



UNIVERSITATEA “BABEȘ-BOLYAI” CLUJ NAPOCA

FACULTATEA DE FIZICĂ

SPECIALIZAREA FIZICĂ COMPUTAȚIONALĂ

LUCRARE DE DISERTAȚIE

COORDONATOR ȘIINȚIFIC:

DR. ERCSEY-RAVASZ MÁRIA

ABSOLVENT:

DERITEI DÁVID

IUNIE 2014



UNIVERSITATEA “BABEȘ-BOLYAI” CLUJ NAPOCA

FACULTATEA DE FIZICĂ

SPECIALIZAREA FIZICĂ COMPUTAȚIONALĂ

LUCRARE DE DISERTAȚIE

DETECTAREA COMUNITĂȚILOR ÎN REȚELE PRIN DIAGrame VORONOI

COORDONATOR ȘIINȚIFIC:

DR. ERCSEY-RAVASZ MÁRIA

ABSOLVENT:

DERITEI DÁVID

IUNIE 2014



BABEȘ-BOLYAI UNIVERSITY, CLUJ-NAPOCA

FACULTY OF PHYSICS

COMPUTATIONAL PHYSICS SPECIALIZATION

DISSERTATION

COMMUNITY DETECTION BY GRAPH VORONOI DIAGRAMS

SCIENTIFIC ADVISER:

DR. ERCSEY-RAVASZ MÁRIA

AUTHOR:

DERITEI DÁVID

JUNE 2014

Abstract

Accurate and efficient community detection in networks is a key challenge for complex network theory and has applications in many different fields such as social sciences, information theory, biology, neuroscience etc. The problem is analogous to cluster analysis in data mining, a field rich in metric space-based methods. Common to these methods is a geometric, distance-based definition of clusters or communities. Voronoi diagrams are a common way of partitioning metric spaces around a set of given seeds. Here we generalised this method to graphs and developed a new geometric approach to graph community detection based on graph Voronoi diagrams. Our method serves as proof of principle that definition of appropriate distance metrics on graphs can bring a rich set of metric space-based clustering methods to network science. We employed a simple edge metric - the edge clustering coefficient - that reflects the intra- or inter-community character of edges, and a graph-density based rule to identify seed nodes of Voronoi cells. We tested our algorithm on benchmark as well as real-world networks and compared it with several widely used algorithms. We have shown that it outperforms most network community detection methods applicable to large networks. In addition to offering a computationally efficient alternative for community detection, our method opens new avenues for adapting a wide range of data mining algorithms to complex networks.

Contents

1	Clustering algorithms - an overview	6
1.1	Community detection methods used for reference	9
2	Benchmark networks and real data sets	11
2.1	Benchmark networks	11
2.2	Real data sets	13
3	Community detection by graph Voronoi diagrams	14
3.1	Graph Voronoi diagrams	15
3.2	Illustration in 2D euclidian space	16
3.3	Distance measure	17
3.4	Identification of generator points	19
4	Results	23
4.1	Benchmark testing	23
4.2	Real world networks.	25
4.3	Computational efficiency	28
5	Discussion and outlook	31
	Publications and conferences	35
	Bibliography	36

Introduction

The science of networks and their application to complex systems has become one of the most widely studied interdisciplinary domain of science in the past decades, having applications in social science, information theory, technology, biology, neuroscience, just to name a few [1, 2]. Whenever using a graph or network representation of a system a common challenge is identification of community structure. Communities are defined as groups of nodes which are more densely connected to each other than with the rest of the network. The paradigm-example of networks exhibiting community structure are social networks (also the word *community* originates from the terminology of social systems). For example people tend to form different communities as the family, friend circles, co-worker groups etc., which are more strongly connected within than with the rest of society.

Detecting communities in networks can be important in sociology, market research, or solving communication problems. In biology detecting clusters in the protein-protein interaction networks can give us a deeper understanding of functional groups or different diseases. Clustering problems similar to graph community detection also occur in data mining, pattern recognition, machine learning and statistical data analysis [14, 13]. In spite the fact that we have many qualitative descriptions, no common mathematical definition of communities (or clusters) has been agreed upon [3]. This is illustrated by an entire subclass of methods that work with operational definitions of network community (see below). While a large variety of community definitions and detection algorithms exist (for an extensive review see [3]), combining meaningful mathematical community definitions with computationally efficient methods remains a challenge.

