in many scientific fields, including physics, biology, and sociology [1–4]. Community structure is a significant property of these networks. A community can be loosely defined as a set of nodes that are more densely connected among themselves than with the rest of the network. The importance of community structure characterization derives from the fact that all nodes in a community are expected to share common attributes, features, or functional connections (reviewed in [5]). Many algorithms and methods have been proposed for extracting the optimal partition of a network into communities. While some of them try to improve a global quality function such as its Modularity [6] or Surprise [7], others search for the optimal partition by minimizing the compression of the information that best describes the network [8], minimizing the Hamiltonian of a Potts-like spin model that represents the graph [9], or deducing the maximum-likelihood model that best fits the structure of the network [10], to name just a few examples. However, none of these algorithms achieves maximal results in all situations. Their performance varies greatly, depending on the topological parameters of the analyzed network [7, 11].

In order to compare the performance of community detection algorithms, several benchmarks have been proposed. The first ones were based on the planted one-partition model [12]. The most popular among them is the Girvan and Newman (GN) benchmark [13], in which a network of 128 nodes is divided into four communities of equal size where each node is connected with 16 other members of its own community. This starting graph can then be progressively degraded by replacing links within communities with links between them, keeping constant the average node degree. The relaxed caveman (RC) benchmarks [7, 14, 15] are similar in concept. In them, the starting network is formed by a set of cliques of variable sizes, and a degradation process identical to that already described for the GN benchmark is performed. Notice that GN and RC communities are, by definition, Erdős-Rényi subgraphs [16] in which, all throughout the degradation process, each pair of nodes is linked with the same probability p. This makes those benchmarks rather inappropriate for representing real-world networks since the latter exhibit much more heterogeneous degree distributions [17, 18]. With this idea in mind, Lancichinetti, Fortunato, and Radicchi developed a novel type of benchmark, called LFR [19], in which both the sizes of the communities and the distribution of node degrees are adjusted to follow power laws. In LFR benchmarks, the fraction of links μ that a node shares with nodes in other communities is tunable. Increasing μ (often called the "mixing parameter") generates an analogous behavior to that of the degradation process described for GN and RC benchmarks, i.e., the proportion of intercommunity links grows and the original communities gradually disappear. We refer to all of these benchmarks (GN, RC, and LFR) as "open", given that the final outcome is "open-ended"(i.e., the precise final community structure of the network is undetermined).

In this paper, we describe in detail a novel type of benchmark (referred to as “closed”) that is based on the conversion of a network of known community structure into a second network whose communities are also known. We already introduced the concept of a closed benchmark in a previous work [7], and we showed how this type of benchmark can be successfully used to compare community detection algorithms. Here, we explain it in detail, give some examples of its performance, and discuss its potential and the significant advantages it presents over the aforementioned open benchmarks. We show that the guided evolution of the networks to a closed end enables us to accurately monitor the transformation progress and to evaluate the goodness of a partition at any moment of the process.

Features of the closed benchmarks

The main concept behind the closed benchmarks is the directed conversion of a network into another one by means of edge rewiring. The starting point is a network whose community structure is known *a priori*. Any type of graph and community structure is valid as an initial network. The algorithm then generates a second, “final”network. The initial and final networks are precisely related. The community structure of the final network is identical to the initial one, but the labels of the nodes are randomly mapped from the former to the latter. Converting the initial network into the final one involves rewiring links in a directed manner, a process depicted in Fig. 1. The details of the procedure are as follows:

1. Links present in both the initial and the final networks will not be rewired.
2. At each step, one of the rewirable links is removed and subsequently a new link is added between two nodes connected only in the final network. Conversion (C) is defined as the percentage of rewirable links modified at a particular point of the process of converting the initial network structure into the final one.
3. At any point of this conversion process, the network can be saved for later analyses. Therefore, a wide set of intermediate structures to test the behavior of community detection algorithms can be obtained.
4. The process stops when the final structure is reached.

A significant feature of the closed benchmarks is that, during the conversion process and because of the directed rewiring of the links, we are approaching the final structure at the same rate that we are leaving the initial one. Calling D the distance between both networks, we can assert that the structure at a distance x from the start is also at a distance $D - x$ from the end of the benchmark. This fact, together with the identical topology of both ends, produces a set of structures that is symmetrical about the $50\,\%$ conversion point. That is, when $C = 50\,\%$, the structure of the network is, on average, at the same distance from both the initial and the final networks. Given these patterns of network evolution, we

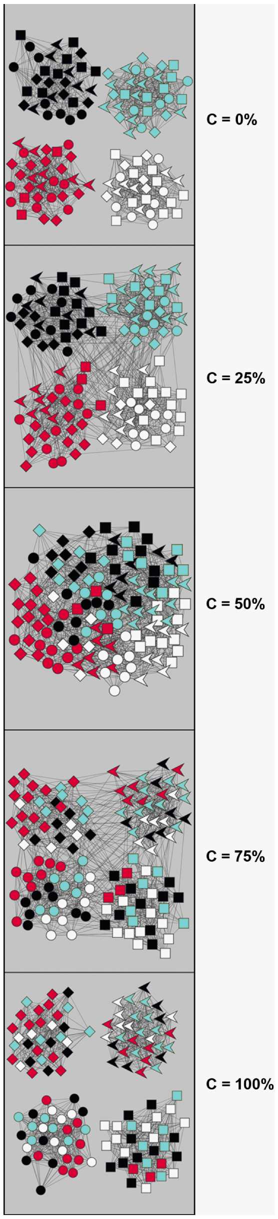


Figure 1: Transformation process in a closed benchmark. In this case, the starting network is the GN benchmark. Links are progressively rewired from the initial (C = 0 %) to the final network (C = 100 %). Nodes color is defined by the initial community to which they belong, whereas their shape corresponds to the final community in which they are contained.

can assume that its community structure undergoes a similar behavior. As we will describe below, this behavior is central to the evaluation of partitions in closed benchmarks.

Any benchmark is associated with one or several measures of performance. In the case of clustering comparison, several methods, based on counting pairs, cluster matching, or information-theory based indexes, have been developed (reviewed in [5, 20]). Among the latter type, the variation of information (V) [21] is an information-based distance useful for measuring the dissimilarity between two partitions, A and B (V_{AB}). In our context, we consider that it has clear advantages over other criteria, especially its metric nature. This property implies that V is positive-definite, a symmetric distance (which is a highly desirable property when comparing clusterings), and, more important for our purposes, it satisfies the triangle inequality [21]. This last fact turns out to be very useful for closed benchmarks evaluation. In these benchmarks, we have two known community structures, those of the initial (I) and final (F) networks. Moreover, the method generates a set of intermediate, estimated structures (E) whose communities can also be determined. We can deduce from the V triangle inequality the following formula:

$$V_{AB} + V_{BC} \geq V_{AC} \tag{2.7}$$

Hence, the sum of V_{IE} and V_{EF} is lower bounded by V_{IF} , which is constant, given that the partitions of the initial and final networks are fixed. If the rewiring of the network has not yet started, the optimal estimated partition is the same as the initial one, $I = E$, and therefore $V_{IE} = 0$ and $V_{EF} = V_{IF}$, satisfying the equality in Eq. 2.7. When the conversion starts, and because the network approaches the final structure at the same rate that it leaves the initial one, V_{IE} should increase as much as V_{EF} decreases. Therefore, unless the structure of the network becomes very different from both the initial and final structures along the conversion process (e.g., as described in the next paragraph), this should make the equality $V_{IE} + V_{EF} = V_{IF}$ true all along the conversion of the initial into the final structure. A significant deduction is that if, for a given estimated partition E, the sum of V_{IE} and V_{EF} deviates from the constant value V_{IF} , then E may not be the optimal partition [7]. Thus, deviation from the expected V_{IF} value may indicate a suboptimal performance of a given algorithm.

If third-party structures, very different from the initial and final ones, are formed along the conversion process, we can find $V_{IE} + V_{EF} > V_{IF}$ even if the partition is optimal. This can be illustrated assuming that the intermediate structure becomes fully random. Two situations are then possible, depending on the density of links in the graph. If, at some point of the rewiring, the intermediate structure becomes a single community containing all the nodes -as expected in a random graph with a high density of links- then $V_{IE} = H(I)$ and $V_{EF} = H(F)$, where $H(I)$ and $H(F)$ are the entropies of the initial and final partitions. Given that $V_{IF} = H(I) + H(F) - 2M(I, F)$, where $M(I, F)$ is the mutual information between the initial and final partitions, we have that $V_{IE} + V_{EF}$ must be somewhat larger than V_{IF}. This derives from the fact that $M(I, F) = 0$ only if I and F are independent, which is not the case here. On the other hand, if the density of links is low and the network is randomized, the community structure may approach a situation in which each node is isolated in a different community. If this is true, it can be shown that $V_{IE} = logN - H(I)$ and $V_{EF} = logN - H(F)$, where N is the total number of nodes. In this case, we will find $V_{IE} + V_{EF} \gg V_{IF}$. Thus, if an algorithm is performing perfectly well ($V_{IE} + V_{EF} = V_{IF}$) until a certain conversion percentage, and if, when conversion progresses further, we find $V_{IE} + V_{EF} > V_{IF}$, this may be due to two reasons: (i) a bad performance of the algorithm with poorly defined community structures, (ii), the emergence of a third-party, potentially random, community structure. This interesting situation will be illustrated in a particular case below.

Tests

Configuration

As mentioned above, the particular features of a network can greatly influence the ability of a given algorithm to detect its community structure. For this reason, we performed tests on computer-generated networks that varied in size, node degree distribution, number of communities, and also community sizes. This last parameter has been shown to be crucial in community detection [7,11]. There are two main reasons for the significant effect of community size variation. First, networks presenting a skewed distribution of community sizes are more rapidly degraded than those with equally sized communities because of the quick destruction of small clusters. Second, a skewed distribution may greatly affect the performance of particular algorithms. For example, any algorithm maximizing a popular global measure for community detection, Newman and Girvan's modularity (Q), will have trouble detecting small communities, given that Q is affected by a resolution limit [22].

A suitable way to measure and compare the distribution of community sizes is using Pielou's index (P), which quantifies how similar are the groups into which a system is divided. This index takes a value of 1 for equal-sized groups and decreases with increasing size variance [23]. In this study, we chose as starting points four different synthetic networks with different P values that correspond to those of already published open benchmarks. We will name them according to the following convention: (i) Girvan-Newman (GN) [13]: already mentioned above. A network of 128 nodes is divided into four communities of equal size ($P = 1$). Nodes are connected only with members of their own community with an average degree of 16. (ii) Lancichinetti-Fortunato-Radicchi network with small communities (LFR$_S$) [11,19]: a network of 5000 nodes. The average degree of the nodes is 20, their maximum degree is 50, the exponent of the degree distribution is -2, and the exponent of the community sizes distribution is -1. The sizes of the

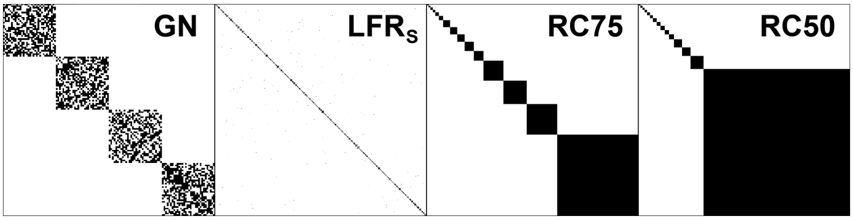


Figure 2: Graphical view of the adjacency matrices of the four initial networks used in the tests. Nodes are ordered according to the communities to which they belong. Black indicates that two nodes are connected. Differences in relative community sizes are evident. In the GN network, the nodes of the four equal-sized communities are sparsely connected. The groups in the LFR$_S$ are also sparse. However, there are so many of them (195) that visualization is difficult at this resolution level. The RC initial networks (RC75 and RC50) are formed by 16 cliques and the distribution of community sizes is highly skewed, especially in RC50, where a single community dominates the network.

communities vary between 10 and 50 nodes (hence the term "small communities"). Among the many networks that can be generated with these parameters, we chose one containing 195 communities of similar sizes ($P = 0{,}98$). (iii) relaxed caveman [14] with Pielou's index = 0.75 (RC75): Because a more skewed distribution of community sizes was required to analyze the behavior of the algorithms in a wider range of network structures, we generated a network of 512 nodes with $P = 0{,}75$, which corresponds to a division into 16 communities, each of them including from 2 to 196 nodes. In the RC75 configuration, the initial network consisted of unconnected communities, each one maximally connected internally, i.e., forming a clique. (iv) Relaxed caveman, $P = 0{,}50$ (RC50): this has an even more extreme variation in community sizes. The initial network is also comprised of 512 nodes forming 16 cliques, but now the largest one contains 354 nodes. Figure 2 graphically displays the pattern of connections of each of these four initial networks. Once obtained, they were progressively modified by increasing C, finally obtaining from each one a set of 101 network structures spanning the whole range from $C = 0$ (initial structure present) to $C = 100$ (final structure present). The corresponding open benchmarks, with the same starting community structures and progressive degradation toward randomness, were also analyzed following standard methods described in previous papers (see, e.g., [13, 14, 19]). We also discuss below in some detail closed benchmarks with random initial structures.

Algorithms

Two community detection algorithms that have shown an excellent performance in recent studies, namely Infomap [8] and SCluster [15], were used in this work. Infomap understands finding the community structure of a network as an information compression problem, detecting communities while compressing the topology of the network. It has achieved excellent results on the LFR benchmarks [7, 11]. On the other hand, SCluster uses a completely different approach. Using iterative hierarchical clustering [15, 24], the algorithm computes the pairwise distances of the nodes from partial clustering solutions. Subsequently, it constructs a hierarchical tree from which the partition of maximum Surprise [7] is chosen as the optimal solution. Surprise is a quality function that estimates the goodness of a partition based on the comparison between the graph and the null model generated by a random distribution of links [7, 24]. SCluster has demonstrated an ability to extract high-quality partitions when dealing with networks whose communities strongly vary in size [7, 15]. Moreover, as a third way to extract the best clustering of the network, we selected from the Infomap and SCluster solutions the one with the highest surprise, given that we showed before that surprise maximization not only qualitatively outperformed maximizing the most commonly used global index, namely Newman and Girvan's Q, but it also improved the solutions generated by any single algorithm [7].

Figure 3 illustrates the results of the three methods in our four closed benchmarks. Each partition estimated along the conversion process is compared, using the variation of information, with both the initial (black circles) and the final (red squares) community structures. $V = 0$ means that the partitions compared are exactly the same. We previously mentioned how the sum of the variation of information from an estimated point to the initial and to the final optimal partitions ($V_{IE} + V_{EF}$) should optimally be constant and equal to the V between the initial and the final partition (V_{IF}). For visualization reasons, half of this sum ($\overline{V} = [V_{IE} + V_{EF}]/2$) is shown in the figures as a dashed line. $\overline{V} = V_{IF}/2$ is expected if the partition is optimal.

Results

The plots show how different is the community detection process, depending on both the algorithm applied and the topology of the network analyzed. When using the GN network as an input, Infomap performs very well [Fig. 3(a)]. The variation of information between the initial and the estimated partition (V_{IE}, black dots) is zero or near zero along the first half of the benchmark. Moreover, when the conversion (C) breaks the 50 % mark, the V between the estimated and the final partition (V_{EF}, gray squares) behaves in the same way. That is, the algorithm recognizes the initial structure until $C = 49\,\%$ and the final one above $C = 51\,\%$. This is not the case when applying SCluster [Fig. 3(e)], which only recognizes the initial partition up to $C = 30\,\%$ and starts recognizing the final partition beyond $C = 70\,\%$. As expected, $\overline{V}$ graphically shows this different quality in the performance of both algorithms. While in the Infomap plot $\overline{V}$ falls in an almost straight line, matching $V_{IF}/2$, the partitions estimated with SCluster produce a significant deviation from that line in the interval $30 - 70\,\%$, where we already detected that the communities were poorly estimated.

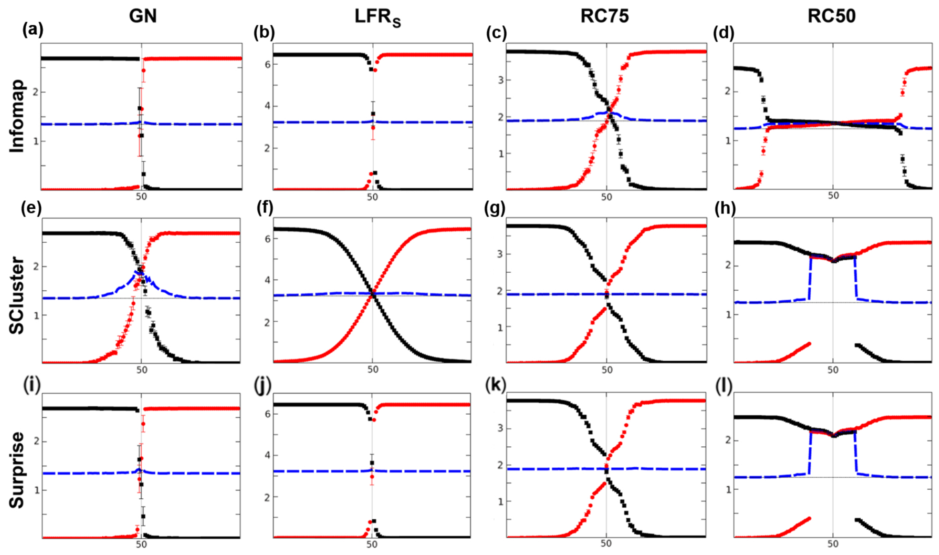


Figure 3: Variation of information behavior in the four closed benchmarks used in this study. Black circles depict the V between the initial and the estimated partition (V_{IE}). Red (gray) squares show the V between the estimated and the final partition (V_{EF}). $\overline{V}$ appears as a dashed line, which should follow a straight line if the performance of the algorithm is optimal during the whole process of conversion (i.e., $V_{IE} + V_{EF} = V_{IF}$).

When the input of the benchmark is the LFR$_S$ network, Infomap also produces a symmetrical plot, with $\overline{V}$ almost perfectly matching $V_{IF}/2$ [Fig. 3(b)]. SCluster also shows in this case a symmetrical performance, although with a slight deviation from the optimal values [Fig. 3(f)], i.e., working again worse than Infomap. In these first two examples, the sizes of the communities are equal or very similar ($P \approx 1$), and they are expected to be degraded, on average, at the same time. The original partition is thus present during the first half of the conversion (giving $V_{IE} \approx 0$), and then the community structure suddenly swaps to the final one (and then $V_{EF} \approx 0$). On the other hand, when analyzing networks with a strongly skewed distribution of community sizes (RC75, RC50), the performance of the algorithms radically changes. In the RC75 test, Infomap exhibits a nonsymmetrical behavior [Fig. 3(c)], with $\overline{V} > V_{IF}/2$ when $C = 40 - 60\,\%$. On the contrary, SCluster shows a symmetrical pattern with $\overline{V} = V_{IF}/2$ [Fig. 3(g)]. We can see how the V between the initial and the estimated partition (V_{IE} , black circles) is equal to zero until around the 30 %, at which point it starts to increase. It is very significant that, in an open benchmark (see, e.g., Refs. [13, 14, 19]), this would be the only available information. Thus we might conclude that from $C = 30\,\%$ on, these two algorithms fail to recognize the optimal partition. However, a bad al-

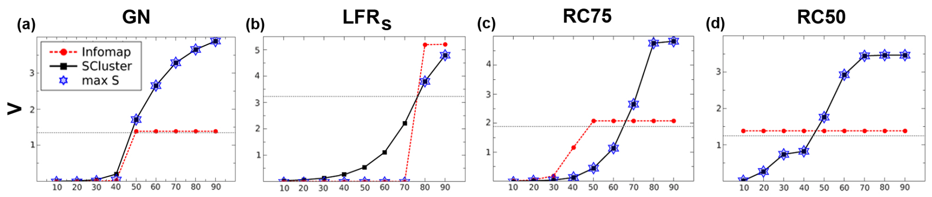


Figure 4: Open benchmarks with starting structures identical to the initial structures of the closed benchmarks shown in Fig. 3. These structures are progressively degraded by randomly shuffling links. The percentage of rewired links is indicated on the x axis. The dashed line indicates the $V_{IF}/2$ value of the corresponding closed benchmark. Stars indicate the partitions with the highest surprise values.

gorithm performance is not the only explanation for such patterns. Alternatively, it is possible that the initial partition must not be detected as optimal anymore because the community structure has changed. The closed benchmarks offer a solid way to check if this latter hypothesis is correct. In Figs. 3(c), 3(g), and 3(k), we can see that, although V_{IE} soon starts to grow, V_{EF} begins to decrease at the same rate. That is, the community structure of the initial partition is shifting toward the final one much before the $C = 50\,\%$ mark is passed, a pattern that is due to the rapid destruction of small communities, typical of benchmarks with low P. This behavior was impossible to check in any of the benchmarks published so far, although it is critical for algorithm evaluation. Now, we can assert that the behavior of SCluster is optimal, given that V follows a straight line: it satisfies the equality in Eq. 2.7 during the whole conversion process. In the last case, RC50, the performance of the algorithms follows a pattern that is a bit different from the rest of the benchmarks. Infomap seems to rapidly collapse, with $\overline{V}$ moving away from the optimal straight line, when $C \geq 10 - 12\,\%$ [Fig. 3(d)]. In the case of SCluster [Fig. 3(h)], $\overline{V}$ values are close to the line quite a bit longer (C around 30 %), but then the algorithm starts recognizing third-party structures, far away from both the initial and final partitions ($\overline{V} > V_{IF}/2$). These behaviors are due to the extremely skewed distribution of community sizes, with a very large group that dominates the network [Fig. 2(d)]. For these reasons, a quasi-random graph is formed as the conversion process of the benchmark approaches 50 %. Infomap interprets this situation as if most of the network is included into a single community. Hence, as we discussed above, $\overline{V}$ approximates $H(I)$ (which in this example takes a value of 1.38). SCluster, on the other hand, interprets the network structure as including many singletons. Therefore, $\overline{V}$ becomes much larger than $V_{IF}/2$ for the reasons previously discussed.

Figures 3(i)-3(l) show the evolution of each benchmark using as the estimated partition that with the highest Surprise between the solutions provided by the two algorithms. As expected [7], this approach always selects the best partition between those two. The equality in Eq. 2.7 is satisfied all along the first three networks. In the fourth case, the pattern is identical to that produced by SCluster. The Surprise values of the RC50 benchmark suggest that the SCluster interpretation, defining many small clusters of the quasi-random intermediate structure generated when $C > 30\,\%$, is preferable to the one suggested by Infomap (dominated by a single huge cluster), in good agreement with the fact that SCluster is, as already indicated above, performing better in this benchmark than Infomap in the adjacent conversion range ($30\,\% \geq C \geq 12\,\%$). For comparative purposes, we also generated the corresponding open benchmarks, which start with the same structures as those of our closed benchmarks but are then progressively degraded toward undetermined, random structures by rewiring their links [13, 14, 19]. Figure 4 shows the variation of information between the original partition and those obtained by the SCluster and Infomap algorithms. The partition with maximum Surprise is marked with a star. We have also depicted in Fig. 4 the value of $\overline{V}$ in the corresponding closed benchmarks (dashed line). As found before in related cases [7], neither of the two algorithms is the best in all situations. If we use the surprise values as a guide, it can be seen that SCluster improves upon Infomap when degradation is very high and systematically in the benchmarks with the lowest Pielou's indices (RC75 and RC50), while Infomap works better when degradation is low and Pielou's index is high (see GN and LFR_S benchmarks). This situation is fundamentally caused by Infomap solutions often consisting in single communities (this happens in all the cases shown in Fig. 4, in which the Infomap V values are above the $\overline{V}$ dashed lines). These results for the open benchmarks are fully compatible with those shown in Fig. 3 for closed benchmarks.

The comparison of the values of $\overline{V}$ in Figs. 3 and 4 enables us to precisely understand the relationships between both types of benchmarks. Looking at the dashed lines in those figures allows us to estimate the approximate difficulty of reconstructing the community structure present in the closed benchmarks when compared with the open ones. Thus, we can see that $C = 50\,\%$ in the GN closed benchmark corresponds to a rewiring percentage of more than $40\,\%$ in the corresponding open GN benchmark, while $C = 50\,\%$ in the LFR_S, RC75, and RC50 closed benchmarks may correspond, respectively, to rewiring about $80\,\%$, $60\,\%$, and (this can be ascertained less precisely) 50-70 % of the links in the corresponding open benchmark. Thus, the GN, LFR_S, and RC75 closed benchmarks always have a substantial level of structure, which explains the good fit to the $\overline{V}$ value observed in Fig. 3.

Random networks can also be used as starting points for a closed benchmark. The comparison with these random network-based benchmarks may contribute to determine whether or not a given network has a statistically significant community structure, a topic that has recently received some attention [25, 26]. To address this issue, we generated four types of random graphs, each of them having the same number of nodes and edges as one of the initial networks described above (GN, LFR_S, RC75, and RC50), but randomly distributed. Given that, for

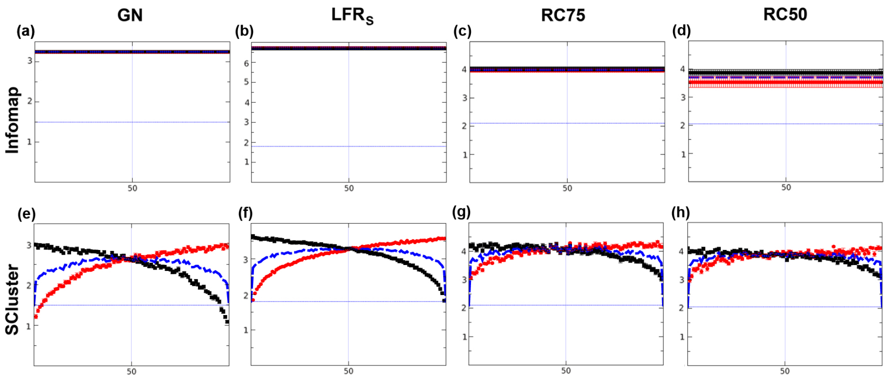


Figure 5: Random networks with the same number of nodes as the corresponding closed benchmarks indicated on top. As in Fig. 3, the dashed line corresponds to the $\overline{V}$ value, while red (gray) dots correspond to V_{EF} values and black squares to V_{IE} values. The values of V_{IE}, V_{EF}, and $\overline{V}$ largely/fully coincide in Infomap analyses, appearing as a single line or close parallel lines. Notice that as soon as conversion starts, $V_{IE} + V_{EF} \gg V_{IF}$. Differences between Infomap and SCluster are due to the different way they interpret the random structures present, i.e., as a single cluster (Infomap) or as many individual clusters (SCluster).

generating a closed benchmark, a community structure must be assumed *a priori*, Infomap and SCluster were tested in those random networks and the community structure with the highest surprise value was selected. Figure 5 shows the results of closed benchmarks generated using the four random networks. As occurred above, Infomap returns partitions in which all nodes [Figs. 5(a)-5(c)], or at least more than 90 % of the nodes [Fig. 5(d)], belong to one community. The $\overline{V}$ observed is the entropy of the initial (or final) partition $H(I) = H(F)$, given that, if all nodes are in a single community, $H(E) = 0$. On the other hand, SCluster generates solutions with a high number of communities [Figs. 5(e)-5(h)], interpreting that even a random graph contains a certain degree of community structure. In these random graph benchmarks, an interesting point is to appreciate the extremely fast degradation of the partitions when only 1 % of the links have been rewired (Fig. 5). When compared with its analogous nonrandom network, V_{IE} rises instantaneously, which is the behavior expected for networks in which communities are barely defined. This kind of comparison between variation of information patterns may enable us to evaluate the robustness of a network, similarly to what has been done using other methods [25].

Discussion

The development of methods that can accurately detect community structure in networks is critical in many scientific fields, since they can reveal deep underlying relationships among the elements of a system. Therefore, it is very important to compare and evaluate such methods against a set of synthetic benchmarks in order to select one method, or a combination of methods, that can produce reliable results when analyzing real-world networks. Several standard benchmarks for testing community detection algorithms have been proposed, most of them of the class we called open: they start with a network of well-defined community structure and then the structure is degraded by randomly rewiring links [13, 14, 19]. During this process, the communities gradually disappear toward an "open end" when the precise community structure is undetermined. This type of benchmark is useful for comparing the relative performance of algorithms but inadequate for assessing their intrinsic quality (i.e., whether the solutions provided are optimal or not). In this paper, we have fully described the closed benchmarks, which also degrade an initial network with defined communities, but this time evolving toward a second, known network structure. This evolution is produced by a directed rewiring of the links from the initial to the final network, and it enables us to control the progression of the structure between both ends. We have also shown that the variation of information provides valuable information about the goodness of a partition and its possible optimality: the configuration of our closed benchmarks allows us to lower bound the expected V value using the triangle inequality that the metric must satisfy. Another relevant improvement over the available open benchmarks is the fact that any network can be used as input for the degradation process, enabling us to carry out extensive studies over a wide variety of network topologies. These features clearly represent qualitative improvements over the benchmarks published so far. The comparisons of open and closed benchmarks, or of networks of known structure and random networks (Figs. 4 and 5), are also interesting ways to further develop this methodology.

As we have shown, there may be scenarios with very skewed distributions of community sizes, such as the RC50 network (Fig. 3), where the equality in Eq. 2.7 is not satisfied during the whole process of conversion. Nevertheless, this behavior in such extreme networks does not diminish the potential of our approach because, even then, there are several conditions that a good algorithm must fulfill. First, when 50 % of the links have been rewired, V_{IE} must be, on average, equal to V_{EF}. Second, the initial partition has to be recognized better than the final one during the first half of the benchmark and, from there on, the behavior should be exactly the opposite. Third, a good algorithm will provide solutions with $V_{IE} + V_{EF} = V_{IF}$ along a longer range of the conversion process than a bad one. In summary, the properties of the closed benchmarks make them highly valuable for the development and evaluation of computational methods to effectively characterize the community structure of a network.

References

[1] S. Wasserman and K. Faust. *Social network analysis: Methods and applications*, volume 8. Cambridge university press, 1994.

[2] S.H. Strogatz. Exploring complex networks. *Nature*, 410:268–276, 2001.

[3] A.L. Barabási and Z.N. Oltvai. Network biology: understanding the cell's functional organization. *Nature Reviews Genetics*, 5:101–113, 2004.

[4] M. Newman. *Networks: an introduction.* Oxford University Press, Inc., 2010.

[5] S. Fortunato. Community detection in graphs. *Physics Reports*, 486:75–174, 2010.

[6] M.E.J. Newman and M. Girvan. Finding and evaluating community structure in networks. *Physical review E*, 69:026113, 2004.

[7] R. Aldecoa and I. Marín. Deciphering network community structure by surprise. *PloS one*, 6:e24195, 2011.

[8] M. Rosvall and C.T. Bergstrom. Maps of random walks on complex networks reveal community structure. *Proceedings of the National Academy of Sciences*, 105:1118–1123, 2008.

[9] P. Ronhovde and Z. Nussinov. Multiresolution community detection for megascale networks by information-based replica correlations. *Physical Review E*, 80:016109, 2009.

[10] MEJ Newman and EA Leicht. Mixture models and exploratory analysis in networks. *Proceedings of the National Academy of Sciences*, 104:9564–9569, 2007.

[11] A. Lancichinetti and S. Fortunato. Community detection algorithms: a comparative analysis. *Physical Review E*, 80:056117, 2009.

[12] A. Condon and R.M. Karp. Algorithms for graph partitioning on the planted partition model. *Random Structures and Algorithms*, 18:116–140, 2001.

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

[14] D.J. Watts. *Small worlds: the dynamics of networks between order and randomness*. Princeton university press, 2003.

[15] R. Aldecoa and I. Marín. Jerarca: Efficient analysis of complex networks using hierarchical clustering. *PloS one*, 5:e11585, 2010.

[16] P. Erdős and A. Rényi. On the evolution of random graphs. *Publ. Math. Inst. Hungar. Acad. Sci.*, 5:17–61, 1960.

[17] A.L. Barabási and R. Albert. Emergence of scaling in random networks. *Science*, 286:509–512, 1999.

[18] S. Boccaletti, V. Latora, Y. Moreno, M. Chavez, and D.U. Hwang. Complex networks: Structure and dynamics. *Physics Reports*, 424:175–308, 2006.

[19] A. Lancichinetti, S. Fortunato, and F. Radicchi. Benchmark graphs for testing community detection algorithms. *Physical Review E*, 78:046110, 2008.

[20] N.X. Vinh, J. Epps, and J. Bailey. Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance. *The Journal of Machine Learning Research*, 11:2837–2854, 2010.

[21] M. Meilă. Comparing clusterings—an information based distance. *Journal of Multivariate Analysis*, 98:873–895, 2007.

[22] S. Fortunato and M. Barthelemy. Resolution limit in community detection. *Proceedings of the National Academy of Sciences*, 104:36–41, 2007.

[23] ECJ Pielou. The measurement of diversity in different types of biological collections. *Journal of theoretical biology*, 13:131–144, 1966.

[24] V. Arnau, S. Mars, and I. Marín. Iterative cluster analysis of protein interaction data. *Bioinformatics*, 21:364–378, 2005.

[25] B. Karrer, E. Levina, and M.E.J. Newman. Robustness of community structure in networks. *Physical Review E*, 77:046119, 2008.

[26] A. Lancichinetti, F. Radicchi, J.J. Ramasco, and S. Fortunato. Finding statistically significant communities in networks. *PloS one*, 6:e18961, 2011.

Surprise maximization reveals the community structure of complex networks

Rodrigo Aldecoa and Ignacio Marín
Instituto de Biomedicina de Valencia. Consejo Superior de Investigaciones Científicas (IBV-CSIC) Calle Jaime Roig 11. Valencia, Spain

Scientific Reports 3, 1060 (2013)

How to determine the community structure of complex networks is an open question. It is critical to establish the best strategies for community detection in networks of unknown structure. Here, using standard synthetic benchmarks, we show that none of the algorithms hitherto developed for community structure characterization perform optimally in all networks. Significantly, evaluating the results according to their modularity, the most popular measure of the quality of a partition into communities, systematically provides mistaken solutions. However, a novel quality function, called Surprise, can be used to elucidate which is the optimal division into communities. Consequently, we show that the best strategy to find the community structure of these complex networks involves choosing among the solutions provided by all the different algorithms the one with the highest Surprise value. We conclude that Surprise maximization precisely reveals the community structure of complex networks.

The analysis of networks has profound implications in very different fields, from sociology to biology [1–5]. One of the most interesting features of a network is its community structure [6, 7]. Communities are groups of nodes that are more strongly or frequently connected among themselves than with the other nodes of the network. The best way to establish the communities present in a network is an open problem. Two related questions are still unsolved. First, which is the best algorithm to characterize networks of known community structure. Second, how to evaluate algorithm performance when the community structure is unknown. The first question requires testing the algorithms in benchmarks composed of complex networks where the community structure is established *a priori*. In these benchmarks, it has been found that algorithm performance depends on how different is the density of intracommunity links from the average density of links in the network. In addition, it has been determined that most algorithms perform well when the networks are small and the communities have similar sizes, but many perform quite poorly in benchmarks composed of large networks with many communities of heterogeneous sizes [8–18]. Thus, benchmarks with the latter features have become crucial to rank algorithm performances. Among them, the Lancichinetti-Fortunato-Radicchi (LFR) benchmarks [11–18] and the Relaxed Caveman (RC) benchmarks [14, 19, 20] have shown to be particularly useful. Both benchmarks pose a stern test for algorithms that deal poorly with the presence of many communities, of small communities or of a mixture of communities of different sizes (see e. g. refs. [11, 13, 14]).

The second question, how to determine the best performance when the community structure is unknown, involves devising an independent measure of the quality of a partition into communities that can be reliably applied to any type of network. The first and still today most popular such measure is called modularity [21] often abbreviated as Q). Modularity compares the number of links within each community with the expected number of links in a random graph of the same size and same distribution of node degrees and then adds the differences between expected and observed values for all the communities. It was proposed that the optimal partition of a network could be found by maximizing Q [21]. However, it was later determined that modularity-based evaluations are often incorrect when small communities are present in the network, i. e. Q has a resolution limit [22]. Several other works have found additional, subtle problems caused by using modularity maximization to determine network community structure [17, 23–26]. All these results suggest that using Q provides incorrect answers in many cases.

We recently suggested an alternative global measure of performance, which we called Surprise [14]. Surprise assumes, as a null model, that links between nodes emerge randomly. It then evaluates the departure of the observed partition from the expected distribution of nodes and links into communities given that null model. To do so, it uses the following cumulative hypergeometric distribution:

$$S = -\log \sum_{j=p}^{min(M,n)} \frac{\binom{M}{j}\binom{F-M}{n-j}}{\binom{F}{n}} \tag{2.8}$$

Where F is the maximum possible number of links in a network ($(k^2 - k)/2$, being k the number of units), n is the observed number of links, M is the maximum possible number of intracommunity links for a given partition, and p is the total number of intracommunity links observed in that partition [14]. Thus, S measures how unlikely (or "surprising", hence the name of the parameter) is the distribution of links and nodes in the communities defined in the network. In previous studies, we showed that Surprise improved on modularity in standard benchmarks and that choosing algorithms with high S values leads to accurate community structure characterization [14, 18]. Although these results were encouraging, whether S maximization could be used to obtain optimal partitions was not rigorously tested. This was due to the fact that Surprise values were estimated from the partitions provided by just a few algorithms. Given that other algorithms could provide even higher S values, it was unclear how optimal these results were.

Here, we test the best strategies currently available to characterize the structure of complex networks and we compare them with the results provided by Surprise maximization in both LFR and RC benchmarks. We first show that none among a large number of state-of-the-art algorithms work consistently well in all these complex benchmarks. Particularly, all modularity-based heuristics behave poorly. Also, we demonstrate that evaluating the performance of an algorithm using modularity is incorrect. We then show that a simple meta-algorithm, which consists in choosing in each network the algorithm that maximizes Surprise, very efficiently determines the community structure of all the networks tested. This method clearly performs better than any of the algorithms devised so far. We conclude that Surprise maximization is the strategy of choice for community characterization in complex networks.

Results

In order to determine the performance of different algorithms for community structure characterization, we explored two standard benchmarks, an LFR benchmark with 5000 units and an RC benchmark with 512 units (see Methods). Variation of Information (VI) was used to determine the degree of congruence between the partitions into communities suggested by 18 different algorithms and the real community structure present in the networks. A perfect congruence corresponds to a value VI = 0. Figures 1a and 1d display the general results obtained in the two benchmarks. A sharp VI increase was found when the community structure was weakened by highly increasing the number of intercommunity links, as occurs

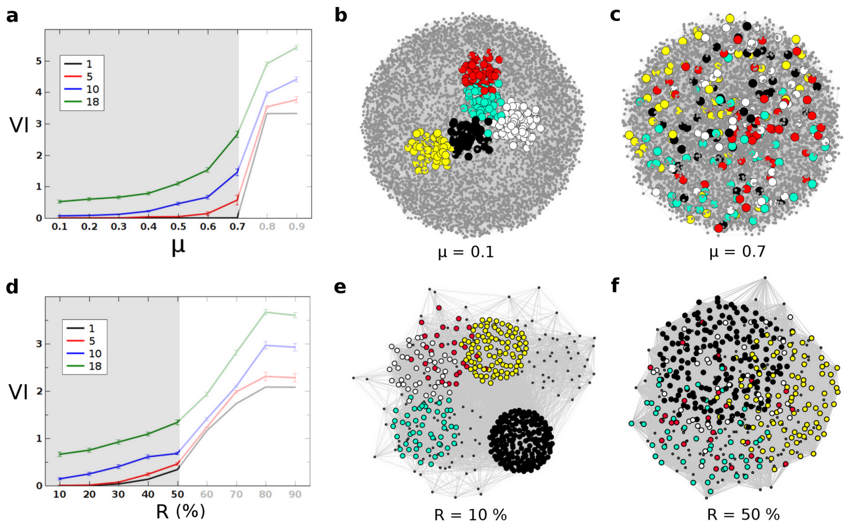


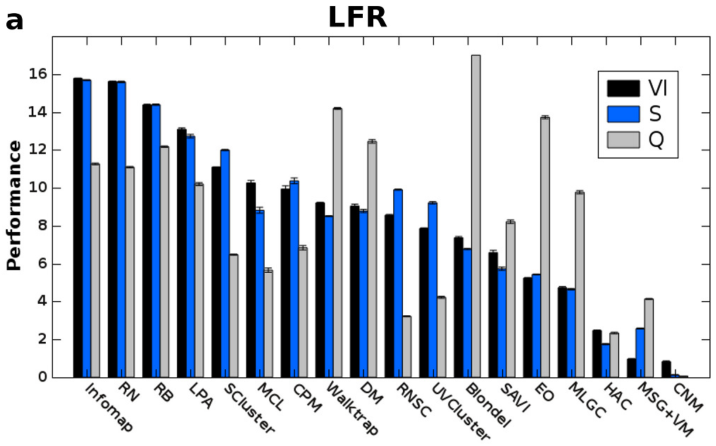
Figure 1: Global performance of the algorithms. a) Behavior of the algorithms in the LFR benchmark. To obtain this figure, the algorithms were first ordered according to the VI results obtained for each μ condition. Then, we plotted the results for the algorithm with the best VI value (black line, indicated with "1"), the average of the top five algorithms (red line), average of the top ten ones (blue line) or average for all the 18 algorithms (green line). The grey region corresponds to the values of μ (0.1 - 0.7) chosen to perform the main comparative analyses (see text). Beyond that region, even the best algorithms obtain VI values considerably higher than zero, meaning that the original structure of the network has been significantly modified by the increase in intercommunity links. b) An example showing the five largest communities in a LFR network (5000 units) when $\mu = 0{,}1$. Nodes are distributed into two dimensions with a spring-embedded algorithm [27] and drawn using Cytoscape [28]. Communities are well-isolated groups. c) The five largest communities when $\mu = 0{,}7$. They are barely distinguishable in this representation because the mixing of links was quite extreme. However, several algorithms were still often able to detect these fuzzy communities. d) - f): The same results for the RC benchmark (512 units). Panel e depicts the five largest communities when R = 10 % and Panel f to the same communities when R = 50 %. Again, notice in panel d) the sharp increase in VI values when R $>$ 50 %. An extreme degree of superimposition among communities is observed already when R = 50 % (f). In the LFR benchmark, the rapid increase in VI values when the intercommunity links goes from μ = 0.7 to 0.8 (Panel a) is explained by all communities being of similar sizes. Therefore, they are destroyed at about the same time. On the contrary, the more progressive increase in VI when R grows, which we observed in Panel d, is due to the heterogeneous sizes of the communities present in that benchmark, which break down at different times.

when the mixing parameter μ of the LFR benchmark has values above 0.7 or the rewiring parameter R of the RC benchmarks is higher than 50 % (see also Methods for the precise definitions of μ These results mean that, above $\mu = 0.7$ or R = 50 %, the community structure originally present in the networks was substantially altered. In such cases, we could not determine whether the partitions suggested by the algorithms were correct or not: there would not be a known structure with which to compare. Thus, we decided to restrict our subsequent analyses to the LFR networks with $0{,}1 \leq \mu \leq 0{,}7$ (100 realizations per μ value, giving a total of 700 networks) and the RC networks with $10\,\% \leq R \leq 50\,\%$ (again, 100 realizations per R value, for a total of 500 different networks). These conditions generate some community structures that are very difficult to detect (Figure 1).