Chapter 1

Clustering algorithms - an overview

A large variety of approaches can be found in the literature about clustering methods and community detection. The main problem these methods is, that there is no exact, commonly accepted mathematical definition of communities. Their purpose is usually to detect the more densely connected regions of a graph (or network), mostly based on the topology of the network. A more advanced aspect of community detection is the identification of overlapping communities (fuzzy clustering) and also the detection of hierarchical or nested communities. We differentiate two terms in case of graph clustering: partitioning and detection of complex communities. In case of partitioning we assign each node to one and only one cluster or community. When detecting complex communities, however, we can assign nodes to more communities with different weights. We can understand these concepts by taking the simple example of a citation network, where the nodes are the published papers, and the links are the citations (two papers are connected if one cites the other). Intuitively the few main clusters will be the different domains of science like physics, biology, chemistry etc. Yet we expect some communities to overlap with each other through the cross-disciplinary sciences like biophysics, biochemistry and so on. We also expect some smaller, nested communities within the big domains, since the different scientific domains have many sub-domains, for example in physics particle physicists surely cite other particle physics related articles more frequently, than plasma physics papers.

Here we introduce the more significant methods and concepts in the area of network clustering.

One class of methods formulate community detection as a global optimization problem. In these methods the quality of a particular graph partitioning is measured by a fitness function (e.g., mod-

ularity, separability), and communities are found by maximizing this quality function [4, 5, 6, 3]. The most commonly used quality function is the *modularity* function defined as:

$$Q = \frac{1}{2m} \sum_{ij} (A_{ij} - P_{ij}) \delta(V_i, V_j) \quad (1.1)$$

where the sum runs over all pairs of vertices, A is the adjacency matrix, m the total number of edges of the graph, and P_{ij} represents the expected number of edges between vertices i and j in the null model, in present case, the random graph model. The δ function yields one if vertices i and j are in the same community ($V_i = V_j$), zero otherwise [4]. An other popular quality measure is the *cluster level separability*, which is simply the ratio between the internal and external number of edges of a cluster:

$$g = \frac{m}{c + m} \quad (1.2)$$

where m and c are the internal and external number of edges, respectively. The network level separability of a partitioning is the cluster size weighted average of the cluster level separabilities [31]. Nevertheless, finding the optimum partition by maximizing these fitness functions is an NP-hard problem, thus extremely inefficient on large networks. This difficulty lead to development of approximations based on local optimization [7].

A second class of algorithms are characterized by operational definitions of community structure. They prescribe a procedure or dynamics that partitions the graph, and the partitioning is accepted as its community structure. For example, divisive/agglomerative algorithms consist of an iterative process of link removal/inclusion based on some link property (e.g., betweenness centrality [8] or similarity in link-community detection [9]), resulting in a dendrogram (Figure 1.1).

The betweenness centrality based divisive algorithm introduced by Newman et. al. [5] became very popular since it is very intuitive and easy to implement: we assign a number to each node (or edge) called the betweenness centrality, which is defined as the number of shortest paths going through the given node (or edge) from all the possible shortest paths. Intuitively those elements of a network, which have the most shortest paths going through them are most likely connecting clusters. Through an iterative process we always remove the elements with the highest betweenness scores, and then recalculate the betweenness values, until we reduce the network to elementary components, in the meantime constructing the dendrogram. A similar method was introduced by