Figure 2 summarizes the individual performance of the algorithms according to three global measures of partition quality. The first one is VI, the gold standard for algorithm performance in these benchmarks. The other two, already mentioned above, are Surprise (S) and modularity (Q). The performance values measured according to the VI scores shown in Figure 2 indicate two very important facts. On one hand, none of the algorithms was the best in all LFR or in all RC networks. On the other hand, the best algorithms in LFR networks often performed poorly in RC networks, and vice versa (see e. g. the results of RB, LPA or RNSC in Figure 2). This can be rigorously shown by ordering within each benchmark the algorithms according to their performance, assigning a rank, from best to worst, and comparing the ranks in both benchmarks. We found that Kendall's non-parametric correlation coefficient for these ranks was very weak, just $\tau = 0.31$ (p = 0.04, one-tailed comparison). We conclude that using single algorithms for community characterization is inadvisable, given that their performance is strongly dependent on the particular structure of the network.

If we focus now on the Surprise (S) and modularity (Q) results shown in Figure 2, another two striking facts become apparent. First, there was a very strong correlation between the performance of the algorithms according to VI and according to S. Kendall's correlation coefficient for the ranks of the performances of the algorithms ordered according to VI and to S values is $\tau = 0.91$ in the LFR benchmarks (p = 4.9 x 10^{-11}, one-tailed comparison) and $\tau = 0.83$ in the RC benchmark (p = 1.4 x 10^{-8}, one-tailed test). These results demonstrate that S is an excellent measure of the global quality of a division into communities, confirming and extending the conclusions of one of our previous works [14]. Second, the performance of the algorithms evaluated using Q only weakly correlated with their performance according to VI in the LFR benchmarks (Kendall's $\tau_{LFR} = 0.29$, $p = 0.048$, one-tailed test) and these two measures did not significantly correlate in the RC benchmarks ($\tau_{RC} = 0.27$, $p = 0.66$, again one-tailed test). These results indicate that evaluating the quality of a partition according to its modularity is inappropriate. It was therefore logical to find out that both the algorithms devised to maximize Q (Blondel, EO, MLGC, MSG+VM and CNM [29–33]) and the algorithms that use Q to evaluate the quality of their partitions (Walktrap,

a



b

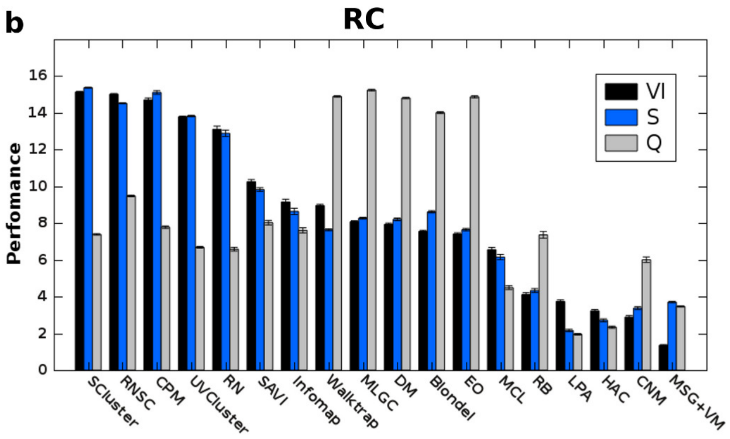


Figure 2: Performance of the algorithms according to Variation of Information (VI), Surprise (S) and Modularity (Q) in LFR and RC benchmarks. Average performance and standard errors of the mean are shown. Performance values were obtained by the following method: 1) the VI, S or Q values of the partitions provided by the 18 algorithms in each of the networks (i. e. 700 values for LFR benchmarks, 500 values in RC benchmarks) were established; 2) For each network, the algorithms were assigned a rank according to their performance (1 = optimal, 18 = worse); identical ranks were given to tied algorithms (i. e. the ranks that would correspond to each of them were summed up and then divided by the number of tied algorithms); and, 3) Performance was calculated as 18 – average rank, meaning that 17 is the maximum possible value that would obtain an algorithm that outperforms the rest in all networks, and 0 equals to being the worst in all networks.

DM [34,35] were poor performers (Figure 2).

If indeed maximization of Surprise is an optimal strategy for community characterization, as its strong correlation with VI suggests, then it should be possible to improve on the results of any single algorithm by simply picking up among many algorithms the one that generates the highest S value (S_{max}) in each particular network. Also, this S-maximization strategy should provide VI values very close to zero in our benchmarks. These two expectations are fulfilled, as shown in Figure 3. The top panel (Figure 3a) demonstrates that choosing in each particular case the algorithm with the highest S value is better than selecting any of the state-of-the-art algorithms tested. It is remarkable that the S_{max} values in Figure 3a derived from the combined results of as many as 7 algorithms (CPM, Infomap, RB, RN, RNSC, SCluster and UVCluster [16,20,36–40]). In addition, Figure 3b indicates that the sum of the average VI values obtained using S_{max} in the 1200 networks analyzed (with μ = 0.1 - 0.7 and R = 10 - 50 %) were just slightly above zero, i. e. almost optimal. The average values were 0.002 $\pm$ 0.000 in the LFR benchmarks and 0.100 $\pm$ 0.007 in the RC benchmarks. We may ask why these VI values are not exactly zero, given that VI = 0 would be expected for a perfect global measure. We detected two reasons for this minor discrepancy. The first reason was that, in some cases (mainly in the RC benchmark with R = 50 %), the available algorithms failed to obtain the highest possible S values. We found that the S values expected assuming that the original community structure of the network was intact (S_{orig}) were often higher than S_{max} (Table 1). This obviously means that these algorithms did not found the community structure that maximizes S. That structure could still be the original one – which indeed has the highest S value observed so far in our analyses – or some alternate structure, but clearly not any of those found by the algorithms, which had lower S values. The second reason observed was the presence of minor changes in community structure that occurred in some networks when intercommunity links increased. Thus, the exact original structure of the network was not present anymore. This was deduced from the fact that S_{max} values were sometimes slightly higher than S_{orig} both in the LFR benchmarks with μ = 0.6 - 0.7 and the RC benchmarks with R = 10 - 40 % (Table 1). These results suggested that the algorithms obtained optimal partitions, but they were a bit different from the original ones. To establish that fact, we examined the 23 cases where $S_{max} > S_{orig}$ in the RC benchmarks with R = 10 %. We found that the partitions with S_{max} values generally differed from the original structures in one of the smallest communities having lost single units (Supplementary Table S1; see example in Figure 4).

Significantly, in those 23 networks we always found just one partition with $S_{max} > S_{orig}$ and several algorithms often recovered exactly that same partition (Supplementary Table S1). All these results indicate that real, small changes in community structure occurred in those networks, suggesting that the partitions with $S_{max} > S_{orig}$ values were indeed optimal. From the data in Table 1, we also obtain an indirect validation of our decision of using the LFR benchmarks with $\mu \leq 0.7$ and the RC benchmarks with R $\leq$ 50 % to evaluate algorithm

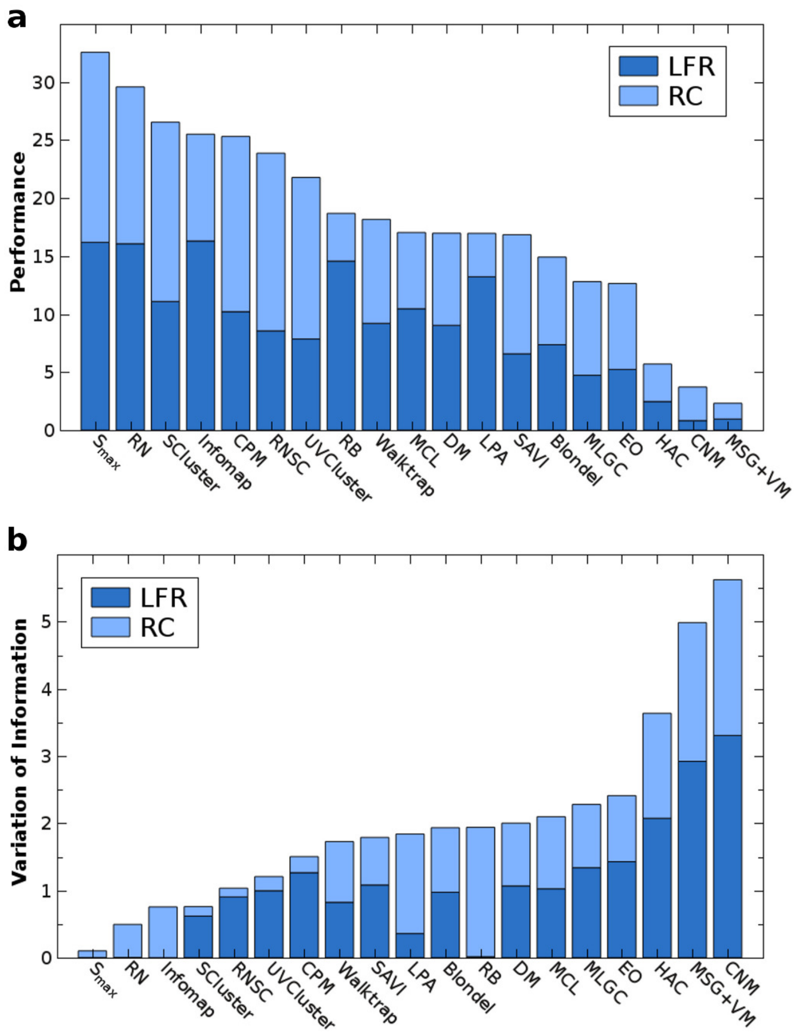
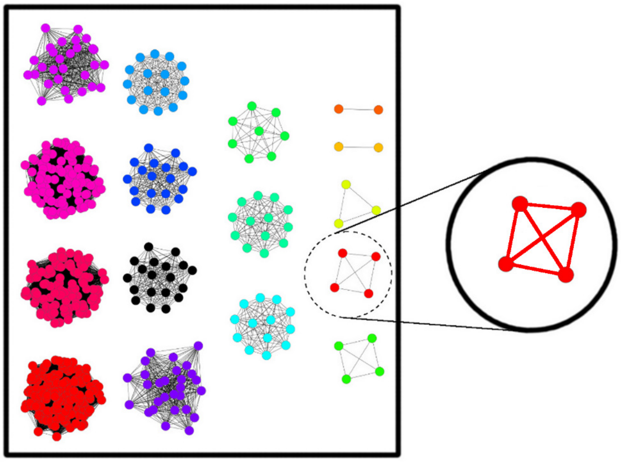


Figure 3: A simple meta-algorithm based on Surprise maximization improves over all known community detection algorithms. a) Performances (calculated as in Figure 2) for all the algorithms are compared in both the LFR and the RC benchmarks with the performance of a strategy that consists in picking up the algorithm that provides the highest S value (S_{max}). b) For the S_{max} strategy, the average VI values for the 1200 networks analyzed are very close to zero, i. e. an almost optimal performance.

a



b

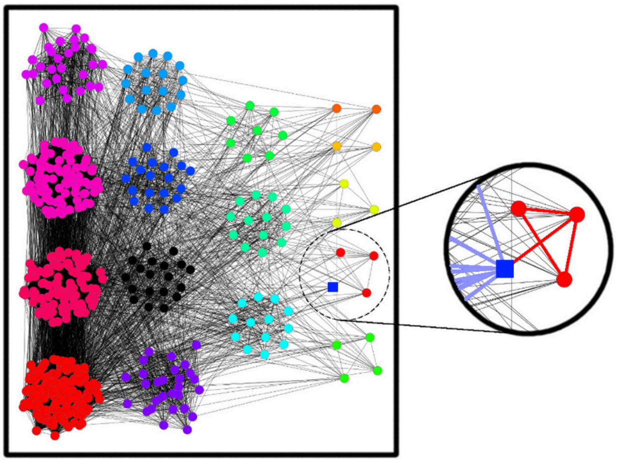


Figure 4: When VI and Surprise maximum values do not coincide, the difference is often due to minimal changes in the community structure of the network. This is an example from the RC benchmark where $S_{max} > S_{orig}$ (see text). a) original structure. b) after R = 10 % has been applied. S_{max} is obtained when a single unit (square) is classified as being isolated from its original 4-nodes community (highlighted). As shown in panel b), the critical unit has become almost fully separated from the rest of the nodes in its original community, only one link remains, while it has been connected to many nodes in other communities.

performance. As shown in that Table, up to those limits, the S_{max} and S_{orig} values are not significantly different, while, beyond those limits, very significant differences are found. This means that the original structures, or structures almost identical to them, were indeed present in the networks examined to generate the results summarized in Figures 2 and 3, which precisely was the only condition required for a reliable measure of algorithm performance.

The important results described in Figures 2 and 3 indicate that S maximization should allow determining with a very high precision the community structure of any network. We have explored whether this may be the case even when the community structure is very poorly defined by analyzing the results of our 600 additional networks, corresponding to the LFR benchmark with mixing parameter $\mu = 0.8$ and $\mu = 0.9$ (i. e. 200 networks) and the RC benchmark with R = 60 % to R = 90 % (400 networks). As indicated above, in these networks, the VI-based optimality criterion (i. e. VI = 0 means finding the original community structure) cannot be confidently used (Figure 1; Table 1). However, alternative, unknown structures may be present that the algorithms should be able to detect. If this is the case, a reasonable prediction is that the algorithms that are providing the maximum S values in the conditions that are closest to those extreme ones (i. e. when $\mu = 0.7$ in the LFR benchmarks and R = 50 % in the RC benchmarks) should also provide the best S values in the most extreme networks. Figure 5

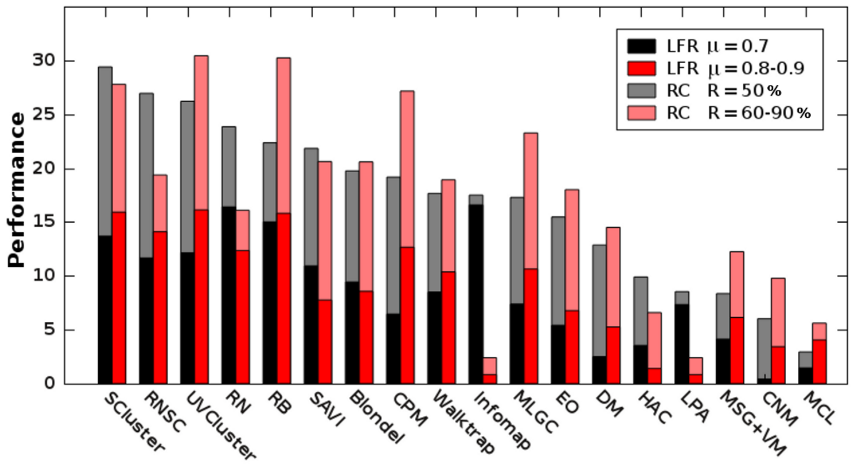


Figure 5: The performance of the algorithms in the limit cases ($\mu = 0.7$, R = 50 %) and beyond those limits ($\mu = 0.8$ - 0.9, R = 60 - 90 %) are correlated. A statistically significant correlation was found, despite the fact that some algorithms, such as Infomap or LPA, totally collapsed. These algorithms established partitions consisting in a single community, which led to VI = 0 when compared with the original distribution.

shows that there is indeed a good correlation between the results obtained in the limit cases and in the most extreme cases. Kendall's non-parametric correlation coefficients for the ranks of the algorithms in the limit networks and in the most extreme networks are significant in both the LFR and RC benchmarks ($\tau_{LFR} = 0.42$; $p = 0.007$ and $\tau_{RC} = 0.49$, $p = 0.020$, one-tailed tests). This occurs despite some algorithms, as Infomap or LPA [36,41], totally failing in these quasi-random networks (Figure 5). UVCluster, RB, CPM and SCluster [16,20,37,40] emerge as the best algorithms to characterize the structure in networks with poorly defined communities, in good agreement with previous results [14,16].

We decided to perform some final tests to determine whether the limitations that affect Q when communities are very small may also affect S. For this purpose, we used two extreme networks of known structure suggested before [17, 22, 42]. The first one includes just three communities, one of them very large (400 nodes with average degree = 100) and the other two much smaller (cliques of 13 nodes). These three communities are connected by single links (Figure 6a). We found in this network that the maximum value of Q (EO algorithm, Q = 0.0836) did not correspond to a partition into the three natural communities, but, as already noted by other authors in similar cases [17,22] led instead to a mistaken solution, in this case with five communities. On the other hand, the three communities were correctly found by multiple algorithms (CPM, Infomap, LPA, RNSC, SCluster and UVCluster), and this partition indeed corresponded to the maximum value of S (1230.73). The second extreme type of network was precisely the ring of cliques in which the resolution limit of Q was first described [22], which is schematized in Figure 6b. Here, a variable number of cliques, each one composed of five units, were connected to each other by single links to form a ring. We were interested in determining whether, even if we increase the number of cliques, a solution in which all cliques are separately detected always has a better S value than one in which pairs of cliques are put together. We tested networks of sizes up to 1 million nodes, finding that the best partition was always the one in which the cliques are considered independent communities. On the contrary, when Q is used, the cliques are considered independent units only if the network size is smaller than 150 units.

Discussion

Our results lead to two main conclusions. The first one is that none of the algorithms currently available generates optimal solutions in all networks (Figures 2, 3). In fact, there is just a weak correlation of the algorithm performances in the two standard benchmarks used in this study. More precisely, we can say that there are some algorithms that clearly fail in both benchmarks and the rest tend to perform much better in one of the benchmarks than in the other (see Figure 2). Most of the best overall performers were already found to be outstanding in other studies [12–14, 16, 18, 38]. The exception is RNSC [39], which had not been tested in depth before. Among the ones that always perform poorly are

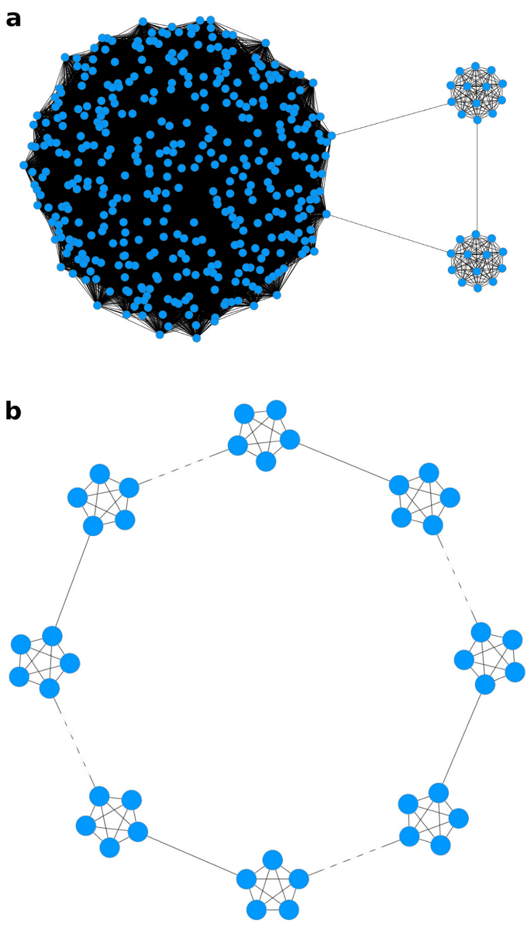


Figure 6: Two extreme networks designed to test the behavior of Surprise when small communities are present. a) A network with three communities (sizes 400, 13, 13). The nodes of the largest community have an average degree of 100, while the nodes in the two smallest communities form cliques. The three communities are interconnected by single links, as shown. b) Cliques, each one with five nodes, which are connected also by single links in a way that can be depicted as a ring. The figure shows an example with eight cliques, but that number was progressively increased to determine whether the partition with highest S still corresponded to the one in which each cliques was an independent community.

all the algorithms that use modularity as either a global parameter to maximize or as a way to evaluate partitions. This fact, together with the demonstration that Q does not correlate with VI in networks of known structure (Figure 2), and also the good performance of S, including its ability to cope with extreme networks in which Q traditionally fails due to its resolution limit (Figure 6), should definitely deter researchers from using modularity. A strong corollary is that a reevaluation of the hundreds of already published papers in fields as varied as sociology, ecology, molecular biology or medicine that are based on modularity analyses seems advisable.

The second, and most important, conclusion is that the community structure of a network can be determined by maximizing S, for example by simply taking the results of as many algorithms as possible and choosing the one that provides partitions with the highest Surprise value. In a previous paper, we showed that Surprise can be used to efficiently evaluate the quality of a partition, behaving much better than modularity [14], but the precise performance of the S-maximization strategy was not determined. Here, we extend those results, to show that using S maximization leads to an almost perfect performance. We were very close to solve the correct community structure of all the networks of these two benchmarks, as is strikingly demonstrated by the S_{max} results shown in Figure 3b. It is significant that they were obtained by combining results of the 7 algorithms with the best average performances, as detailed in Figure 3a: RN, SCluster, Infomap, CPM, RNSC, UVCluster and RB. Another important result is summarized in Figure 5, which indicates that Surprise can also be used in cases in which the community structure is so blurred as to become almost random. Given these results, we conclude that Surprise is the parameter of choice to characterize the community structure of complex networks. Future works should use Surprise maximization, instead of modularity maximization or other methods, to establish that structure.

It is significant that only two algorithms (SCluster and UVCluster) use the maximization of Surprise to choose among partitions generated by consensus hierarchical clustering [20, 40]. This may explain their good average results (Figures 2, 3, 5). However, no available algorithm performs searches to directly determine the maximal Surprise values. That type of algorithms could overcome the limitations detected in all the currently available ones, potentially allowing the characterization of optimal partitions even in the most difficult networks.

We may ask why Surprise is able to evaluate with such efficiency the quality of a given partition, while modularity cannot. In our opinion, the difference rests on the fact that modularity is based on an inappropriate definition of community. Newman and Girvan [21] verbally defined a community as a region of a network in which the density of links is higher than expected by chance. However, the precise mathematical model used to deduce the modularity formula implies a definition of community that does not take into account the number of nodes required to achieve such a high density [21]. By not evaluating the number of nodes, modularity falls prey of a resolution limit: small communities cannot be detected [17, 22]. On the other hand, Surprise analyses often choose as best a solution where some communities are just isolated units (see examples in Figure 4 and Ref. 14). This happens because the Surprise formula precisely evaluates not only the number of links, but also the number of nodes within each community. For instance, incorporating a single poorly connected unit into a community is often forbidden by the fact that such incorporation sharply increases the number of potential intracommunity links (all those that might connect the units already present in the community with the new unit) while barely increasing the number of real intracommunity links. This leads to an S value much smaller than if the unit is kept separated. It is also significant that a general problem of modularity maximization and other related algorithms - as those based on Potts models with multiresolution parameters - is that they cannot find a perfect equilibrium between merging and splitting communities [17,25,26]. In these methods, each community is evaluated independently, one at a time. The global value to be maximized is the sum of the qualities of the individual communities. However, in complex networks with communities of very different sizes, it may be often impossible to find a single rule (even using a tunable parameter, as in these multiresolution methods) to split some communities while keeping intact the rest17. Surprise analyses are not affected by this problem, because communities are not defined independently, one by one, but emerge as regions of nodes statistically enriched in links, according to the general features (i. e. the total number of nodes and links) of the whole network.

Methods

We searched the literature to select the best community detection algorithms available to analyze networks with unweighted, undirected links. Our final results are based on 18 of them (summarized in Table 2). Algorithms known to behave poorly in similar benchmarks or specifically designed to characterize communities with overlapping nodes were discarded. Some other algorithms that seemed interesting but we were unable to test for diverse reasons (e. g. they were not provided by the authors, did not complete the benchmarks, etc.) are detailed in Supplementary Table S2. We performed extensive tests with these selected algorithms, using their default parameters, in two very different benchmarks. They were chosen both difficult and very dissimilar, with the idea that the results could be general enough as to be extrapolated to networks of unknown structure. The first was a standard LFR benchmark already used in other studies where algorithms were compared [12–14, 17]. It is composed of networks with 5000 nodes, structured in small communities with 10-50 nodes. The distribution of node degrees and community sizes were generated according to power laws with exponents -2 and -1, respectively. The sizes of the communities in the networks of this benchmark have average Pielou's indexes [43] with a value of 0.98. This index is equal to 1 when all communities are of the same size. The chief difficulty of this benchmark thus lies on the presence of many small communities. The second benchmark was one of the Relaxed Caveman (RC) type, very similar to the ones used in our previous works [14, 18]. The networks in this RC benchmark have 512 units and 16 communities, with sizes defined according to a broken-stick model to obtain an average Pielou's index = 0.75. This makes this benchmark very difficult, given that it consists of networks with communities of very different sizes, some of them very small (see e. g. Figure 4). It was not convenient to our purposes to use larger RC benchmarks given that the total number of links in these networks quickly grows when the number of nodes is increased and many algorithms become too slow.

These two benchmarks are "open", meaning that they have a tunable parameter that, when increased, makes the network community structure to become less and less obvious until it shifts towards a totally unknown structure, potentially very different from the original one and close to random [11, 13, 14, 18]. This parameter increases intracommunity links and lowers the number of intercommunity links. In the case of the LFR benchmarks, the "mixing parameter", μ, indicates the fraction of links connecting each node of a community with nodes outside of the community [11]. For the RC benchmarks, we defined Rewiring (R) as the percentage of links that is randomly shuffled among units. Thus, R = 10 % means that 10 per cent of the links were first randomly removed and then added again, to link randomly chosen nodes.

Variation of information (VI) [44] was used to measure the agreement between the original community structure present in the network and the structure deduced by each algorithm. The advantages of using VI have been discussed in our previous works [14, 18]. A perfect agreement with a known structure will provide a value of VI = 0. In addition, two global quality functions, Newman and Girvan's modularity (Q) [21] and Surprise (S) [14], (see Formula [1]), were also used to evaluate the results. The values of S and Q for the partitions proposed by each algorithm were calculated and then all the values were used to determine the correlations of S and Q with VI and to establish these maximum values of S.

LFR benchmark			
μ	S_{orig}	S_{max}	p
0.1	99065.69 ± 111.50	99065.69 ± 111.50	ns
0.2	82631.18 ± 93.92	82631.18 ± 93.92	ns
0.3	67847.35 ± 90.78	67847.35 ± 90.78	ns
0.4	54354.47 ± 76.71	54354.47 ± 76.71	ns
0.5	41991.16 ± 48.70	41991.16 ± 48.70	ns
0.6	30807.18 ± 40.09	30807.38 ± 40.09	ns
0.7	20563.37 ± 26.92	20570.70 ± 26.78	ns
0.8	*11598.83 ± 17.91*	*10168.11 ± 28.15*	$<$ *0.0001*
0.9	*4204.50 ± 7.62*	*8368.94 ± 4.21*	$<$ *0.0001*

RC benchmark			
R	S_{orig}	S_{max}	p
10	19012.72 ± 67.33	19012.94 ± 67.32	ns
20	13505.84 ± 34.14	13506.72 ± 34.11	ns
30	9298.98 ± 11.88	9301.12 ± 11.88	ns
40	6013.69 ± 3.92	6017.58 ± 4.09	ns
50	3487.65 ± 11.54	3479.92 ± 12.99	ns
60	*1647.42 ± 13.82*	*1540.76 ± 16.79*	$<$ *0.0001*
70	*475.35 ± 10.42*	*899.96 ± 7.98*	$<$ *0.0001*
80	*11.84 ± 1.52*	*963.73 ± 9.42*	$<$ *0.0001*
90	*0.00 ± 0.00*	*1003.21 ± 9.95*	$<$ *0.0001*

Table 1: Average S_{orig} and S_{max} values in the LFR and RC benchmarks. Statistical significance (p) was estimated using a two-tailed Student t test. ns: non-significant differences. In italics, the benchmarks containing quasi-random networks, discarded for the main analyses (summarized in Figures 2 and 3), but included in the analyses shown in Figure 5

Name of the Algorithm	**Strategy used by the algorithm**	**References**
Blondel	Multilevel modularity maximization	[29]
CNM	Greedy modularity maximization	[33]
CPM	Multiresolution Potts model	[16]
DM	Spectral analysis + modularity maximization	[35]
EO	Modularity maximization	[30]
HAC	Maximum Likelihood	[45]
Infomap	Information compression	[36]
LPA	Label propagation	[41]
MCL	Simulated flow	[46]
MLGC	Multilevel modularity maximization	[31]
MSG+VM	Greedy modularity maximization + refinement	[32]
RB	Multiresolution Potts model	[37]
RN	Multiresolution Potts model	[38]
RNSC	Neighborhood tabu search	[39]
SAVI	Optimal prediction for random walks	[47]
SCluster	Hierarchical Clustering + Surprise maximization	[20]
UVCluster	Hierarchical Clustering + Surprise maximization	[20, 40]
Walktrap	Random walks + modularity maximization	[34]

Table 2: Details of the algorithms used in this study. A summary of the strategies implemented by the algorithms and the corresponding references are indicated.

References

[1] S. Wasserman and K. Faust. *Social network analysis: Methods and applications*, volume 8. Cambridge university press, 1994.

[2] S.H. Strogatz. Exploring complex networks. *Nature*, 410:268–276, 2001.

[3] A.L. Barabási and Z.N. Oltvai. Network biology: understanding the cell's functional organization. *Nature Reviews Genetics*, 5:101–113, 2004.

[4] L.F. Costa, F.A. Rodrigues, G. Travieso, and P.R.V. Boas. Characterization of complex networks: A survey of measurements. *Advances in Physics*, 56:167–242, 2007.

[5] M. Newman. *Networks: an introduction.* Oxford University Press, Inc., 2010.

[6] S. Fortunato. Community detection in graphs. *Physics Reports*, 486:75–174, 2010.

[7] V. Labatut and J.M. Balasque. Detection and interpretation of communities in complex networks: Practical methods and application. *Computational Social Networks*, pages 81–113, 2012.

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

[9] L. Danon, A. Díaz-Guilera, J. Duch, and A. Arenas. Comparing community structure identification. *Journal of Statistical Mechanics: Theory and Experiment*, page P09008, 2005.

[10] L. Danon, A. Díaz-Guilera, and A. Arenas. The effect of size heterogeneity on community identification in complex networks. *Journal of Statistical Mechanics: Theory and Experiment*, page P11010, 2006.

[11] A. Lancichinetti, S. Fortunato, and F. Radicchi. Benchmark graphs for testing community detection algorithms. *Physical Review E*, 78:046110, 2008.

[12] A. Lancichinetti and S. Fortunato. Community detection algorithms: a comparative analysis. *Physical Review E*, 80:056117, 2009.

[13] G. Orman, V. Labatut, and H. Cherifi. Qualitative comparison of community detection algorithms. *arXiv preprint arXiv:1207.3603*, 2012.

[14] R. Aldecoa and I. Marín. Deciphering network community structure by surprise. *PloS one*, 6:e24195, 2011.

[15] P. Ronhovde and Z. Nussinov. Multiresolution community detection for megascale networks by information-based replica correlations. *Physical Review E*, 80:016109, 2009.

[16] VA Traag, P. Van Dooren, and Y. Nesterov. Narrow scope for resolution-limit-free community detection. *Physical Review E*, 84:016114, 2011.

[17] A. Lancichinetti and S. Fortunato. Limits of modularity maximization in community detection. *Physical Review E*, 84:066122, 2011.

[18] R. Aldecoa and I. Marín. Closed benchmarks for network community structure characterization. *Physical Review E*, 85:026109, 2012.

[19] D.J. Watts. *Small worlds: the dynamics of networks between order and randomness*. Princeton university press, 2003.

[20] R. Aldecoa and I. Marín. Jerarca: Efficient analysis of complex networks using hierarchical clustering. *PloS one*, 5:e11585, 2010.

[21] M.E.J. Newman and M. Girvan. Finding and evaluating community structure in networks. *Physical review E*, 69:026113, 2004.

[22] S. Fortunato and M. Barthelemy. Resolution limit in community detection. *Proceedings of the National Academy of Sciences*, 104:36–41, 2007.

[23] B.H. Good, Y.A. de Montjoye, and A. Clauset. Performance of modularity maximization in practical contexts. *Physical Review E*, 81:046106, 2010.

[24] J.P. Bagrow. Communities and bottlenecks: Trees and treelike networks have high modularity. *Physical Review E*, 85:066118, 2012.

[25] J. Xiang and K. Hu. Limitation of multi-resolution methods in community detection. *Physica A: Statistical Mechanics and its Applications*, 2012.

[26] J. Xiang, XG Hu, XY Zhang, JF Fan, XL Zeng, GY Fu, K. Deng, and K. Hu. Multi-resolution modularity methods and their limitations in community detection. *The European Physical Journal B-Condensed Matter and Complex Systems*, 85:1–10, 2012.

[27] T. Kamada and S. Kawai. An algorithm for drawing general undirected graphs. *Information processing letters*, 31:7–15, 1989.

[28] M.E. Smoot, K. Ono, J. Ruscheinski, P.L. Wang, and T. Ideker. Cytoscape 2.8: new features for data integration and network visualization. *Bioinformatics*, 27:431–432, 2011.

[29] V.D. Blondel, J.L. Guillaume, R. Lambiotte, and E. Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, P10008, 2008.

[30] J. Duch and A. Arenas. Community detection in complex networks using extremal optimization. *Physical review E*, 72:027104, 2005.

[31] A. Noack and R. Rotta. Multi-level algorithms for modularity clustering. *Lecture Notes in Computer Science*, pages 257–268, 2009.

[32] P. Schuetz and A. Caflisch. Efficient modularity optimization by multistep greedy algorithm and vertex mover refinement. *Physical Review E*, 77:046112, 2008.

[33] A. Clauset, M.E.J. Newman, and C. Moore. Finding community structure in very large networks. *Physical review E*, 70:066111, 2004.

[34] P. Pons and M. Latapy. Computing communities in large networks using random walks. *Computer and Information Sciences-ISCIS 2005*, 3733:284–293, 2005.

[35] L. Donetti and M.A. Munoz. Detecting network communities: a new systematic and efficient algorithm. *Journal of Statistical Mechanics: Theory and Experiment*, P10012, 2004.

[36] M. Rosvall and C.T. Bergstrom. Maps of random walks on complex networks reveal community structure. *Proceedings of the National Academy of Sciences*, 105:1118–1123, 2008.

[37] J. Reichardt and S. Bornholdt. Statistical mechanics of community detection. *Physical Review E*, 74:016110, 2006.

[38] P. Ronhovde and Z. Nussinov. Local resolution-limit-free potts model for community detection. *Physical Review E*, 81:046114, 2010.

[39] AD King, N. Pržulj, and I. Jurisica. Protein complex prediction via cost-based clustering. *Bioinformatics*, 20:3013–3020, 2004.

[40] V. Arnau, S. Mars, and I. Marín. Iterative cluster analysis of protein interaction data. *Bioinformatics*, 21:364–378, 2005.

[41] U.N. Raghavan, R. Albert, and S. Kumara. Near linear time algorithm to detect community structures in large-scale networks. *Physical Review E*, 76:036106, 2007.

[42] C. Granell, S. Gomez, and A. Arenas. Hierarchical multiresolution method to overcome the resolution limit in complex networks. *International Journal of Bifurcation and Chaos*, 22:1250171, 2012.

[43] ECJ Pielou. The measurement of diversity in different types of biological collections. *Journal of theoretical biology*, 13:131–144, 1966.

[44] M. Meilă. Comparing clusterings—an information based distance. *Journal of Multivariate Analysis*, 98:873–895, 2007.

[45] Y. Park and J.S. Bader. Resolving the structure of interactomes with hierarchical agglomerative clustering. *BMC bioinformatics*, 12:S44, 2011.

[46] A.J. Enright, S. Van Dongen, and C.A. Ouzounis. An efficient algorithm for large-scale detection of protein families. *Nucleic acids research*, 30:1575–1584, 2002.

[47] E. Weinan, T. Li, E. Vanden-Eijnden, et al. Optimal partition and effective dynamics of complex networks. *Proceedings of the National Academy of Sciences*, 105:7907–7912, 2008.

Supplementary Information

N.algs	Algorithms	S_{max}	S_{orig}	Differences
4	CPM, RN, RNSC, SCluster	18616.55	18615.55	A community of 2 nodes is split into two
4	CPM, RN, RNSC, SCluster	18881.92	18879.94	A community of 3 nodes is split into two: 2 nodes + 1 node
4	CPM, RN, RNSC, SCluster	18442.72	18440.74	Two communities of 2 nodes are split into two
4	CPM, RN, RNSC, SCluster	19089.77	19088.78	A community of 2 nodes is split into two
4	CPM, RN, RNSC, SCluster	19187.13	19186.13	A community of 2 nodes is split into two
3	CPM, RN, SCluster	18312.46	18312.11	A community of 4 nodes is divided into two: 3 + 1 (displayed in Figure 4)
3	CPM, RN, SCluster	19897.8	19896.81	A community of 2 nodes is split into two
3	CPM, RN, SCluster	17980.46	17979.13	A community of 5 nodes is split into two: 4 + 1
3	CPM, RN, Scluster	17992.87	17991.54	A community of 5 nodes is split into two: 4 + 1
3	CPM, RN, SCluster	19579.76	19578.77	A community of 2 nodes is split into two
3	CPM, RN, SCluster	18008.52	18005.89	A community of 2 nodes is split into two and a community of 3 nodes is split into two: 2 + 1
3	CPM, RN, SCluster	18835.32	18834.33	A community of 2 nodes is split into two
2	CPM, SCluster	17803.45	17803.14	A community of 2 nodes is split into two and a community of 4 nodes is split into two: 3 + 1
2	CPM, SCluster	19928.95	19927.51	A community of 4 nodes is split into two: 2 + 2
2	CPM, SCluster	17749.38	17748.06	A community of 4 nodes is split into two: 2 + 2
2	RN, SCluster	18295.54	18295.19	A community of 4 nodes is split into two: 3 + 1
1	SCluster	18685.24	18684.48	A community of 5 nodes is split into two: 3 + 2
1	SCluster	19122.29	19121.88	A community of 4 nodes is split into two: 3 + 1
1	SCluster	18837.36	18836.58	A community of 7 nodes is split into two: 6 + 1
1	SCluster	18853.16	18852.39	A community of 5 nodes is split into two: 3 + 2
1	SCluster	19285.36	19284.95	A community of 4 nodes is split into two: 3 + 1
1	SCluster	18875.51	18875.13	A community of 4 nodes is split into two: 3 + 1
1	CPM	18141.17	18139.54	Two communities of 3 nodes are restructured in two communities of 4 and 2 nodes

Table S1: Cases where $S_{max} > S_{orig}$ in the RC benchmarks with R = 10 %. Small differences between S_{max} and S_{orig} are due to the rapid degradation of small communities. In most cases, several algorithms find the S_{max} partition instead of the original one, strongly supporting the idea that the community structure has actually changed.

Name	**Strategy used by the algorithm**	**Reference**	**Reasons for not including the algorithm**
AFG	Multiresolution Potts Model	Arenas, A., Fernandez, A. & Gomez, S. New Journal of Physics 10, 23 (2008).	Ambiguous choice of the best partition. Too slow for good modularity optimization heuristics in our benchmarks
EM	Maximum Likelihood	Ball, B., Karrer, B. & Newman, M.E.J. Phys. Rev. E 84, 036103 (2011)	Needs initialization. Not every nodes are assigned to a single cluster
HQcut	Multilevel modularity maximization	Ruan, J. & Zhang, W. Phys. Rev. E 77, 016104 (2007).	Unable to complete all the analyses
iMod	Modularity maximization	Xu, G., Bennett, L., Papageorgiou, L.G. & Tsoka, S. Algorithms Mol. Biol. 5, 36 (2010).	According to the authors, only they can run the algorithm, given its particular platform and software dependencies
Infomod	Information compression	Rosvall, M. & Bergstrom, C.T. Proc. Natl. Acad. Sci. USA 104, 7327 (2007).	Unable to complete all the analyses
QMC	Qualified Min-Cut	Zhang, X.-S., Li, Z., Wang, R.-S. & Wang, Y. J. Comb. Optim. 23, 425-442 (2010).	Authors did not answer our request
Random walks	Consensus hierarchical clustering	Steinhaeuser, K. & Chawla, N.V. Pattern Recogn Lett 31: 413–421 (2010)	The number of communities must be specified a priori

Table S2: Algorithms not included in our study.

Exploring the limits of community detection strategies in complex networks

Rodrigo Aldecoa and Ignacio Marín
Instituto de Biomedicina de Valencia. Consejo Superior de Investigaciones Científicas (IBV-CSIC) Calle Jaime Roig 11. Valencia, Spain

The characterization of network community structure has profound implications in several scientific areas. Therefore, testing the algorithms developed to establish the optimal division of a network into communities is a fundamental problem in the field. We performed here a highly detailed evaluation of community detection algorithms, which has two main novelties: 1) the use of complex closed benchmarks, which allow precise ways to assess whether the solutions provided by the algorithms are optimal or not; and, 2) A novel type of analysis, based on hierarchically clustering the solutions suggested by the different methods, in order to visualize their relationships. Surprise, a global parameter that evaluates the quality of a partition, confirms the power of these analyses. We show that none of the community detection algorithms tested provide consistently optimal results in all networks and that Surprise maximization, obtained by combining multiple algorithms, obtains quasi-optimal performances in these difficult benchmarks.

Introduction

Complex networks are widely used for modeling real-world systems in very diverse areas, such as sociology, biology and physics [1, 2]. It often occurs that nodes in these networks are arranged in tightly knit groups, which are called communities. Knowing the community structure of a network provides not only information about its global features, i.e., the natural groups in which it can be divided, but may also contribute to our understanding of each particular node in the network, because nodes in a given community generally share attributes or properties [3]. For these reasons, characterizing which are the best strategies to establish the community structure of complex networks is a fundamental scientific problem.

Many community detection algorithms have been proposed so far. The best way to sort out their relative performances is by determining how they behave in standard synthetic benchmarks, consisting of complex networks of known structure. There are two basic types of benchmarks, which we have respectively called *open* and *closed* [4–6]. Open benchmarks use networks with a community structure defined *a priori*, which is progressively degraded by randomly rewiring links in such a way that the number of connections among nodes in different communities increases and the network evolves toward an unknown, "open-ended" structure [5–11]. In open benchmarks, the performance of an algorithm can be measured by comparing the partitions that it obtains with the known, initial community structure, being increasingly difficult to recover that structure as the rewiring progresses. The first commonly used open benchmark was developed by Girvan and Newman (GN benchmark) [12]. It is based on a network with 128 nodes, each with an average number of 16 links, split into four equal-sized communities. It is however well established that the GN benchmark is too simple. Most algorithms are able to provide good results when confronted with it [7, 8]. Also, the fact that all communities are identical in size makes some algorithms that favor erroneous structures (e.g., those unable to detect communities that are small relative to the size of the network [6–8, 13–15]) to perform artificially well in this benchmark. These results indicated the need to develop more complex benchmarks. Lancichinetti, Fortunato and Radicchi suggested a new type of complex benchmarks, called LFR, which has obvious advantages over the GN benchmark [16]. In the GN networks, node degrees follow a Poisson distribution. However, in many real networks the degree distribution displays a fat tail, with a few highly connected nodes and the rest barely linked. This suggests that its distribution may be modeled according to a power law. In the LFR benchmarks, both the degrees of the nodes and the community sizes in the initial networks can be adjusted to follow power laws, with exponents chosen by the user. In this way, realistic networks with many communities can be built. LFR benchmarks are much more difficult than GN benchmarks, with many algorithms performing poorly in them [6, 8–11]. Notwithstanding these advantages, the parameters commonly used in the LFR benchmarks generate networks where all communities have similar sizes [4–6, 8].

This led to the proposal of a third type of benchmark, based on Relaxed Caveman (RC) structures [17]. In this type of benchmarks, the initial networks are formed by a set of isolated cliques, each one corresponding to a community, which are then progressively interconnected by rewiring links. The possibility of selecting the size of each initial clique makes the RC benchmarks ideal for building distributions of community sizes with a high variance, which constitute a very stern test for most algorithms [4–6].