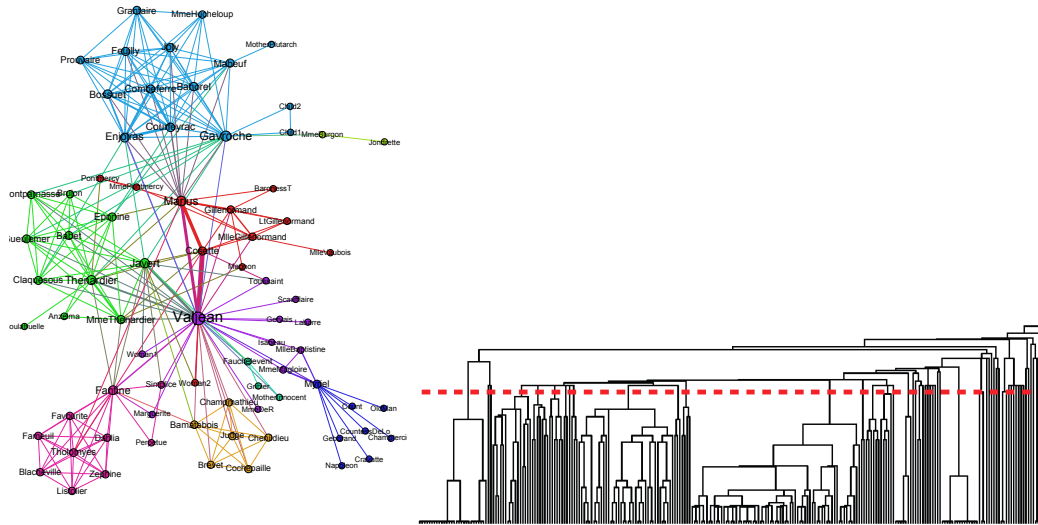


Figure 1.1: Left: The network of the characters of the Les Misérables novel: two characters are connected if they appear in the same scene. Right: The dendrogram of the Les Misérables network based on the similarity in link-community detection. Cutting the dendrogram on different heights can reveal different hierarchical levels, but finding the optimal cutting level is also a challenge.[?].

Radicchi [10], but instead of the betweenness centrality they used the so called edge clustering coefficient (definition in Section 3.3). The key advantage of the ECC is that it is a local measure, and does not require the knowledge of all the shortest paths, resulting in a more efficient algorithm. A similar measure frequently used is the similarity measure [9]. While these methods are typically more efficient than optimization-based clustering, it is usually not obvious where the resulting dendrogram needs to be cut to obtain meaningful partitioning. This problem was mitigated by the formulation of a validity check for communities, making the cutting of dendrograms more reliable [10].

Dynamic processes on complex networks inspired a set of algorithms based on stochastic processes such as random walks [11] and label propagation [12]. In this case, besides choosing between the large spectrum of proposed dynamics, one must also deal with high sensitivity to parameters [3]. In summary, the strong dependence of the efficiency and accuracy of these methods on the particularities of the analyzed network have so far prevented the emergence of a single canonized community detection procedure.

Graph community detection is in many ways analogous to cluster analysis in data mining, pattern recognition and statistical data analysis, with the important difference that these problems

are defined on continuous metric spaces. Voronoi diagrams [15] are a common way of partitioning metric spaces into regions (Voronoi cells) around a given set of points called seeds (or generator points), such that each point of the space belongs to the cell of the closest seed. For example k -means clustering, a well-known NP-hard problem in data mining [16], partitions a set of data points (observations) into k Voronoi cells. To do this, the method identifies a set of seeds for which the average square distance (e.g., Euclidean distance) between data points and the Voronoi seeds they belong to is minimal.

Although several attempts to map graphs to metric spaces have been reported, they did not yield a general, computationally efficient network community detection algorithm. For example, Nishikawa *et al.* defined a multidimensional metric space of arbitrarily chosen node properties, such as the average degree of first neighbors or the clustering coefficient [17]. They then projected this space onto two dimensions and inspected them visually to discover group structures. It is not clear whether this method identifies structural network communities, or node groups that occupy structurally similar positions in the network. Jin *et al.* used density-based clustering techniques to define a distance function between nodes [18]. However, their distance calculation is a multistep procedure involving global optimization. Combined with a sophisticated clustering algorithm, the procedure is of high order time-complexity $O(n^\alpha)$, $\alpha > 2$.

A straightforward alternative of mapping graphs onto metric spaces (although not continuous) states that all edges have a positive length and the distance between two nodes is the length of the corresponding shortest path [19]. Using this approach our goal has been to generalise partitioning methods used in metric spaces to networks.