In open benchmarks, when the original structure is largely degraded – and especially if the networks used in the benchmark are large and have a complex community structure – it generally happens that all algorithms suggest partitions different from the initial one. However, this can be due to two very different reasons: either the algorithms are not performing well or all/some of them indeed are optimally recovering the community structure present in the network, but that structure does not anymore correspond to the original one. The lack of a way to discriminate between these two potential causes is a limitation of all open benchmarks. To overcome this problem, we recently proposed a different type of benchmark, which we called closed [4, 5]. Closed benchmarks also start with a network with known community structure. However, the rewiring of the links is not random, as in open benchmarks. It is instead guided from the initial network toward a second, final network, which has exactly the same community structure that the initial one, but with the nodes randomly reassigned among communities. The rewiring process in these benchmarks is called Conversion (C), and ranges from 0 % to 100 %. When C = 50 %, half of the links that must be modified in the transition from the initial to the final networks have been already rewired and C = 100 % indicates that the final structure has been obtained.

The main advantage of the closed benchmarks is that it is possible to obtain quantitative information regarding whether a given partition is optimal or not. This happens because, along the conversion process, the network moves away from the initial structure at the same rate as it approaches the final one. We can take advantage of this feature with a type of analyses that is based on comparing partitions with a parameter called Variation of Information (VI; [18]). Being a metric [18], VI satisfies the triangle inequality: $\mathrm{VI}_{IE} + \mathrm{VI}_{EF} \geq \mathrm{VI}_{IF}$, where: 1) VI_{IE} is the variation of information for the comparison between the original community structure known to be present in the initial network (I) and the one deduced for an intermediate network (E), generated at a certain point of the conversion process; 2) VI_{EF} is obtained comparing that intermediate structure and the community structure of the final network (F), which is also known; and, 3) VI_{IF} is obtained when the initial and final structures are compared. An algorithm that performs optimally during the whole conversion process should generate solutions satisfying the equality $\mathrm{VI}_{IE} + \mathrm{VI}_{EF} = \mathrm{VI}_{IF}$ – where E is in this context the partition proposed by the algorithm – while deviations from this equality, which can be summarized with the value $\mathrm{VI}_{\delta} = \mathrm{VI}_{IF}$ - $(\mathrm{VI}_{IE} + \mathrm{VI}_{EF})$, indicate suboptimal performance [4, 5]. Another advantage of the closed benchmarks is that the identical community structure in the original and final networks implies a second quantitative feature: the solutions provided by an algorithm must be symmetrical along the conversion of one into the other. For example, at C = 50 %, a correct partition must be equally similar to both the initial and final networks. Finally, it is also significant to point out that closed benchmarks are very versatile, given that any network, for example those traditionally used in open GN, LFR or RC benchmarks, can be also analyzed in a closed configuration. All these features make the analysis of complex closed benchmarks the best test available to evaluate the performance of community detection algorithms.

All the analyses described so far, in both open and closed benchmarks, require the community structure to be known *a priori*. Additional useful information may be obtained by evaluating the results of the different algorithms with measures able to establish the quality of a partition by criteria that are independent of knowing the structures originally present in the networks. In the past, one such global measure of partition quality, called modularity [19], was extensively used. However, multiple works have shown that modularity-based evaluations are often erroneous [4,6,13–15]. In recent studies, we introduced a new global measure, called Surprise (S), which has an excellent behavior in all networks tested [4–6]. We have shown that S can be used to efficiently evaluate algorithms in open benchmarks and that, according to its results in those benchmarks, the best algorithm turned out to be combining multiple methods to maximize S [6]. These results suggest that Surprise may also contribute to evaluate algorithm performance in closed benchmarks and raise the question of whether S maximization could also be the best method to obtain optimal partitions in these complex benchmarks.

In this study, we carry out an extensive and detailed analysis of the behavior in closed benchmarks of a set of algorithms already used in open benchmarks in one of our recent papers [6]. Our work has three well-defined sections. First, we test all those strategies in both LFR and RC closed benchmarks, being able to identify the algorithms which perform well and those that perform poorly or are unstable. Second, we propose a novel approach to compare methods, which involves hierarchically clustering all their solutions. Applying this procedure at different stages of the closed benchmarks, we obtain a better understanding of how the algorithms behave. Finally, we show that, as already demonstrated in open benchmarks, Surprise maximization is the best strategy for community characterization in closed ones.

Methods

Algorithms and benchmarks used in this study

In this work, we evaluated 17 non-overlapping community detection algorithms, selected according to recent studies [Table 1; [4–10, 22, 30, 34]]. These algorithms were exactly the same used in [6], except that we had to discard here one of the programs (implementing an algorithm called MLGC), given that it was

Name	**Strategy used by the algorithm**	**References**
Blondel	Multilevel modularity maximization	[20]
CNM	Greedy modularity maximization	[21]
CPM	Multiresolution Potts model	[22]
DM	Spectral analysis + modularity maximization	[23]
EO	Modularity maximization	[24]
HAC	Maximum Likelihood	[25]
Infomap	Information compression	[26]
LPA	Label propagation	[27]
MLGC	Multilevel modularity maximization	[28]
MSG+VM	Greedy modularity maximization + refinement	[29]
RB	Multiresolution Potts model	[30]
RN	Multiresolution Potts model	[31]
RNSC	Neighborhood tabu search	[32]
SAVI	Optimal prediction for random walks	[33]
SCluster	Hierarchical Clustering + Surprise maximization	[34]
UVCluster	Hierarchical Clustering + Surprise maximization	[34, 35]
Walktrap	Random walks + modularity maximization	[36]

Table 1: Details of the algorithms used in this study. A description of the strategies implemented by the algorithms and the corresponding references are indicated.

unable to complete the analyses. In general, the default parameters of the algorithms were used. For the UVCluster and SCluster algorithms, we used UPGMA as hierarchical algorithm and Surprise as evaluation measure. RB and CPM have a tunable resolution parameter (γ) which defines the type of communities that they obtain. Since the optimal value for such parameter cannot be defined a priori in the absence of information about the community structure of the graph, we tested, for each network, a wide range of values of γ and chose as solution the most stable partition. The RB approach is equivalent to the original definition of modularity when $\gamma = 1$ [30], so we varied the parameter from 0 to as far as 5, ensuring a high coverage of the possible values of γ. In the case of the CPM algorithm, we used $0 \leq \gamma \leq 1$, the only defined range for unweighted networks [22].

Two very different types of networks were used as initial input for our closed benchmarks. The first were standard LFR networks containing 5000 nodes, which were divided into communities having between 10 and 50 nodes. The distribution of node degrees and community sizes were generated according to power laws with exponents -2 and -1, respectively. Since it was essential that the initial communities were well defined, we used a "mixing parameter" $\mu = 0.1$. This value means that in the starting networks each node shared only 10 % of its links with nodes in other communities [16]. As already indicated, LFR communities are small and very numerous, but their sizes are very similar, which may be a limitation. Pielou's index [37] can be used to measure the variation of community sizes. This index, which takes a value of 1 for networks with equal-sized communities, was 0.98 in these LFR benchmarks. We found also that it was higher than 0.95 for all the other standard LFR benchmarks of similar sizes used so far (unpublished data). Thus, we decided to use a second type of benchmark with networks having a much more skewed distribution of community sizes. To this end, we used the Relaxed Caveman (RC) configuration. The networks used in our RC benchmarks contained 512 nodes, split into 16 communities. The Pielou's Index for the distribution of their sizes was 0.75, meaning that the differences in community sizes were very high, spanning two orders of magnitude.

In order to control the intrinsic variation of our analyses, ten different networks with the features defined above were generated as starting points both for the LFR and for the RC configurations. For these 20 different closed benchmarks, we obtained 99 intermediate points between the initial and the final partitions, generated using conversion values ranging from C = 1 % to C = 99 % We expected many different structures, with varied properties, to be produced along these complex conversion processes, thus allowing a thorough test of the community structure algorithms.

Clustering of solutions

We devised an approach for algorithm evaluation in closed benchmarks that allows to compare their solutions and to easily visualize their relationships. In this type of analysis, all the partitions provided by the different algorithms for a given network plus four additional predefined structures were considered. These four structures were: 1) Initial and 2) Final, which respectively correspond to the community structures present at the beginning and the end of the conversion process; 3) One, which refers to a partition in which all nodes are in the same community; and, 4) Singles, which corresponds to a partition in which all communities have a single node. The method used was the following: we choose three conversion values (10 %, 30 % and 49 %) and we calculated the VI values obtained by comparing the partitions generated for a given network by all the algorithms to be tested plus the four predefined structures just indicated. To minimize the variance of the VI values, 100 different networks were analyzed for each conversion value. In this way, a matrix of VI values was obtained for each conversion level. Including the 4 preestablished structures, this matrix has $([k+4]*[k+3])/2$ values, being k the number of algorithms. The values of this VI matrix were then used as distances to perform agglomerative clustering using UPGMA [38]. In this way, dendrograms that graphically depicted the relative relationships among all partitions were obtained. Given that we are using distances, how similar are the solutions of the different algorithms can be precisely evaluated, by considering both the topology of the tree and how long the branches in these dendrograms are. As we will show in the Results section, the four predefined structures were included to be used as landmarks to interpret the dendrograms generated.

Surprise analyses

The quality of a partition can be effectively evaluated by its Surprise (S) value [6]. S is based on a cumulative hypergeometric distribution which, given a partition into communities of a network, computes its probability in a random network [4,35]. Let F be the maximum possible number of links in a network with n links, and M be the maximum possible number of intra-community links given that partition with p intra-community links. Surprise is then calculated with the following formula [4]:

$$S = -\log \sum_{j=p}^{min(M,n)} \frac{\binom{M}{j}\binom{F-M}{n-j}}{\binom{F}{n}} \tag{2.9}$$

The higher the S value, the more unlikely (or "surprising", hence the name of the parameter) is the observed distribution of intra- and intercommunity links, meaning that the communities obtained are maximally connected internally and also maximally isolated from each other.

Results

Detailed behavior of the algorithms

The 17 algorithms tested in the LFR and RC closed benchmarks showed very different behaviors, which are summarized in Figures 1 - 4. In these figures, following methods developed in previous works [4,5], we show the VI values comparing the partitions obtained by the algorithms with the known initial (red lines) and final (black lines) structures. A perfect agreement with any of these structures corresponds to VI = 0. Also, the value $(\mathrm{VI}_{IE}+\mathrm{VI}_{EF})/2$, (where E is the partition suggested by the algorithm, while I and F are, respectively, the initial and final partitions) is indicated with a blue line. As we discussed before, if the performance of an algorithm is optimal, then $\mathrm{VI}_{IE} + \mathrm{VI}_{EF} = \mathrm{VI}_{IF}$. This means that, in these representations, the blue line should, in the best case, be perfectly straight and located just on top of a thin dotted line also included in these figures, which corresponds to the value $\mathrm{VI}_{IF}/2$.

Figure 1 shows the behavior of the six algorithms in the LFR benchmarks that we considered the best, given that they were the only ones able to recover the initial partition when $\mathrm{C} \geq 5\,\%$. None of the other 11 algorithms recovered even a single optimal partition in the whole benchmark. Given that teh conditions used ($\mu = 0.1$, $\mathrm{C} = 5\,\%$) involved a limited number of intercommunity links, these results indicate that most algorithms performed deficiently. The six best algorithms worked however quite well, as indicated by the general closeness of their $(\mathrm{VI}_{IE} + \mathrm{VI}_{EF})/2$ values and the expected $\mathrm{VI}_{IF}/2$ values (Figure 1). Among these algorithms, Infomap [26] was the only one able to perform optimally or quasi-optimally along the whole conversion process, although, around $\mathrm{C} = 50\,\%$, a slight deviation was noticeable (see blue line in Figure 1). Infomap recognizes the initial communities until almost half of the benchmark (red line with values $\mathrm{VI} = 0$) and then, just after $\mathrm{C} = 50\,\%$, it suddenly starts detecting the final ones (as seen by the fact that the black line quickly drops to zero). This rapid change is explained by the very similar sizes of all the communities present in the LFR benchmarks, which are all destroyed at the same time and also rebuilt all together with their final structure as conversion proceeds. Two other algorithms, RB [30] and LPA [27] performed quite similarly to Infomap, again only failing in the central part of the benchmark. The behavior of the other three among the six best-performing algorithms (MCL [28], RN [31] and CPM [20]), was good at the beginning of the conversion process, but clearly worse than Infomap quite soon (Figure 1). Figure 2 shows the results for the other algorithms. In addition of all them not finding any optimal solutions, the worst ones showed highly unstable solutions (e. g. SAVI [33], MSG+VM [29]; notice the large mistakes in Figure 2) or totally collapsed, not finding any structure in these networks (e. g. CNM [21]). We conclude that the behavior of most of the algorithms tested is questionable when analyzed with precision in these difficult closed benchmarks.

In general, the results of the RC benchmark are similar. Again only six algo-

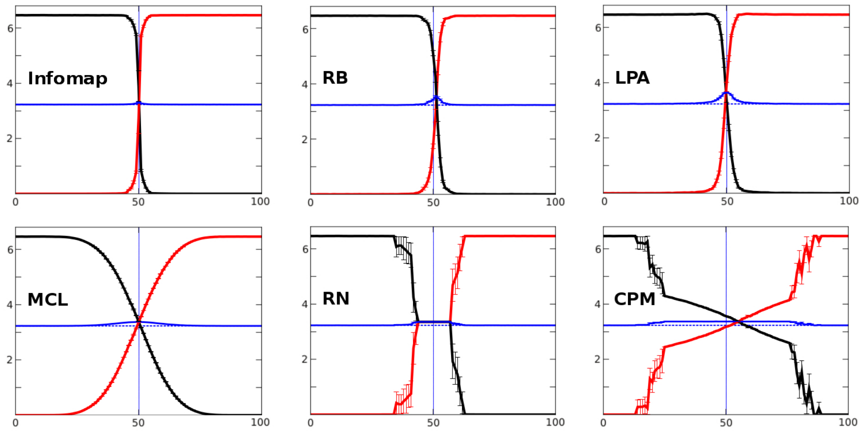


Figure 1: Best algorithms in LFR closed benchmarks. The six algorithms able to recover the initial partition when C $\leq 5\,\%$ are shown. In these diagrams, the x-axis shows the conversion percentage and the y-axis, the VI value. The red line indicates the VI values obtained when the algorithm solution is compared with the initial structure and the black line, the same comparison, but with the final structure. A perfect identity corresponds to the value VI $= 0$. Comparing the $(\text{VI}_{IE} + \text{VI}_{EF})/2$ values (blue line) and the $\text{VI}_{IF}/2$ values (dotted line, often invisible, being just below the blue one), we can conclude that Infomap, RB and LPA achieve optimal values until C is very close to $50\,\%$. MCL, RN and CPM work accurately only in the easiest analyses (both ends of the benchmark).

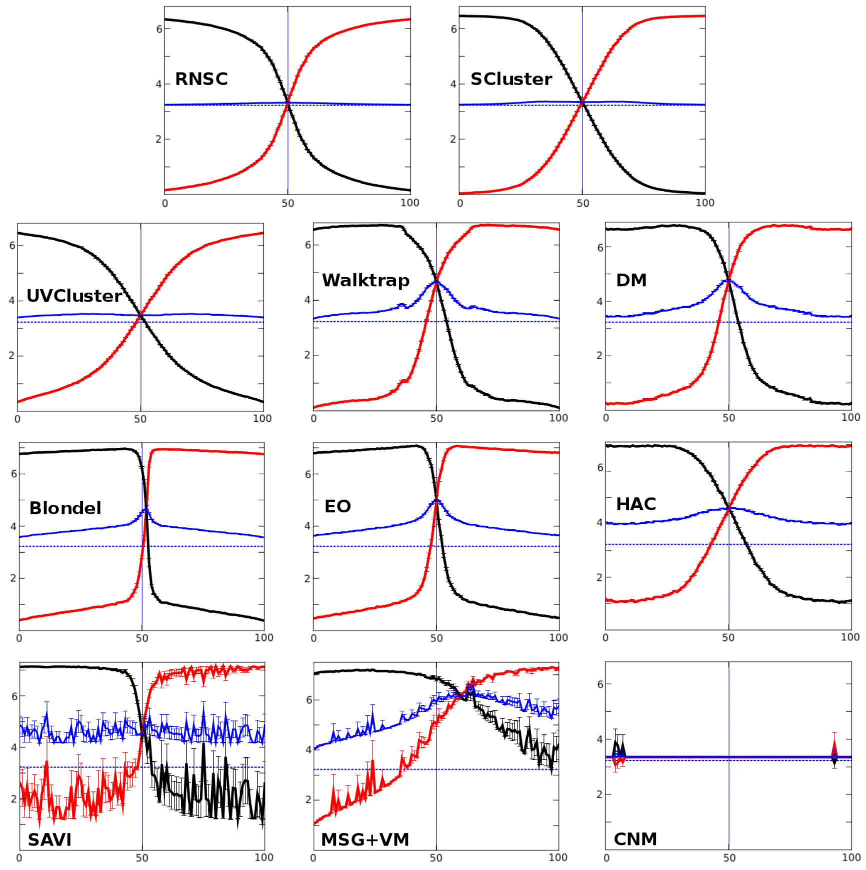


Figure 2: Poor performers in LFR closed benchmarks. These algorithms were unable to recover, even once, the correct partitions of the benchmark. The plots show their very diverse behaviors, ranging from results resembling somewhat those shown in Figure 1 (RNSC or SCluster) to others that are highly asymmetric (MSG+VM), unstable (SAVI) or correspond to algorithms that fail to find any structure (CNM).

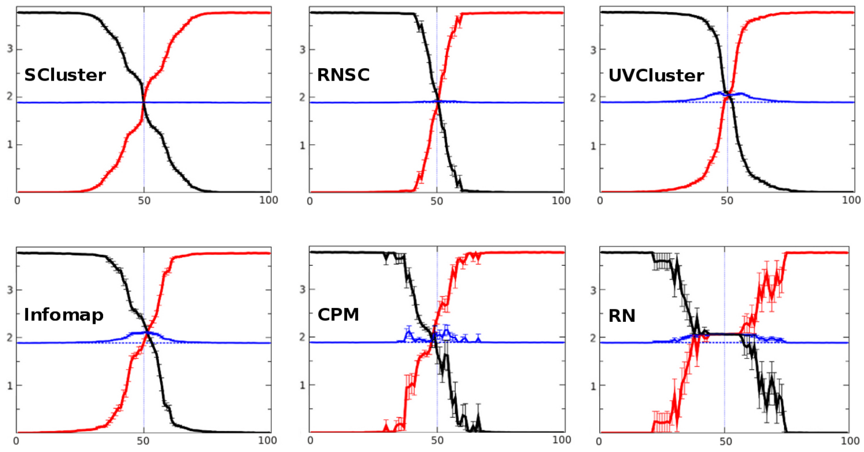


Figure 3: Best algorithms in RC closed benchmarks. As in Figure 1, this figure shows the six algorithms that recovered the initial partition when C $\leq$ 5 %. SCluster and RNSC showed an excellent behavior, displaying an almost straight blue line, while UVCluster failed in the central, most difficult, part of the benchmark. Infomap and CPM results were somewhat asymmetric, with the latter showing also some degree of instability. RN totally collapses when communities are not well defined.

rithms (Figure 3) provided correct values when C $\geq$ 5 %. Interestingly, just three, Infomap, RN and CPM, passed the C $\geq$ 5 % cut in both this benchmark and in the LFR benchmark (Figures 1 and 3). However, very significantly, none of these three were among the top performers in the RC benchmark. We found that only SCluster [34] and RNSC [32] achieved optimal VI values along most of the conversion process in the RC benchmark (see again the blue lines in Figure 3). The remaining four algorithms that passed the C $\geq$ 5 % cutoff (UVCluster [33, 34], Infomap, CPM and RN) worked well during the easiest parts of the benchmark but failed when conversion approached 50 %, in some cases showing asymmetries (CPM and Infomap) or instabilities (CPM and RN). These problems become much more noticeable in the worst algorithms, those that failed the 5 % conversion cut (Figure 4). Again, the results for these algorithms are quite poor. A final point is that, contrary to what we saw in the LFR benchmarks, a sudden swap from the initial to the final structure at around C = 50 % is not observed in the results provided by the best algorithms. This is explained by the greater variability in community sizes in the RC benchmarks respect to the LFR benchmarks. The RC communities disappear at different times of the conversion process.