1.1 Community detection methods used for reference

To verify and test our community detection method, besides benchmark testing, we had to compare our results with results provided by other, already accepted and used methods. In this section we shortly present five popular algorithms used as reference in our research. We chose efficient algorithms which can also be applied on large networks.

The *Louvain* method is a community detection algorithm based on local modularity optimization [7]. The method consists of two main phases. Initially all nodes are put in different communities.

In the first phase nodes are considered in random order and the modularity gain is calculated if a given node is removed from its community and is added to one of its neighbor's community. The gain is calculated for all neighbors, and if there is at least one positive gain, the node is placed in the community, where the gain is maximum. If no more modularity gain can be achieved, the algorithm enters its second phase: communities found in the first phase are replaced by a single node, and one can return to phase one. The two phases can be repeated several times.

GANXiS (SLPA) [32, 33] is a heuristic method capable to detect both overlapping and non-overlapping (crisp) communities. It is an extension of the *Label Propagation Algorithm* [12]. During an iteration step nodes are selected to be listeners in random order. The neighbors of the listener send a label randomly chosen from their label list with a probability proportional to the labels' frequency. The listener selects the most popular label and adds it to its label list, or if it is already present increases its frequency by one. Initially all label lists contain a single label: the label of the node they belong to. After performing a predefined number of iteration steps, label lists are post-processed in order to define communities. The probability distribution of labels present in label lists defines the strength of association to communities to which the node belongs.

The *Infomap* method uses probability flow of random walks as information flow and the network is decomposed into modules by describing the path of this flow [34, 35].

The Link-Community clustering method [9] instead of assuming that a community is a set of nodes with many links between them, this approach considers a community to be a set of closely interrelated links. Moreover, the Link-Community clustering method results a hierarchical clustering. Taking into account the similarity between links a dendrogram is built. Each leaf of the dendrogram represents a link from the network and branches represent link communities. The links stand at unique positions of the dendrogram. Accordingly, the nodes will occupy multiple positions. The link dendrogram represents a hierarchy structure and to obtain community structure it is necessary to realize a horizontal cut on this dendrogram.

Chapter 2

Benchmark networks and real data sets

Testing the performance of a clustering algorithm raises several challenges. In real world data sets we rarely have any reference information about the real community structure, and while there is no rigorous definition of communities, even if we had, it could cause controversy. The only testing procedure consists of comparing the clustering results of an algorithm to other methods, through some quality score, or mutual information. For this reason it became almost necessary to use benchmark networks generated with ground-truth, i.e., predetermined communities. There are methods designed to generate benchmark networks with different properties [28, 29]. The great advantage of benchmark networks is, that we have information about the ground truth community structure, to which we can compare the results of our algorithm. It also allows us to test a huge variety of network types and sizes. As argued by Fortunato et al. in [3], community detection algorithms need to be tested both on benchmark graphs and real-world networks.

2.1 Benchmark networks

The benchmark software framework provides a great variety of different programs, making us able to generate directed, undirected and weighted graphs with disjoint, hierarchical or overlapping communities. In the first phase of our research, we mostly tested our methods on undirected networks, therefore we used a version of the benchmark software, which can generate large scale undirected networks. The software is very flexible and there is a wide range of variable network parameters including: N -the number of nodes, $\langle k \rangle$ -the average degree, k_{max} -the maximum degree,

γ -the degree distribution, β -the community size distribution, μ -the mixing parameter (which sets the rate of edges inside and between clusters), s_{max} -the maximum size of communities, s_{min} - the minimum size of communities. The construction of a realization of the benchmark proceeds through the following steps, according to [29].