Figures 5 and 6 show in more detail the deviations from the optimal values, indicated as $\mathrm{VI}_\delta = \mathrm{VI}_{IF}$ - $(\mathrm{VI}_{IE} + \mathrm{VI}_{EF})$, of the six best algorithms of each

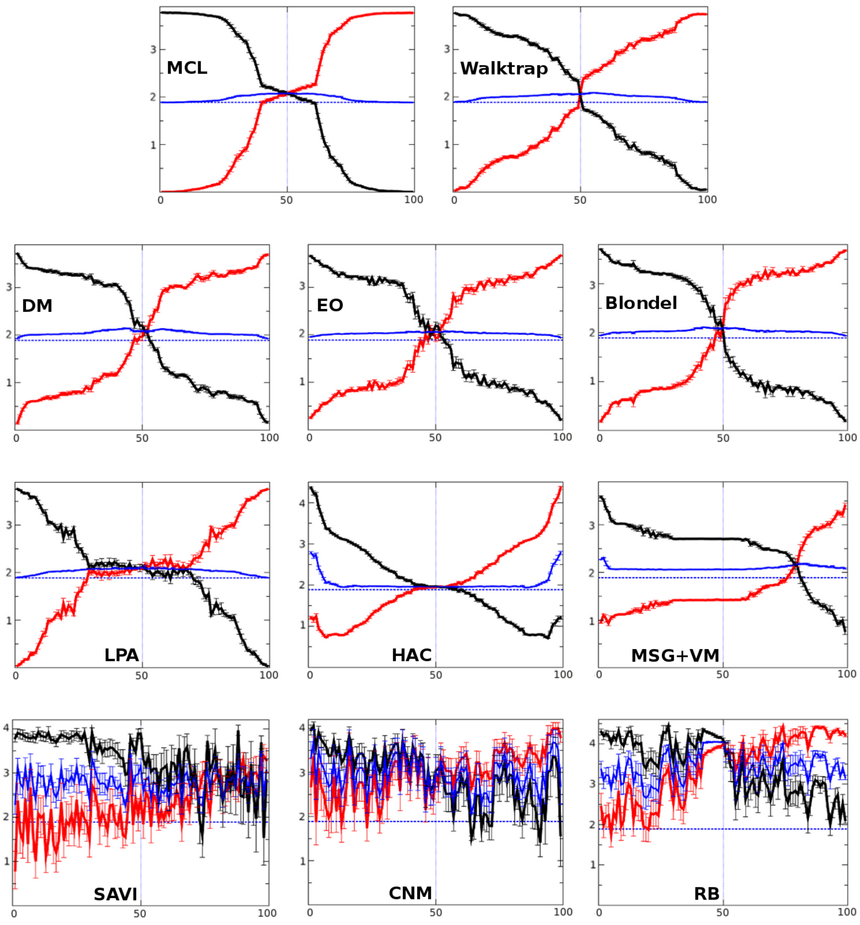


Figure 4: Algorithms that performed poorly in RC closed benchmarks. In this case, the behavior of the algorithms was worse than in the LFR benchmarks showed in Figure 2. MCL worked well only at the very beginning and the very end of the benchmark. The remaining algorithms performed much worse. In particular, MSG+VM showed a very asymmetric pattern and SAVI, CNM and RB results were chaotic.

benchmark. This value is equal to 0 when agreement with the optimal performance is perfect. The larger the deviations from the optimal behavior, the more negative are the values of VIδ. In the LFR benchmarks (Fig. 5), we confirmed that Infomap outperformed the other five algorithms. Its solutions were just very slightly different from the optimal ones around C = 50 %. The other algorithms displayed two different types of behaviors. On one hand, MCL, RB and LPA progressively separated from the optimal value toward the center of the benchmark. Notice, however, that this minimum should appear exactly when C = 50 %, this not being the case for RB, which showed slightly asymmetric results (Figure 5). On the other hand, RN and CPM reached a fixed minimum value that was maintained during a large part of the evolution of the network. This means that, during that period, these algorithms were constantly obtaining the same solution regardless of the network analyzed. In fact, RN always allocated all nodes to different communities while CPM split the 5000 nodes into variable groups, all them with one to four units. Figure 6 displays the analogous analyses for the RC benchmarks. We confirmed that SCluster and RNSC were clearly the best-performing strategies. The other algorithms satisfied the condition of optimality only when the network analyzed was very similar to either the initial or the final structure. This detailed analyses also showed more clearly something that could be suspected already looking at Figure 3, namely that RN and CPM produced abnormal patterns. The quasi-constant value of RN around C = 50 % is explained by the fact that all its solutions in the center of the benchmark consisted of two clusters, one of them containing more than 99 % of the nodes. On the other hand, CPM displayed an unstable behavior. The results in Figures 1-6 indicate that the RC benchmarks are at least as difficult as the LFR benchmarks, even though the number of nodes is much smaller (512 versus 5000). The considerable density of links and the highly skewed distribution of community sizes in the RC networks explain this fact.

Hierarchical analysis of the solutions provided by the different algorithms

As indicated in the Methods section, we obtained hierarchical clusterings of the VI values of the solutions of all the algorithms, together with four artificial partitions (Initial, Final, One and Singles). These analyses were focused on three different stages of the benchmark, C = 10 %, C = 30 % and C = 49 %. The first two were selected because they respectively corresponded to a low and medium degree of community structure degradation. We thought that any reasonable algorithm should easily recover the initial partition if C = 10 %, while the results shown in the previous section indicated that, when C = 30 %, the communities are fuzzier but still clearly detectable by several algorithms. Finally, when C = 49 %, the initial communities should be in the limit of being substituted by the final ones. However, good solutions should still be slightly more similar to the initial partition than to the final one. Figure 7 displays the dendrograms for those three stages

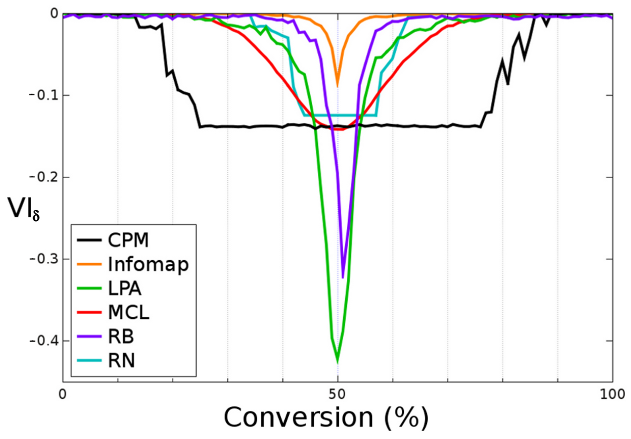


Figure 5: Details of the performance of the best algorithms in LFR benchmarks. The y-axis (VI$_\delta$) corresponds to the difference between the expected value, VI$_{IF}$ and the VI$_{IE}$ + VI$_{EF}$ value of the different solutions. VI$_\delta$ values close to zero correspond to the best performers.

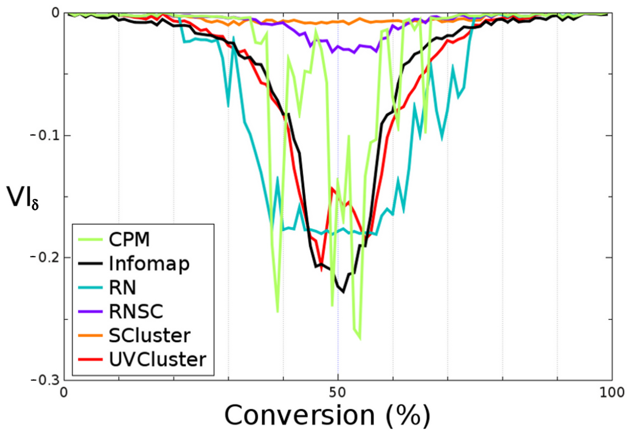


Figure 6: Detailed performance in the RC benchmarks. Again, the better a performance, the closer to a value equal to zero.

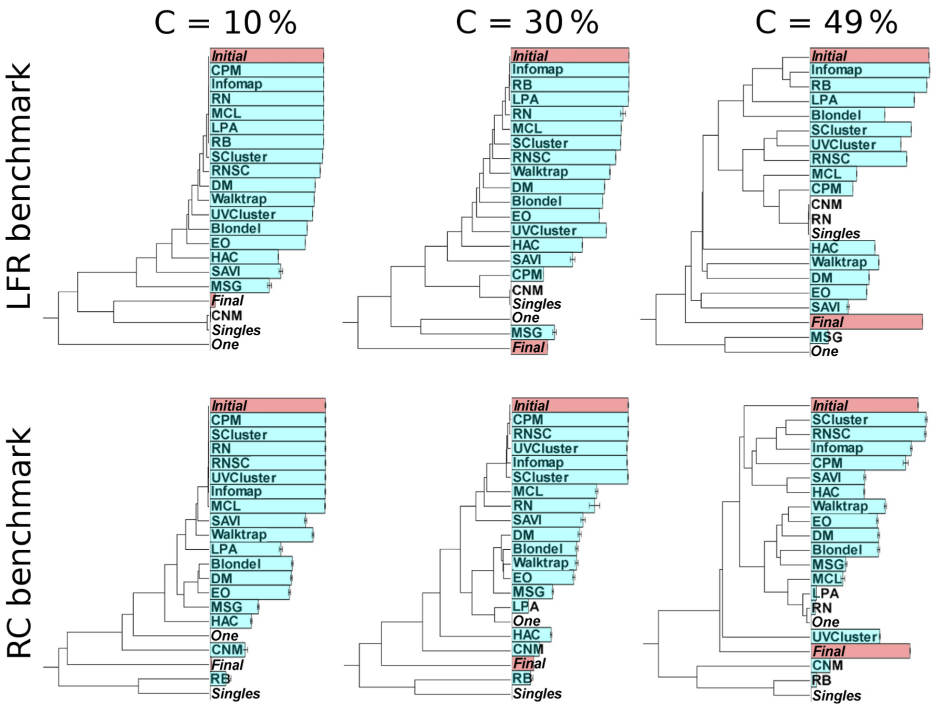


Figure 7: Hierarchical clustering of solutions. Dendrograms representing the hierarchical clustering of the solutions achieved by the different methods in LFR (top panels) and RC (lower panels) closed benchmarks. Three different stages of the network conversion process have been analyzed: C = 10 %, 30 % and 49 %. The four predefined structures (*Initial, Final, One and Singles*) are indicated in italics.

in both benchmarks, LFR and RC. We also include in that figure the Surprise values for each partition, as an independent measure of its quality (see below).

The LFR trees (Figure 7, top panels) display the behavior that could be expected after the detailed analyses shown in the previous section. Several of the best algorithms (e. g. Infomap, RB, LPA), appear in the tree very close to Initial even when C = 49 %, showing that they are indeed recognizing the initial structure or very similar ones along the whole benchmark. However, it is clear that the distances from Initial to the solutions provided by the different algorithms are growing with increasing values of C. This indicates that the structures recognized by even the best algorithms are not exactly identical to the original ones, in good agreement with the results shown in Figure 5. In the case of the RC benchmark (Figure 7, bottom panels), the results are somewhat more complex. When C = 10 % or C = 30 %, the situation is very similar to the one just described for the LFR benchmarks: the best algorithms generate solutions that are very similar to Initial, just as expected. However, when C = 49 % we found that the best algorithms in these benchmarks (SCluster, RNSC) generate solutions that are separated from Initial in the tree. Interestingly, their solutions cluster with those of other algorithms that also performed quite well in these benchmarks, such as Infomap or CPM. These results admit two explanations. The first one would be that the Initial structure (or a structure very similar to Initial) is still present, but all the algorithms have a similar flaw, which makes them find related, but false structures. The second is that, when C = 49 %, they are all recognizing a third type of structure, very different from Initial and Final, which is indeed the real one present in the networks. The first explanation is very unlikely given that these algorithms use totally unrelated strategies (Table 1). However, to accept the second one, we should have an independent confirmation that this may be the case.

Surprise values can be used to obtain such confirmation. In Figure 7, those values are also shown as horizontal bars with a size that is proportional to the S value obtained for each algorithm. As it can be easily seen in that figure, there is a strong correlation between the performance of an algorithm according to S values and its proximity to the Initial solution. This shows that S values are indeed indicating the quality of a partition with a high efficiency, as we already demonstrated in previous works [4-6]. Notice also that the S values for Initial and Final become more similar as the conversion progresses. This was expected, given that, at C = 50 %, the optimal partition should be exactly halfway between the initial and final community structures, and therefore, these values must then be identical. The fact that, in both the LFR and RC benchmarks with C = 49 %, there are structures different from the initial one is indicated by the S values for the Initial partition not being the highest. The S value of the Infomap partition is statistically significantly higher ($p = 0.0043$; t test) than Initial in the LFR benchmarks with C = 49 %. The same occurs in the RC benchmarks with C = 49 %: both the SCluster and the RNSC partitions have Surprise values significantly higher than the one found for Initial ($p < 0.0001$ in both cases; again, t tests were used). These results indicate that the top algorithms in these benchmarks are recognizing real, third-party structures, very different from Initial and Final, which emerged along the conversion process.

If the inclusion of Initial and Final was obviously critical for our purposes, the fact that we have also included One and Singles allows to easily visualize how some algorithms collapse, failing to find any significant structures in these networks. In the LFR benchmarks, this happens for CNM (already when C = 10 %), RN, CPM and MSG+VM. All of them generate partitions very similar to either One or Singles. In the RC benchmarks, this same problem occurs with RB (again already with C = 10 %), LPA, RN and CNM. We can conclude that these algorithms are often insensitive to the presence of community structure in a network.

Surprise maximization results

It is obvious from all the analyses shown so far that most algorithms performed poorly in these difficult benchmarks. Even those that worked very well in one type of benchmark often had serious problems detecting the expected partitions in the other one. In a recent work [6], we showed in open benchmarks that a meta-algorithm based on choosing for each network the algorithm that generated the solution with the highest Surprise value worked better than any isolated algorithm and provided values that were almost optimal. Here, following that same strategy, we confirmed those results in closed benchmarks. Figures 8a and 8b show the behavior of choosing the maximal value of Surprise (S_{max}) in, respectively, the LFR and the RC benchmarks. S_{max} values were obtained selecting solutions from six algorithms in the case of the LFR benchmark (ordered according to the number of times that they contribute to Smax, as follows: Infomap, RN, CPM, LPA, RB and MCL) and seven algorithms in the RC networks (i.e. CPM, RNSC, RN, SCluster, UVCluster, Infomap and MCL, ordered in the same way). All the other failed to provide any Smax values. As expected for a very good algorithm, the blue lines obtained for the Smax meta-algorithm are almost straight in both benchmarks (Figures 8a, 8b). If we measure the average distances to the dotted, optimal line, i.e. the average of VIδ for all conversion values, we found that it is minimal for the S_{max} meta-algorithm, and just slightly different from zero (Figure 8c), being clearly better than the results of all algorithms taken independently (also shown in Figure 8c).

Discussion

We recently showed that closed benchmarks have advantages over the commonly used open benchmarks to characterize the quality of community structure algorithms [5]. The main advantage is that the behavior of an algorithm can be more precisely understood by controlling the rewiring process, which leads to two

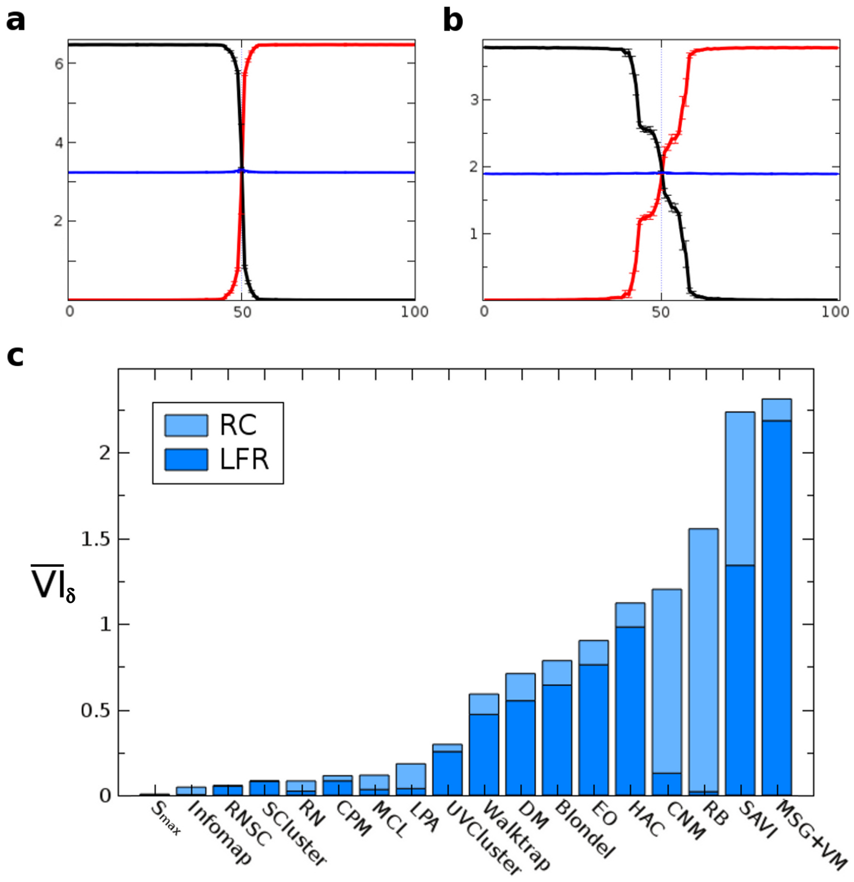


Figure 8: Results of the S_{max} meta-algorithm. Performance in LFR (panel a) and RC (panel b) benchmarks of the meta-algorithm that selected for each network the solution, among all the ones provided by the algorithms, which had the highest Surprise value. Panel c): Average values of the distance to the optimal performance (defined as the averages of the absolute values of VI_δ) for all the algorithms.

testable predictions that any good algorithm must comply. The first is just a general, qualitative feature, namely the symmetry respect to the initial and final configurations along the conversion process. The second prediction is much more precise, being based on the fact that the relationship $\mathrm{VI}_{IE} + \mathrm{VI}_{EF} = \mathrm{VI}_{IF}$ indicates optimal performance. These interesting properties of the closed benchmarks were already tested with a couple of algorithms in a previous work 5. Here, we extended those analyses to obtain a general evaluation of all the best available community structure algorithms in two types of closed benchmarks. The general conclusions of this work are the following: 1) Closed benchmarks can be used to quantitatively classify algorithms according to their quality; 2) None of the algorithms works efficiently in all benchmarks; 3) Surprise, a global measure of quality of a partition into communities, may be used to improve our knowledge of algorithm behavior; and, 4) Surprise maximization behaves as the best strategy in closed benchmarks, as it does in open ones 6. We will now discuss, in turn, these four conclusions.

Closed benchmarks allow for a much more detailed analysis of algorithm performance than open benchmarks

We have shown that algorithms can be easily classified according to their performance in closed benchmarks based on different parameters. As just indicated above, two of them ($\mathrm{VI}_{IE} + \mathrm{VI}_{EF} = \mathrm{VI}_{IF}$ relationship, expected symmetry of the results) were already described in our previous works. In addition to these two fundamental cues, additional parameters have been used for the first time in this work. Among them, we have first considered the ability of the algorithms to detect the initial community structure present in the networks when conversion starts growing. The critical value $\mathrm{C} \geq 5\,\%$ has been used as a cutoff value to select the best algorithms, given that those that do not recognize the original structure even when C is as low as $5\,\%$, are clearly poor performers. Another feature used here was $\mathrm{VI}\delta$, the distance to the optimal VI value, which was used both to explore in detail the behavior of the algorithms along the conversion process (Figures 5 and 6) or, as an average, to rate them in a quantitative way (Figure 8). Finally, a novel strategy, based on hierarchically classifying the algorithms using the VIs among their partitions as distances, has been also proposed (Figure 7). We have shown that it allows to determine the behavior of the algorithms, such as establishing that, at high C values, some algorithms group together, all proposing related community structures, which are however very different from both the initial and final ones (Figure 7). The combination of all these methods, and its complementation with Surprise analyses (see below), allow for a very precise characterization of the performance of the algorithms. These methods are much more complete than simply establishing how different from the initial structure is the solution proposed by an algorithm, as is currently done in all studies based on open benchmarks.

Most algorithms fail in closed benchmarks

If we now consider our results respect to how the algorithms performed, we must be pessimistic. Only three algorithms, Infomap, RN and CPM, passed the first cutoff, i.e., optimal performance beyond C = 5 %, in both LFR and RC benchmarks (Figures 1, 3). Further analyses showed that others, such as RNSC, SCluster, MCL, LPA or UVCluster work reasonably in average (Figure 8). However, they typically perform well in one of the benchmarks, but poorly in the other one (see Figures 1 - 4). Finally, a single algorithm, RB, works very well in the LFR benchmarks, but chaotically in the RC benchmarks (as becomes clear in the results shown in Figures 1 and 4 and quantitatively evaluated in Figure 8). This behavior is caused by the inability of this particular multiresolution algorithm to detect the communities of very different sizes present in the RC benchmarks [14, 15]. The other algorithms failed to recover accurate solutions in both the LFR and the RC benchmarks (Figures 2, 4, 7): in addition to their general lack of power to find the subtle structures present in these benchmarks when C increases, they often showed asymmetries, which we noticed were sometimes caused by a dependence of the results on the order in which the nodes were read by the programs (not shown).

Several papers have examined many of the algorithms used here in open GN, LFR and RC benchmarks. The general conclusions of those works can be summarized as follows: 1) As indicated already in the Introduction section, the GN benchmark is too easy, with most algorithms doing well [7, 8] while the LFR and RC benchmarks are much more difficult, with many algorithms working poorly [5,6,8]. This means that tests on the GN benchmark should not be used to support that new algorithms perform well; 2) Among the ones tested here, Infomap is the best algorithm for LFR benchmarks, with several others (RN, RB, LPA, SCluster) following quite closely [6, 8–10]; 3) However, SCluster, RNSC, CPM, UVCluster and RN are the best algorithms in RC benchmarks [5, 6]. Therefore, the agreement of the results in LFR and RC open benchmarks is far from complete; 4) All modularity maximizers behave poorly [6, 9, 10]. These results are in general congruent with the ones obtained here in closed benchmarks, but some significant differences in the details have been observed. Comparing the results of the 17 algorithms analyzed here using closed benchmarks (Figure 8) with the performance of those same algorithms in open benchmarks that start with the same exact network [6], we found that the top four average performers (Infomap, RN, RNSC and SCluster) were exactly the same in both types of benchmark. However, several algorithms (most clearly, RB and SAVI) performed worse here. These poor performances of RB and SAVI were due to their unstable behavior in RC benchmarks (Figure 4). These results indicate that these closed benchmarks can provide more information than the corresponding open ones, and thus they can be extensively used for testing community detection algorithms.

Surprise can be used to refine algorithm evaluation

In recent works, we have shown that Surprise (S) is an excellent global measure of the quality of a partition [4–6]. In this work, we have taken advantage of that fact to improve our understanding of how algorithms behave. The combination of the hierarchical analyses described above with Surprise calculations have allowed to establish the presence of third-party community structures that the best algorithms find, and which are different from both the initial and final structures defined in the benchmarks (Figure 7). These differences are small in the LFR benchmark, in which the best algorithms, Infomap and RB, suggested community structures which are very similar to the initial one, even when C = 49 % (Figure 7, top). They are however quite considerable in the RC benchmark, in which the best algorithms, SCluster and RNSC, plus several other among the best performers, appear together in a branch distant from the initial structure when C = 49 % (Figure 7, bottom).

Surprise maximization as the strategy of choice for community structure characterization

In previous works, we proposed that, given that Surprise is an excellent measure for the quality of a partition into communities, a good strategy for obtaining that partition would involve maximizing S. However, S-maximizing algorithms do not yet exist. So far, only UVCluster and SCluster use Surprise maximization as a tool to select the best partition among those found in the hierarchical structures that those algorithms generate [34, 35], but the true S_{max} partition is often not found with those strategies (as shown in refs. [4–6] and this work). Given that we have not yet developed an S_{max} algorithm, we decided to use a meta-algorithm that involves choosing among all the available algorithms, the one that produced the highest S value. This simple strategy was recently shown to outperform all known algorithms in open benchmarks [6]. In this work, we have shown that the same occurs in closed benchmarks (Figure 8). Even more significant is the fact that, both in open and closed benchmarks, there is only a limited room for further improvement: by combining several algorithms using their S values as a guide, we obtain performances which are almost optimal (see [6] and Figure 7). Therefore, in total agreement with our previous results, we conclude here that S maximization is the best available strategy for characterizing the community structure of complex networks, outperforming all the other algorithms analyzed so far. The interest of generating S-maximizing algorithms, which could improve even on the combined strategy or meta-algorithm used so far in our works, is clear.

Conclusions

In summary, we have shown the advantages of these strategies and of using complex closed benchmarks for community structure characterization and the potential of Surprise-based analyses for complementing those tests. We have also shown that all currently proposed algorithms, even the best ones, fail to some extent in these critical benchmarks and that a Surprise maximization meta-algorithm outperforms all them. The heuristic potential of these closed benchmarks is clear. They can be used in the future by anyone interested in checking the quality of an algorithm. A program to generate the conversion process typical of the closed benchmarks that can be applied to any network selected by the user is freely available at `https://github.com/raldecoa/ClosedBenchmarks`.

References

[1] S.N. Dorogovtsev and J.F.F. Mendes. *Evolution of networks: From biological nets to the Internet and WWW.* Oxford University Press, Oxford, 2003.

[2] M. Newman. *Networks: an introduction.* Oxford University Press, Oxford, 2010.

[3] S. Fortunato. Community detection in graphs. *Physics Reports*, 486:75–174, 2010.

[4] R. Aldecoa and I. Marín. Deciphering network community structure by surprise. *PloS ONE*, 6:e24195, 2011.

[5] R. Aldecoa and I. Marín. Closed benchmarks for network community structure characterization. *Physical Review E*, 85:026109, 2012.

[6] R. Aldecoa and I. Marín. Surprise maximization reveals the community structure of complex networks. *Scientific Reports*, 3:1060, 2013.

[7] L. Danon, A. Díaz-Guilera, J. Duch, and A. Arenas. Comparing community structure identification. *Journal of Statistical Mechanics: Theory and Experiment*, page P09008, 2005.

[8] A. Lancichinetti and S. Fortunato. Community detection algorithms: a comparative analysis. *Physical Review E*, 80:056117, 2009.

[9] G.K. Orman, V. Labatut, H. Cherifi, et al. On accuracy of community structure discovery algorithms. *Journal of Convergence Information Technology*, 6:283–292, 2011.

[10] G.K. Orman, V. Labatut, and H. Cherifi. Comparative evaluation of community detection algorithms: a topological approach. *Journal of Statistical Mechanics*, page P08001, 2012.

[11] A. Lancichinetti, F. Radicchi, J.J. Ramasco, and S. Fortunato. Finding statistically significant communities in networks. *PloS one*, 6:e18961, 2011.

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

[13] S. Fortunato and M. Barthelemy. Resolution limit in community detection. *Proceedings of the National Academy of Sciences*, 104:36–41, 2007.

[14] A. Lancichinetti and S. Fortunato. Limits of modularity maximization in community detection. *Physical Review E*, 84:066122, 2011.

[15] J. Xiang and K. Hu. Limitation of multi-resolution methods in community detection. *Physica A*, 391:4995–5003, 2012.

[16] A. Lancichinetti, S. Fortunato, and F. Radicchi. Benchmark graphs for testing community detection algorithms. *Physical Review E*, 78:046110, 2008.

[17] D.J. Watts. *Small worlds: the dynamics of networks between order and randomness*. Princeton university press, 2003.

[18] M. Meilă. Comparing clusterings—an information based distance. *Journal of Multivariate Analysis*, 98:873–895, 2007.

[19] M.E.J. Newman and M. Girvan. Finding and evaluating community structure in networks. *Physical review E*, 69:026113, 2004.

[20] V.D. Blondel, J.L. Guillaume, R. Lambiotte, and E. Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics*, page P10008, 2008.

[21] A. Clauset, M.E.J. Newman, and C. Moore. Finding community structure in very large networks. *Physical review E*, 70:066111, 2004.

[22] VA Traag, P. Van Dooren, and Y. Nesterov. Narrow scope for resolution-limit-free community detection. *Physical Review E*, 84:016114, 2011.

[23] L. Donetti and M.A. Munoz. Detecting network communities: a new systematic and efficient algorithm. *Journal of Statistical Mechanics*, page P10012, 2004.

[24] J. Duch and A. Arenas. Community detection in complex networks using extremal optimization. *Physical review E*, 72:027104, 2005.

[25] Y. Park and J.S. Bader. Resolving the structure of interactomes with hierarchical agglomerative clustering. *BMC bioinformatics*, 12:S44, 2011.

[26] M. Rosvall and C.T. Bergstrom. Maps of random walks on complex networks reveal community structure. *Proceedings of the National Academy of Sciences*, 105:1118–1123, 2008.

[27] U.N. Raghavan, R. Albert, and S. Kumara. Near linear time algorithm to detect community structures in large-scale networks. *Physical Review E*, 76:036106, 2007.

[28] A.J. Enright, S. Van Dongen, and C.A. Ouzounis. An efficient algorithm for large-scale detection of protein families. *Nucleic acids research*, 30:1575–1584, 2002.

[29] P. Schuetz and A. Caflisch. Efficient modularity optimization by multistep greedy algorithm and vertex mover refinement. *Physical Review E*, 77:046112, 2008.

[30] J. Reichardt and S. Bornholdt. Statistical mechanics of community detection. *Physical Review E*, 74:016110, 2006.

[31] P. Ronhovde and Z. Nussinov. Local resolution-limit-free potts model for community detection. *Physical Review E*, 81:046114, 2010.

[32] AD King, N. Pržulj, and I. Jurisica. Protein complex prediction via cost-based clustering. *Bioinformatics*, 20:3013–3020, 2004.

[33] E. Weinan, T. Li, E. Vanden-Eijnden, et al. Optimal partition and effective dynamics of complex networks. *Proceedings of the National Academy of Sciences*, 105:7907–7912, 2008.

[34] R. Aldecoa and I. Marín. Jerarca: Efficient analysis of complex networks using hierarchical clustering. *PloS ONE*, 5:e11585, 2010.

[35] V. Arnau, S. Mars, and I. Marín. Iterative cluster analysis of protein interaction data. *Bioinformatics*, 21:364–378, 2005.

[36] P. Pons and M. Latapy. Computing communities in large networks using random walks. *Computer and Information Sciences-ISCIS 2005*, pages 284–293, 2005.

[37] ECJ Pielou. The measurement of diversity in different types of biological collections. *Journal of Theoretical Biology*, 13:131–144, 1966.

[38] R.R. Sokal. A statistical method for evaluating systematic relationships. *Univ Kans Sci Bull*, 38:1409–1438, 1958.

Capítulo 3

DISCUSIÓN

Los resultados obtenidos se pueden dividir en tres grandes partes. Primero, la generación y optimización de una serie de algoritmos que posibilitan una detección eficiente de las comunidades de una red. Segundo, el desarrollo de un nuevo tipo de *benchmarks*, con importantes ventajas sobre los existentes en la literatura, donde testar la multitud de métodos publicados para extraer comunidades en redes. Y por último, la confirmación de que *Surprise* es una excelente medida estadística para evaluar, de forma precisa, la calidad de la partición en comunidades de una red. Además, los análisis realizados para llevar a cabo cada uno de estos tres bloques, han permitido probar y evaluar un buen número de algoritmos de detección de comunidades. A continuación, se detallan pormenorizadamente estos resultados.

Jerarca: Generación y mejora de algoritmos de detección de comunidades

Mayor eficiencia

Jerarca es una *suite* de algoritmos de *clustering*, que permite al usuario extraer información sobre la topología de una red, su estructura jerárquica y sus comunidades más significativas. Como se indicó en la sección 1.2.1 de la introducción, la heurística de los algoritmos desarrollados se basa en la estrategia conocida como *clustering jerárquico iterativo* [Arnau et al. 2005]. Jerarca implementa una nueva versión de UVCluster, aunque significativamente más eficiente. En la versión original, se necesitaba ejecutar internamente el algoritmo de Floyd [Floyd 1962] para calcular las distancias entre los nodos de la red. En esta nueva implementación, en cada iteración sólo se tienen en cuenta los nodos adyacentes a un nodo dado, con lo cual no es necesario conocer los caminos más cortos entre cada par de nodos de la red. De este modo, la complejidad computacional pasa de $O(n^3)$

a $O(n^2)$. Esto ha permitido realizar durante esta tesis análisis de miles de redes de 5000 unidades [Aldecoa y Marín 2013, Aldecoa y Marín 0000], mientras que el máximo tamaño analizado con la versión anterior había sido de menos de 700 nodos [Marco y Marín 2009].

Nuevos algoritmos

Al iniciar una iteración de UVCluster cada nodo forma su propia comunidad. A continuación, se elige un nodo al azar y se van agregando nodos a esa comunidad siempre y cuando cada nodo añadido tenga conexiones con todos los nodos de la comunidad. Es decir, la comunidad se forma, vorazmente, a partir de un *clique* (subgrafo completo) creciente. Cuando no existe ningún nodo agregable más, la comunidad está completa y se empieza el proceso de nuevo, eligiendo un nodo de los restantes al azar.

Además de esta estrategia de UVCluster, Jerarca incluye dos nuevos algoritmos, RCluster y SCluster. Las iteraciones de RCluster también comienzan con cada nodo aislado en su comunidad. A partir de ahí y hasta un cierto criterio de parada, se seleccionan dos comunidades al azar y se fusionan si todos los nodos de una de ella están conectados con todos los de la otra. Es decir, por definición, cada comunidad tiene todos sus nodos formando un *clique* y se unirá con otra si y sólo si su unión forma de nuevo un *clique*. Cuando ya no quedan más comunidades que se puedan unir, la partición está finalizada y se empieza una nueva iteración. Como se puede intuir, RCluster, del mismo modo que UVCluster, genera comunidades que son siempre *cliques*. La novedad de RClsuter es que los *cliques* se forman de manera aleatoria, todos a la vez y no de uno en uno, como en UVCluster. Sin embargo, el hecho de que RCluster tenga un coste computacional mayor, unido a que sus resultados no sean significativamente mejores que los de UVCluster, nos hizo descartarlo para análisis de redes superiores a unos pocos centenares de nodos. No obstante, sigue siendo de interés si se quiere realizar un análisis exhaustivo de una red concreta, ya que puede ser capaz de detectar comunidades con propiedades distintas de las que detectarían los demás algoritmos incluidos en Jerarca.

Por su parte, SCluster, tras inicializar cada nodo en una comunidad distinta, selecciona uno de ellos y agrega a su comunidad a todos sus vecinos. Una vez hecho esto, la comunidad está completa y se pasa a seleccionar otro nodo de entre los restantes al azar. Se repite este proceso hasta que no quedan nodos seleccionables. Este proceso, más rápido que los anteriores, consigue un coste computacional de $O(n \log n)$, lo que permite realizar análisis de redes con decenas de miles de unidades en muy poco tiempo. Aunque la estrategia es de una gran simplicidad, ha sido el algoritmo de los incluidos en Jerarca que ha mostrado un mejor comportamiento en la gran mayoría de las redes. Los únicos casos observados en los que UVCluster es superior es cuando la densidad de conexiones en la red es muy baja y la estructura de comunidades es prácticamente inexistente (véanse las Figuras 2 y S1 en el artículo original de Jerarca). La explicación se debe a que en esos momentos, el prototipo de comunidad definido intrínsecamente en SCluster es demasiado laxo e incapaz de reflejar correctamente las relaciones vecinales entre nodos. UVCluster por el contrario, y como ya hemos indicado, define las comunidades en forma de *cliques* crecientes [Arnau et al. 2005] y de este modo puede extraer de este tipo de redes al menos los triángulos y pequeños cliques que se puedan formar.

En nuestra experiencia, tanto SCluster como UVCluster se encuentran entre los mejores algoritmos de detección de comunidades [Aldecoa y Marín 2011, Aldecoa y Marín 2012, Aldecoa y Marín 2013, Aldecoa y Marín 0000]. Un problema habitual en detección de comunidades es que muchas de las estrategias comúnmente utilizadas tienen dificultades para reconocer particiones cuyas comunidades tienen tamaños muy distintos. Es en este tipo de redes donde ambos algoritmos superan a la mayoría de métodos publicados hasta el momento. En particular, SCluster aparece como el mejor algoritmo de los publicados hasta el momento en los benchmarks RC que hemos testado [Aldecoa y Marín 2013, Aldecoa y Marín 0000].

Por otro lado, el programa original UVCluster realizaba el *clustering* jerárquico de la matriz de distancias secundarias exclusivamente mediante UPGMA [Sokal y Michener 1958]. Jerarca también contiene esta estrategia y además añade Neighbor-Joining [Saitou y Nei 1987] (un algoritmo muy popular en el ámbito del análisis filogenético) como segunda opción. La inclusión de otra alternativa se debe a dos razones. Por un lado, en muchas ocasiones, UPGMA no es capaz de construir el mejor árbol a partir de una matriz de distancias, siendo a menudo superado por Neigbor-Joining [Saitou y Nei 1987]. Por otro lado, al disponer de dos árboles, que en la mayoría de los casos serán distintos, podemos evaluar el doble de particiones a la hora de determinar las comunidades óptimas de la red.

Evaluación de la estructura de comunidades

Una vez extraído el árbol jerárquico en el que el proceso de *clustering jerárquico iterativo* transforma la red, los cortes a diferentes alturas del dendrograma se corresponden con diferentes particiones de la red en comunidades. Por tanto, asumiendo que la jerarquía es correcta, es posible definir como mejor partición de la red aquel nivel del árbol cuyas comunidades estén mejor definidas. Para ello, en el artículo original de UVCluster [Arnau et al. 2005] se propone una medida de calidad basada en una distribución hipergeométrica que, como ya hemos indicado en la Introducción, más tarde modificamos ligeramente y llamamos *Surprise*. En Jerarca, además de esta medida, incluimos también la evaluación de las particiones mediante *Modularity* (Q). Aunque más tarde demostramos que Q no es una buena medida para este fin, el programa ofrece al usuario la posibilidad de extraer la mejor partición de la red con una u otra estrategia. Además, la posibilidad de seleccionar ambos índices como medida de calidad nos ha permitido, en todos los trabajos posteriores, comparar fácilmente los resultados de *Modularity* y *Surprise* en distintas redes, simplemente ejecutando Jerarca en cada una de ellas.

Fácil interpretación de los resultados

Jerarca también presenta nuevas ventajas para el usuario final, generando ficheros que pueden ser trasladados posteriormente a programas bien conocidos como MEGA [Tamura et al. 2007] o Cytoscape [Smoot et al. 2011]. Para el primero, Jerarca produce ficheros que contienen las descripciones de los árboles jerárquicos generados. Por otro lado, los ficheros creados como entrada para Cytoscape contienen la asignación de cada nodo a su comunidad, con lo cual puede visualizarse gráficamente (por ejemplo, mediante diferentes colores) la estructura de comunidades de la red.

Por último, cabe destacar que Jerarca permite ejecutar con un solo comando todas las posibles combinaciones de los distintos algoritmos -tres iterativos (UVCluster, RCluster y SCluster), dos jerárquicos (UPGMA y Neighbor-Joining)- y las dos medidas de calidad de las particiones (Q y S). De este modo el usuario obtiene un amplio abanico de soluciones con las que trabajar. Dado que los tres algoritmos iterativos (UVCluster, RCluster y SCluster) son altamente paralelizables (ya que cada iteración es completamente independiente de las demás), en un futuro próximo Jerarca podría ser capaz de analizar redes de cualquier tamaño. También hay que resaltar la estructura modular del código de Jerarca, la cual permite fácilmente añadir nuevos algoritmos o funciones de evaluación de las comunidades al programa. Por todo lo comentado anteriormente, podemos afirmar que Jerarca cumple con los dos objetivos para los que fue diseñado. Por un lado, asienta la estrategia de *clustering jerárquico iterativo* propuesta en nuestro anterior trabajo [Arnau et al. 2005], añadiendo nuevos algoritmos y análisis alternativos, y por otro nos ofrece una poderosa herramienta, fácil de utilizar para el usuario, para detectar comunidades en redes complejas, la cual está a la altura de algunos los mejores algoritmos propuestos hasta el momento. El código del programa, liberado bajo licencia GPLv3, puede ser libremente descargado en `http://jerarca.sourceforge.net`.

Desarrollo de un nuevo tipo de *benchmarks*

Hemos propuesto un nuevo tipo de *benchmarks*, denominados *cerrados* [Aldecoa y Marín 2012]. Este nuevo modelo direcciona el barajeo de las conexiones hacia una estructura final predefinida, en vez de degradar las comunidades hacia un grafo aleatorio como ocurre en los *benchmarks* tradicionales. Esta segunda red es estructuralmente idéntica a la inicial, lo que favorece que se produzca un comportamiento simétrico conforme las conexiones se barajean desde la estructura inicial a la final. Además, para inicializar el *benchmark*, el usuario puede utilizar cualquier tipo de red, lo que permite testar el comportamiento de un algoritmo en diferentes topologías.

La característica más importante de un *closed benchmark* es poder afirmar si un algoritmo se ajusta al comportamiento que debería seguir en caso de ser óptimo. Durante el proceso de conversión de la estructura inicial (I) a la final (F), esperamos que la red intermedia (E) se aleje de I a la misma velocidad que se acerca a F. Por tanto, utilizando la Variación de Información [Meilă 2007] como medida de distancia, esperamos que si se cumple la propiedad citada, la suma de las distancias de I a E (VI_{IE}) y de E a F (VI_{EF}) sea igual a la distancia de I a F (VI_{IF}). De este modo, para un algoritmo óptimo, se debería cumplir $\mathrm{VI}_{IE} + \mathrm{VI}_{EF} = \mathrm{VI}_{IF}$ durante todo el *benchmark*. Además, calculando la distancia entre el lado izquierdo y el derecho de esa ecuación (a la cual llamamos VI_{δ} en el artículo técnico), podemos obtener un valor que nos indica cuán alejada está una partición estimada (E) del comportamiento óptimo. De esta manera, podemos utilizar VI_{δ} para evaluar y clasificar algoritmos [Aldecoa y Marín 2012]. Por otro lado, también hay ciertas propiedades que un buen algoritmo debe satisfacer en estos *benchmarks*, como que su comportamiento sea simétrico respecto al 50 % de conversión de una red en otra y que durante la primera mitad de este proceso sus soluciones obtenidas sean más similares a la partición inicial que a la final.

Como se puede apreciar, desde un punto de vista teórico existen importantes diferencias respecto a todos los *benchmarks* publicados anteriormente. Además, las gráficas obtenidas mediante análisis empíricos de diferentes algoritmos, muestran diferencias entre sus comportamientos que no se podrían apreciar utilizando *open benchmarks* [Aldecoa y Marín 2012, Aldecoa y Marín 0000]. Todas estas propiedades hacen de los *closed benchmarks* una herramienta de gran interés para evaluar detalladamente algoritmos de detección de comunidades. Puede encontrarse un *script* en Perl para generar *closed benchmarks* en `https://github.com/raldecoa/ClosedBenchmarks`.

Surprise: Una excelente medida para evaluar la calidad de una partición

Esta es, sin duda, la mayor aportación de esta tesis doctoral al campo de estructura de comunidades. Establecer una medida que evalúe correctamente la calidad de una partición en comunidades es crítico en multitud de areas científicas. Un objetivo principal de nuestro trabajo era comprobar si dicha evaluación se podía llevar a cabo eficientemente utilizando una distribución hipergeométrica para calcular la "rareza" de la disposición de las conexiones de una partición observada, como sugerimos en [Arnau et al. 2005]. A lo largo del trabajo hemos testado ampliamente y bajo multitud de redes con diferentes topologías esta medida, a la que hemos denominado *Surprise* (S), debido a que evalúa cuán sorprendente (es decir, cuán improbable) es la distribución de conexiones dentro y fuera de las comunidades de una partición dada.

En un primer artículo mostramos cómo este índice funcionaba apropiadamente y sus resultados eran mejores que los de *Modularity* (Q), la medida más popular y utilizada en este contexto desde 2004 [Aldecoa y Marín 2011]. Ambas estrategias fueron testadas en dos *benchmarks* con distintas propiedades (LFR y RC) y bajo dos configuraciones distintas (*open* y *closed*). Los resultados de S son mejores que los de Q en la gran mayoría de los casos. Además, aplicando *Surprise* a redes reales obtuvimos resultados excelentes, como en el caso de una red de interacción proteína-proteína en *Saccharomyces cerevisiae* o en una red de equipos de fútbol americano muy conocida [Aldecoa y Marín 2011]. El único caso donde las comunidades devueltas por *Surprise* no fueron las esperadas *a priori* fue en la famosa red de un club de kárate recopilada por W.W. Zachary [Zachary 1977]. Sin embargo, prácticamente ningún método de los utilizados en la literatura detecta dos comunidades y no existe un consenso sobre cual es la estructura de comunidades real de la red.