1. Each node is given a degree taken from a power law distribution with exponent γ . The extremes of the distribution k_{min} and k_{max} are chosen such that the average degree is k .
2. Each node shares a fraction $1 - \mu$ of its links with the other nodes of its community and a fraction μ with the other nodes of the network; μ is the mixing parameter.
3. The sizes of the communities are taken from a power law distribution with exponent β , such that the sum of all sizes equals the number N of nodes of the graph. The minimal and maximal community sizes s_{min} and s_{max} are chosen so to respect the constraints imposed by our definition of community: $s_{min} > k_{min}$ and $s_{max} > k_{max}$. This ensures that a node of any degree can be included in at least one community.
4. At the beginning, all nodes are homeless, i.e., they are not assigned to any community. In the first iteration, a node is assigned to a randomly chosen community; if the community size exceeds the internal degree of the node i.e., the number of its neighbors inside the community, the node enters the community, otherwise it remains homeless. The procedure stops when there are no more homeless nodes.
5. To enforce the condition on the fraction of internal neighbors expressed by the mixing parameter μ , several re-wiring steps are performed, such that the degrees of all nodes stay the same and only the split between internal and external degree is affected, when needed. In this way the ratio between external and internal degree of each node in its community can be set to the desired share μ with good approximation.

The output files of the benchmark software are an edge list with 'source target' format, a file containing a node list with the community data (node cluster-id format), and a file with statistical information about the network.

The authors of [29] also provided another software, the so-called *mutual information* [30], which

returns the rate of efficiency of a method, if given the community structure computed by the tested method, and the community data generated by the benchmark.

2.2 Real data sets

There are many data sets and networks available in the literature. A plausible way to test an algorithm is to test it on a large variety of networks, but depending on the computational efficiency and other properties of an algorithm we have to choose wisely. We tested our algorithm on several real-world networks with widely different structure and properties, frequently used in the literature.

1. The updated version of the collaboration network of scientists posting preprints on the condensed matter archive at www.arxiv.org (CONDMAT). This is based on preprints posted to the archive between January 1, 1995 and March 31, 2005 [22] and it contains 39576 nodes and 175692 links.
2. Protein interaction network for yeast (YEAST) [23] including 1845 nodes and 4405 edges.
3. Network of links between american political blogs (POLBLOGS) [24] with 1223 nodes and 19087 edges.
4. Neural network of the nematode *C. Elegans* (CEL) [25, 26] contains 297 nodes and 2359 links.
5. The famous network of friendships between the 34 members of the Zachary karate club linked with 78 edges [27].

Chapter 3

Community detection by graph Voronoi diagrams

Voronoi diagrams [15] are a common way of partitioning metric spaces into regions (Voronoi cells) around a given set of points called seeds (or generator points), such that each point of the space belongs to the cell of the closest seed (Fig. 3.1A).

Graphs can be mapped onto metric spaces (although not continuous) if all edges have a positive length and the distance between two nodes is the length of the corresponding shortest path [19]. We expand on this alternative, and define network communities as graph Voronoi cells [20] (Fig. 3.1B), where the distance between two points is the weighted shortest path. We argue that this simple geometrical approach provides reliable results for a broad range of networks at low computational cost.

Our approach raises two major challenges. *First*, we need a meaningful definition for the length of the edges, one that efficiently separates communities in the metric space. A variety of widely used local topology-based similarity/dissimilarity measures meet this criterion [3]. We chose one of the simplest measures, namely the inverse of the edge-clustering coefficient [10]. This measure assures that links connecting different clusters are longer than intra-community links. While more complex definitions are certainly possible, we have found this simple approach to provide very good results. The *second* challenge is to identify Voronoi seed nodes based on topological information. In contrast to k -means clustering, we do not a priori specify number of communities. As Voronoi seeds need to be in the “center” of communities, we propose a density-based approach.

We identify seeds as local maxima in the “density landscape” of the graph. Visually one can imagine that each node has a height proportional to its relative local density [3]. Communities appear as hills in this landscape, while generator nodes are those at the top of the hills (see Fig. 3.2B).

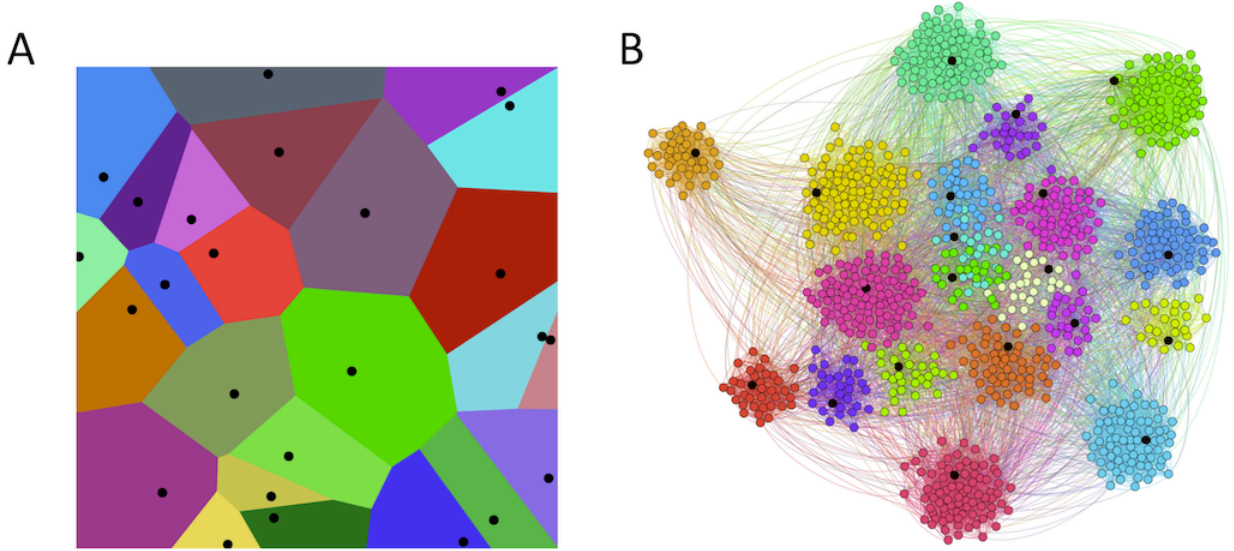


Figure 3.1: **Voronoi diagrams** (A) Illustration of Voronoi partitioning in 2D Euclidean space. (B) Graph Voronoi diagram as represented by the graph drawing application Gephi using the ForceAtlas2 layout algorithm[21]. Generator nodes are shown in black.

3.1 Graph Voronoi diagrams

Let $G = (V, E)$ be a directed and weighted graph with set V of N vertices (nodes) and set E of M edges (links). We denote the length (weight) of an edge connecting nodes ν and μ as $l(\nu, \mu) > 0$. The *length* of a path is obtained by summing up the length of edges constituting the path. The *distance* $d(\nu_i, \nu_j)$ between two nodes ν_i and ν_j is the length of the shortest path between the two nodes. Thus, definition of an edge length assures that the graph can be treated as a metric space. Naturally, the simplest choice of $l(\nu, \mu)$ is 1 if ν and μ are directly connected, 0 otherwise.

Let us now choose a set $S \equiv (\gamma_1, \gamma_2, \dots, \gamma_g) \subset V$ of *generator points* or *seeds*. The *Voronoi diagram* of graph G with respect to S is thus the partitioning of V into node sets $V_1, V_2, \dots, V_g \subset V$, where each point set or *Voronoi cell* is associated with a generator point and

1. The Voronoi cells cover the original graph with no overlaps:

$$\bigcup_{i=1}^g V_i = V, \quad V_i \cap V_j = \emptyset, \quad i, j = 1, 2, \dots, g.$$

2. Nodes that belong to a Voronoi cell are closest to the generator point associated with the cell:

$$d(\nu, \gamma_i) \leq d(\nu, \gamma_j) , \quad \text{for all } \nu \in V_i , \quad i, j = 1, 2, \dots, g .$$

Mathematical properties of graph Voronoi diagrams, along with different methods for their identification and their respective computational complexity are detailed in [15].

To illustrate the potential of the Voronoi partitioning method for community detection, we first applied its crudest version to an extremely modular benchmark network with pre-defined communities (Fig. 3.1B). Here, we selected arbitrarily a generator point from each known community (black nodes), and simply assumed that all edges have unit length. The ForceAtlas2 layout algorithm in the Gephi drawing application[21] aptly coalesces the sharply separable clusters of this network. We then added node color to indicate Voronoi cell membership. The near-perfect mapping between Voronoi cells and the community structure visualized by the layout algorithm suggests a low sensitivity of Voronoi partitioning to the choice of generator nodes and of the distance measure (definition of the length of edges). Next, we show that identification of a meaningful distance measure and optimal generator points leads to excellent results even on more complex graphs.