Viendo el potencial de esta medida y para comprobar si realmente *Surprise* podría ser una buena candidata para resolver el problema de la evaluación de la calidad de una partición, decidimos llevar a cabo análisis más exhaustivos. En un trabajo publicado recientemente [Aldecoa y Marín 2013], utilizamos para ello dos *benchmarks* muy distintos (LFR y RC), en sus formas *open* y *closed* y 18 de los mejores algoritmos de detección de comunidades. Para conseguir particiones de la red con altos valores de *Surprise*, elegimos de entre las soluciones de todos algoritmos, siempre aquella de mayor S. Los resultados son contundentes. Utilizando esta estrategia, obtenemos mejores resultados que con cualquiera de los algoritmos por separado. Además las soluciones son, en la gran mayoría de casos, idénticas a las esperadas (i.e., a las particiones originales de los *benchmarks*). Investigando las pocas situaciones en las que no ocurre esto, pudimos observar que era debido a unos pocos nodos, siempre perteneciente a pequeñas comunidades que se veían ligeramente modificadas. El hecho de que esto ocurriese en casos donde la degradación de la red era muy alta y, sobre todo, que dichas particiones discordantes fuesen detectadas a la vez por varios algoritmos, nos dice que probablemente éstas fuesen ahora las comunidades reales existentes y no las esperadas inicialmente [Aldecoa y Marín 2013].

Por último, testamos *Surprise* en redes donde otros índices, como Q o estrategias multiresolución basadas en modelos de Potts, fracasan al recuperar la estructura de comunidades. El primer escenario es el denominado *anillo de cliques*, donde Q fusiona comunidades pequeñas debido a su *límite de resolución* [Fortunato y Barthelemy 2007]. Tras evaluar redes de hasta un millón de nodos generadas con esa topología, *Surprise* fue capaz de reconocer las comunidades esperadas sin ningún problema. El segundo escenario presenta una red simple pero con comunidades de tamaños muy distintos en una misma partición [Granell et al. 2012]. En esta topología, los algoritmos multiresolución basados en modelos de Potts son incapaces de detectar las comunidades pequeñas sin fragmentar las grandes o de detectar las grandes correctamente sin fusionar las pequeñas [Lancichinetti y Fortunato 2011, Xiang y Hu 2012]. Por el contrario, *Surprise* recupera en todo momento la partición esperada.

En resumen, el conjunto de análisis realizados, la ausencia de problemas en redes donde otras medidas fracasan y el extremadamente bajo error cometido en todos los casos analizados, apuntan a que *Surprise* podría ser una medida óptima para evaluar la calidad de la estructura de comunidades de una red.

Otra conclusión importante de este trabajo es la evidencia de que el uso de *Modularity* para evaluar la calidad de una partición es erróneo. Aunque se sabía que la medida tenía ciertos defectos [Fortunato y Barthelemy 2007, Lancichinetti y Fortunato 2011, Xiang y Hu 2012], hasta ahora no se había demostrado tan claramente que sus resultados no correlacionan con la calidad real de las comunidades de una red [Aldecoa y Marín 2013]. Este hecho es altamente importante, ya que existen cientos de artículos, en numerosos campos científicos, donde *Modularity* ha sido utilizada para resolver problemas reales (e.g., [Lusseau et al. 2006, Henderson y Robinson 2011, Alexander-Bloch et al. 2012, Doron et al. 2012, Albert et al. 2013] y muchos otros) y cuyos resultados con gran probabilidad son erróneos.

Comparación de algoritmos

Clustering jerárquico de sus soluciones

En el trabajo que se encuentra en revisión [Aldecoa y Marín 0000], proponemos un nuevo tipo de análisis para caracterizar y clasificar algoritmos de detección de comunidades. La idea consiste en comparar las soluciones de los algoritmos para una red determinada. Realizando un *clustering* jerárquico de todas estas soluciones podemos observar las diferencias relativas entre el comportamiento de los distintos métodos. Aunque este tipo de análisis se puede aplicar a cualquier red individual, en dicho trabajo lo aplicamos a *closed benchmarks*. De este modo, además de observar las diferencias entre algoritmos, podemos ver cómo estas relaciones evolucionan conforme avanza el barajeo de las conexiones de la estructura inicial a la final. Esta novedosa estrategia permite una caracterización más profunda de los métodos de detección de comunidades y se plantea como una dura prueba a la que poder someter cualquier nuevo algoritmo que se genere en un futuro.

Estado de la cuestión

Como se ha comentado anteriormente, durante toda la tesis doctoral, hemos realizado multitud de análisis con numerosos algoritmos de detección de comunidades, los cuales quedan reflejados a lo largo de los diferentes artículos. En nuestro trabajo más reciente, mostramos cómo los *closed benchmarks* ayudan a mostrar la gran diferencia existente entre los comportamientos de esos métodos [Aldecoa y Marín 0000]. Este hecho ya se podía intuir en nuestro artículo anterior, donde se observan importantes cambios en los resultados dependiendo de la estrategia utilizada [Aldecoa y Marín 2013]. Ningún algoritmo es capaz de conseguir los mejores resultados en todos los *benchmarks*, lo cual es una conclusión más importante de lo que podría parecer en un principio. Muchos de estos algoritmos que, hasta ahora eran considerados “buenos”, han sido aplicados a redes reales buscando extraer información de sus distintas comunidades (refs. [Meunier et al. 2009, Sethi et al. 2009, Lewis et al. 2010, Song et al. 2011, Kenah et al. 2011] entre otras). Sin embargo, como queda demostrado en nuestros análisis en *benchmarks open* y *closed*, no podemos confiar plenamente en sólo uno de ellos, dado que su comportamiento depende de la topología de la red analizada. Por tanto, es posible que las conclusiones de algunos de estos estudios no sean del todo acertadas o incluso sean erróneas. Es importante también resaltar que, como cabía esperar, todos los algoritmos basados en *Modularity* se encuentran siempre clasificados entre los que peor comportamiento tienen [Aldecoa y Marín 2013, Aldecoa y Marín 0000]. Este hecho, junto con la demostración de que Q no evalúa correctamente la estructura de comunidades de una red, debería finalmente hacer desisitir del uso de Q en nuevas publicaciones.

Por tanto, como demuestran nuestros dos últimos artículos [Aldecoa y Marín 2013, Aldecoa y Marín 0000], la mejor estrategia para detectar la estructura de comunidades de una red actualmente es la maximización de *Surprise* mediante la combinación de algoritmos. Es decir, ejecutar el máximo número posible de algoritmos sobre una red y, de entre sus soluciones, extraer aquella de mayor S. Esto se debe a que no existe todavía ningún método diseñado específicamente para maximizar *Surprise*. Obviamente, la generación de algoritmos en esta dirección puede mejorar cualitativamente el campo de la detección de comunidades en redes complejas.

Un nuevo paradigma en la detección de comunidades

Surprise no es simplemente una medida más para la evaluación de comunidades en redes, sino que representa una aproximación distinta al problema, un nuevo paradigma en el campo de la estructura de comunidades. En nuestra opinión, la estructura de comunidades es una propiedad *sistémica* de las redes. Es decir, las comunidades no existen por sí mismas, sino que emergen como grupos de nodos significativamente estadísticamente conectados dada la configuración y el patrón de conexiones de la red entera. Esta afirmación choca de frente con una parte importante de los estudios realizados hasta el momento y, sobre todo, con las medidas de evaluación de comunidades presentes en la literatura.

Existe una diferencia importante, a nivel conceptual, entre *Surprise (S)* y otros tipos de medidas propuestas hasta el momento, como *Modularity (Q)* y similares (e.g., evaluaciones basadas en *modelos de Potts*). Cada uno de los términos del sumatorio en la fórmula original de Q (fórmula 1.1 de la introducción), representa la calidad de una comunidad. Es decir, las comunidades se evalúan localmente y luego se suman sus calidades para obtener el valor Q. Por tanto, aunque Q aparece como un índice global de la partición, en realidad evalúa de forma local cada comunidad. Por el contrario, *Surprise* evalúa la calidad de la partición de forma sistémica, donde la significancia de una comunidad no depende sólo de ella si no también de como están distribuidos el resto de las conexiones de la red. S calcula la improbabilidad de la partición en su conjunto, teniendo en cuenta al mismo tiempo la distribución de nodos y conexiones de la red entera.

Aunque los resultados mostrados en todos los análisis realizados hasta la fecha nos indican que *Surprise* muestra un comportamiento excelente, es posible que no sea la medida "*final*", la solución al problema de la detección de comunidades. Sin embargo, consideramos que este tipo de aproximaciones sistémicas son el camino a explorar y la dirección a seguir para una correcta identificación de la estructura de comunidades de una red.

Capítulo 4

CONCLUSIONES

1. Jerarca, una herramienta de gran interés desarrollada para cualquier tipo de usuario, permite extraer información de la estructura jerárquica y de comunidades de una red.

2. UVCluster y en especial SCluster, dos de los algoritmos incluidos en Jerarca, han demostrado un excelente comportamiento a la hora de determinar la estructura de comunidades de una red. En particular, SCluster ha mostrado el mejor comportamiento de entre todos los algoritmos testados en *benchmarks* del tipo *Relaxed Caveman.*

3. El uso de *closed benchmarks* para evaluar algoritmos de detección de comunidades presenta importantes ventajas teóricas, ha demostrado ser una prueba más dura y reporta más información que los resultados obtenidos mediante *open benchmarks.*

4. Queda demostrado que la estructura de comunidades de una red puede ser evaluada utilizando una medida basada en una distribución hipergeométrica, a la que llamamos *Surprise. Surprise* muestra un comportamiento excelente en todos los casos analizados y sus resultados son cualitativamente superiores a los obtenidos por *Modularity.*

5. La evaluación de un gran número de algoritmos de detección de comunidades tanto en *open* como en *closed benchmarks* muestra que ninguno de ellos es capaz de obtener buenas soluciones en todos los casos. Sin embargo, la estrategia de maximizar *Surprise* (basada en elegir para cada red el algoritmo que consigue un valor más alto de *Surprise*) obtiene siempre soluciones óptimas o cuasi-óptimas.

6. El nulo o extremadamente bajo error producido al detectar comunidades maximizando *Surprise* sugiere que esta estrategia podría estar muy cerca

de solucionar el problema de la detección de la estructura de comunidades de una red.

Bibliografía

[Abe y Suzuki 2012] Abe, S. y Suzuki, N. (2012). Dynamical evolution of the community structure of complex earthquake network. *EPL (Europhysics Letters)*, 99:39001.

[Albert et al. 2013] Albert, E. M., Fortuna, M. A., Godoy, J. A., y Bascompte, J. (2013). Assessing the robustness of the networks of spatial genetic variation. *Ecology letters.*

[Albert y Barabási 2002] Albert, R. y Barabási, A. (2002). Statistical mechanics of complex networks. *Reviews of modern physics*, 74:47.

[Aldecoa y Marín 0000] Aldecoa, R. y Marín, I. (0000). Exploring the limits of community detection strategies in complex networks. *Enviado.*

[Aldecoa y Marín 2010] Aldecoa, R. y Marín, I. (2010). Jerarca: Efficient analysis of complex networks using hierarchical clustering. *PloS one*, 5:e11585.

[Aldecoa y Marín 2011] Aldecoa, R. y Marín, I. (2011). Deciphering network community structure by surprise. *PloS one*, 6:e24195.

[Aldecoa y Marín 2012] Aldecoa, R. y Marín, I. (2012). Closed benchmarks for network community structure characterization. *Physical Review E*, 85:026109.

[Aldecoa y Marín 2013] Aldecoa, R. y Marín, I. (2013). Surprise maximization reveals the community structure of complex networks. *Scientific reports*, 3:1060.

[Alexander-Bloch et al. 2012] Alexander-Bloch, A., Lambiotte, R., Roberts, B., Giedd, J., Gogtay, N., y Bullmore, E. (2012). The discovery of population differences in network community structure: New methods and applications to brain functional networks in schizophrenia. *Neuroimage*, 59:3889–3900.

[Arnau et al. 2005] Arnau, V., Mars, S., y Marín, I. (2005). Iterative cluster analysis of protein interaction data. *Bioinformatics*, 21:364–378.

[Barabási y Albert 1999] Barabási, A. y Albert, R. (1999). Emergence of scaling in random networks. *Science*, 286:509–512.

[Barabási y Oltvai 2004] Barabási, A. y Oltvai, Z. (2004). Network biology: understanding the cell's functional organization. *Nature Reviews Genetics*, 5:101–113.

[Blondel et al. 2008] Blondel, V. D., Guillaume, J.-L., Lambiotte, R., y Lefebvre, E. (2008). Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, page P10008.

[Bollobás 1998] Bollobás, B. (1998). *Modern graph theory*, volume 184. Springer Verlag.

[Borge-Holthoefer y Arenas 2010] Borge-Holthoefer, J. y Arenas, A. (2010). Semantic networks: Structure and dynamics. *Entropy*, 12:1264–1302.

[Croft et al. 2005] Croft, D., James, R., Ward, A., Botham, M., Mawdsley, D., y Krause, J. (2005). Assortative interactions and social networks in fish. *Oecologia*, 143:211–219.

[Danon et al. 2005] Danon, L., Díaz-Guilera, A., Duch, J., y Arenas, A. (2005). Comparing community structure identification. *Journal of Statistical Mechanics: Theory and Experiment*, page P09008.

[Doron et al. 2012] Doron, K. W., Bassett, D. S., y Gazzaniga, M. S. (2012). Dynamic network structure of interhemispheric coordination. *Proceedings of the National Academy of Sciences*, 109:18661–18668.

[Duch y Arenas 2005] Duch, J. y Arenas, A. (2005). Community detection in complex networks using extremal optimization. *Physical review E*, 72:027104.

[Erdős y Rényi 1959] Erdős, P. y Rényi, A. (1959). On random graphs i. *Publ Math Debrecen*, 6:290–297.

[Euler 1741] Euler, L. (1741). Solutio problematis ad geometriam situs pertinentis. *Commentarii academiae scientiarum Petropolitanae*, 8:128–140.

[Floyd 1962] Floyd, R. W. (1962). Algorithm 97: shortest path. *Communications of the ACM*, 5:345.

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

[Fortunato y Barthelemy 2007] Fortunato, S. y Barthelemy, M. (2007). Resolution limit in community detection. *Proceedings of the National Academy of Sciences*, 104:36–41.

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

[Granell et al. 2012] Granell, C., Gomez, S., y Arenas, A. (2012). Hierarchical multiresolution method to overcome the resolution limit in complex networks. *International Journal of Bifurcation and Chaos*, 22.

[Henderson y Robinson 2011] Henderson, J. y Robinson, P. (2011). Geometric effects on complex network structure in the cortex. *Physical Review Letters*, 107:018102.

[Kenah et al. 2011] Kenah, E., Chao, D. L., Matrajt, L., Halloran, M. E., y Longini, I. M. (2011). The global transmission and control of influenza. *PloS one*, 6:e19515.

[Kumpula et al. 2007] Kumpula, J., Saramäki, J., Kaski, K., y Kertesz, J. (2007). Limited resolution in complex network community detection with potts model approach. *The European Physical Journal B*, 56:41–45.

[Lancichinetti y Fortunato 2009] Lancichinetti, A. y Fortunato, S. (2009). Community detection algorithms: a comparative analysis. *Physical Review E*, 80:056117.

[Lancichinetti y Fortunato 2011] Lancichinetti, A. y Fortunato, S. (2011). Limits of modularity maximization in community detection. *Physical Review E*, 84:066122.

[Lancichinetti et al. 2008] Lancichinetti, A., Fortunato, S., y Radicchi, F. (2008). Benchmark graphs for testing community detection algorithms. *Physical Review E*, 78:046110.

[Lewis et al. 2010] Lewis, A. C., Jones, N. S., Porter, M. A., y Deane, C. M. (2010). The function of communities in protein interaction networks at multiple scales. *BMC systems biology*, 4:100.

[Lucas et al. 2006] Lucas, J. I., Arnau, V., y Marín, I. (2006). Comparative genomics and protein domain graph analyses link ubiquitination and rna metabolism. *Journal of molecular biology*, 357:9–17.

[Lusseau et al. 2006] Lusseau, D., Wilson, B., Hammond, P. S., Grellier, K., Durban, J. W., Parsons, K. M., Barton, T. R., y Thompson, P. M. (2006). Quantifying the influence of sociality on population structure in bottlenose dolphins. *Journal of Animal Ecology*, 75:14–24.

[Magwene et al. 2004] Magwene, P. M., Kim, J., et al. (2004). Estimating genomic coexpression networks using first-order conditional independence. *Genome Biology*, 5(12):R100.

[Marco y Marín 2007] Marco, A. y Marín, I. (2007). A general strategy to determine the congruence between a hierarchical and a non-hierarchical classification. *BMC bioinformatics*, 8:442.

[Marco y Marín 2009] Marco, A. y Marín, I. (2009). Interactome and gene ontology provide congruent yet subtly different views of a eukaryotic cell. *BMC systems biology*, 3:69.

[Meilă 2007] Meilă, M. (2007). Comparing clusterings—an information based distance. *Journal of Multivariate Analysis*, 98:873–895.

[Meunier et al. 2009] Meunier, D., Lambiotte, R., Fornito, A., Ersche, K. D., y Bullmore, E. T. (2009). Hierarchical modularity in human brain functional networks. *Frontiers in neuroinformatics*, 3.

[Newman 2005] Newman, M. E. J. (2005). Power laws, pareto distributions and zipf's law. *Contemporary physics*, 46:323–351.

[Newman y Girvan 2004] Newman, M. E. J. y Girvan, M. (2004). Finding and evaluating community structure in networks. *Physical review E*, 69:026113.

[Orman y Labatut 2009] Orman, G. y Labatut, V. (2009). A comparison of community detection algorithms on artificial networks. In *Discovery Science*, pages 242–256. Springer.

[Pons y Latapy 2005] Pons, P. y Latapy, M. (2005). Computing communities in large networks using random walks. *Computer and Information Sciences-ISCIS 2005*, pages 284–293.

[Radicchi et al. 2004] Radicchi, F., Castellano, C., Cecconi, F., Loreto, V., y Parisi, D. (2004). Defining and identifying communities in networks. *Proceedings of the National Academy of Sciences of the United States of America*, 101:2658–2663.

[Ravasz y Barabási 2003] Ravasz, E. y Barabási, A. (2003). Hierarchical organization in complex networks. *Physical Review E*, 67:026112.

[Reichardt y Bornholdt 2006] Reichardt, J. y Bornholdt, S. (2006). Statistical mechanics of community detection. *Physical Review E*, 74:016110.

[Ronhovde y Nussinov 2009] Ronhovde, P. y Nussinov, Z. (2009). Multiresolution community detection for megascale networks by information-based replica correlations. *Physical Review E*, 80:016109.

[Rosvall y Bergstrom 2008] Rosvall, M. y Bergstrom, C. T. (2008). Maps of random walks on complex networks reveal community structure. *Proceedings of the National Academy of Sciences*, 105:1118–1123.

[Saitou y Nei 1987] Saitou, N. y Nei, M. (1987). The neighbor-joining method: a new method for reconstructing phylogenetic trees. *Molecular biology and evolution*, 4:406–425.

[Sethi et al. 2009] Sethi, A., Eargle, J., Black, A. A., y Luthey-Schulten, Z. (2009). Dynamical networks in trna: protein complexes. *Proceedings of the National Academy of Sciences*, 106:6620–6625.

[Shen y Cheng 2010] Shen, H.-W. y Cheng, X.-Q. (2010). Spectral methods for the detection of network community structure: a comparative analysis. *Journal of Statistical Mechanics: Theory and Experiment*, page P10020.

[Shi et al. 2009] Shi, C., Wang, Y., Wu, B., y Zhong, C. (2009). A new genetic algorithm for community detection. *Complex Sciences*, pages 1298–1309.

[Smoot et al. 2011] Smoot, M. E., Ono, K., Ruscheinski, J., Wang, P.-L., y Ideker, T. (2011). Cytoscape 2.8: new features for data integration and network visualization. *Bioinformatics*, 27:431–432.

[Sokal y Michener 1958] Sokal, R. R. y Michener, C. (1958). A statistical method for evaluating systematic relationships. *Univ Kans Sci Bull*, 38:1409–1438.

[Song et al. 2011] Song, D.-M., Tumminello, M., Zhou, W.-X., y Mantegna, R. N. (2011). Evolution of worldwide stock markets, correlation structure, and correlation-based graphs. *Physical Review E*, 84:026108.

[Spirin y Mirny 2003] Spirin, V. y Mirny, L. (2003). Protein complexes and functional modules in molecular networks. *Proceedings of the National Academy of Sciences*, 100:12123–12128.

[Strogatz 2001] Strogatz, S. (2001). Exploring complex networks. *Nature*, 410:268–276.

[Tamura et al. 2007] Tamura, K., Dudley, J., Nei, M., y Kumar, S. (2007). Mega4: molecular evolutionary genetics analysis (mega) software version 4.0. *Molecular biology and evolution*, 24:1596–1599.

[Traag et al. 2011] Traag, V., Van Dooren, P., y Nesterov, Y. (2011). Narrow scope for resolution-limit-free community detection. *Physical Review E*, 84:016114.

[Traud et al. 2011] Traud, A., Mucha, P., y Porter, M. (2011). Social structure of facebook networks. *Physica A: Statistical Mechanics and its Applications.*

[Wasserman y Faust 1994] Wasserman, S. y Faust, K. (1994). *Social network analysis: Methods and applications*, volume 8. Cambridge university press.

[Watts y Strogatz 1998] Watts, D. y Strogatz, S. (1998). Collective dynamics of'small-world'networks. *Nature*, 393:440–442.

[Watts 2003] Watts, D. J. (2003). *Small worlds: the dynamics of networks between order and randomness.* Princeton university press.

[Weinan et al. 2008] Weinan, E., Li, T., Vanden-Eijnden, E., et al. (2008). Optimal partition and effective dynamics of complex networks. *Proceedings of the National Academy of Sciences*, 105:7907–7912.

[Xiang y Hu 2012] Xiang, J. y Hu, K. (2012). Limitation of multi-resolution methods in community detection. *Physica A: Statistical Mechanics and its Applications.*

[Zachary 1977] Zachary, W. W. (1977). An information flow model for conflict and fission in small groups. *Journal of anthropological research*, 33:452–473.