3.2 Illustration in 2D euclidian space

In order to introduce the logic of our clustering method, first we illustrate its essentials on a clustering problem defined in Euclidean space. Imagine a set of points thrown onto a two dimensional plane, unevenly distributed to form local groups (Fig. 3.2A, black dots). The question is, how can we partition the space to separate these clusters. First, we define a square lattice across the plane, and calculate the local density in each small square as the number of black dots included in it. For example, the space in Fig. 3.2A is partitioned into 12×25 squares and colored based on local density (from green/low to red/high). Second, we choose squares with the largest density inside their neighborhood with radius r as Voronoi cell seeds (generator points). In the top image in Fig. 3.2A, we chose $r = 4$ units, yielding 7 seeds (blue stars; gray circles indicate their local neighborhood). Third, we assign each point of the plane to a Voronoi cell belonging to the seed

closest to the point. This yields a Voronoi partitioning diagram, where partition boundaries are equidistant lines to at least two Voronoi seeds (blue lines). The resulting Voronoi cells partition the set of black dots into groups which correspond well to communities identifiable by the naked eye. Increasing the radius leads to sequential disappearance of seeds, resulting in larger clusters. This process typically merges whole clusters, without splitting them in meaningless ways. For example, increasing the radius from $r = 4$ to $r = 7$ (Fig. 3.2A) merges smaller clusters and partitions the same set of black points into two clusters. This suggests that varying the parameter r may be leveraged to detect structural hierarchy.

Occasionally, two squares closer than r happen to have the same local density. In this case, the Voronoi seed is chosen randomly from the two candidates. This rarely occurs in large systems (especially on graphs, see below). Moreover, the algorithm is not sensitive to the above random choice, as illustrated in Fig. 3.2A, top panel. Here, the squares with coordinates $(8, 7)$ and $(9, 8)$ have the same density. In the figure we chose $(9, 8)$ as the seed, but choosing the other one only changes the community association of two black dots ($(7, 4)$ and $(9, 12)$). Both dots hang at the edge of their cluster in either partitioning result, not being strongly associated with either cluster.

In addition to the radius r , the method applied to this Euclidean toy model depends on the lattice cell size used to calculate local density. On graphs, a more natural definition for local density exists that does not require an initial lattice. We emphasize that the above method is presented for the sole purpose of visually illustrating the elements of our proposed graph community detection method. As clustering points in Euclidean space is a central issue in data mining, efficient methods abound [14, 13]. However, most of these algorithms cannot be directly applied to graphs.

3.3 Distance measure

For optimal use of our geometrical approach, we need a distance measure definition which translates node membership in different communities into separation in metric space. In other words, the more likely an edge connects two different clusters, the longer it should be. There are many distance measures which qualify for this purpose (see discussion). Here we chose to adapt one of the simplest, well accepted and computationally cost-effective measures, namely the Edge Clustering Coefficient (ECC) introduced in [10]. The *ECC* of an edge connecting node i to node j is

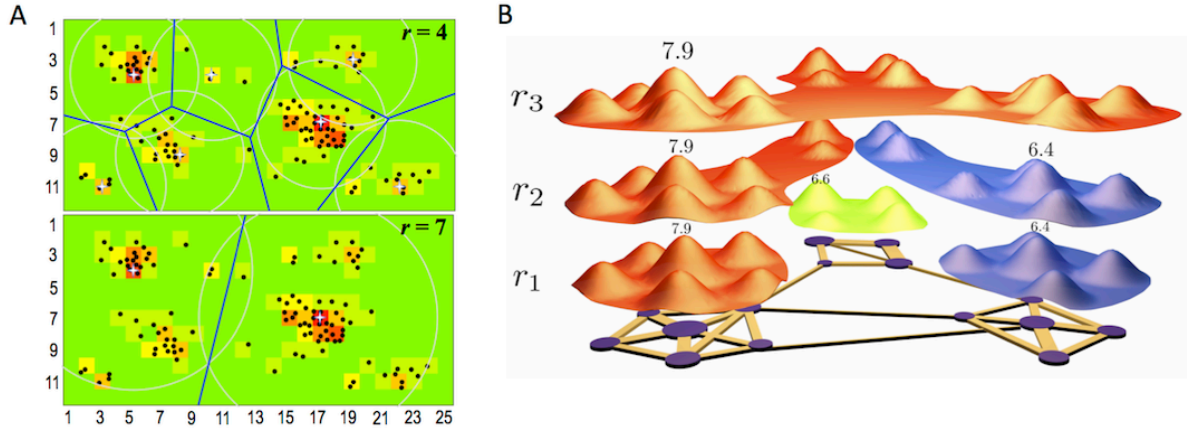


Figure 3.2: **Identifying cell centers.** (A) Illustration in Euclidean space: In order to separate the black dots into clusters we partition the plane into 12×25 squares and measure the local density in each square as the number of dots included. This is indicated by the color on a scale from green (0 density) to red (largest is 6 dots/square). A square becomes a generator point (white star) if it has the largest density in its neighborhood (gray circle) with radii $r = 4$ (top) and $r = 7$ (bottom) (one unit is the edge of one square). The Voronoi partitioning is shown with blue lines. (B) Generalizing the algorithm to graphs each node will have a local density (indicated by the size of nodes and also the height of small hills above them). Length of edges is proportional with the inverse of the ECC (shown by the width of edges). Generator nodes (the value of their local density is shown) have the highest density in their neighborhood with radius $r_1 < r_2 < r_3$. When increasing r clusters are sequentially merged together.

defined as

$$C_{i,j} = \frac{z_{i,j} + 1}{\min[(k_i - 1), (k_j - 1)]} \quad (3.1)$$

where k_i, k_j are the degrees of the two nodes, $z_{i,j}$ is the number of triangles the edge belongs to, and $\min[(k_i - 1), (k_j - 1)]$ is the number of potential triangles it could belong to, as it is the smaller value of the degrees of the two adjacent nodes, minus one (the examined edge) Fig 3.3. This measure characterizes the probability that an edge connects nodes of different clusters (inter-cluster) or nodes in the same cluster (intra-cluster). The smaller the ECC, the more likely an edge is to connect nodes in different clusters. Consequently, we define edge length (weight) as the inverse of ECC in our graph Voronoi clustering method. The length of any path in the graph is then the sum of the length of edges along the path, and the distance between two nodes is the length of the shortest path.

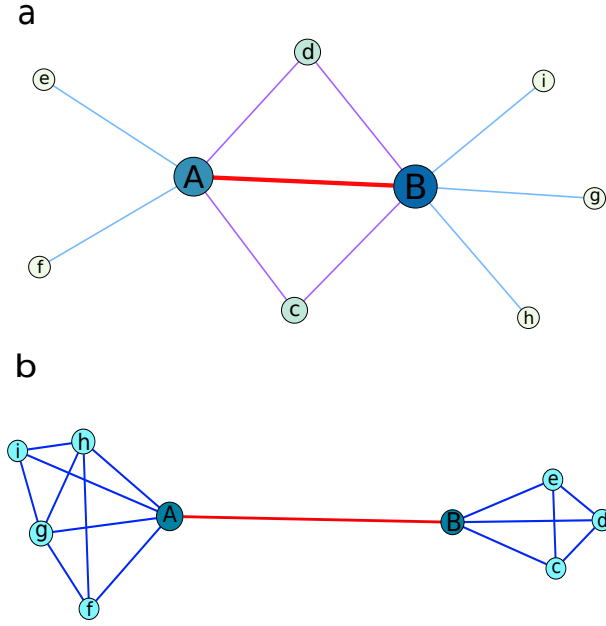


Figure 3.3: **a** The edge clustering coefficient of the red edge (between nodes A and B) is calculated as the number of triangles it belongs to (purple edges with nodes c and d), plus one (to avoid division by zero), divided by the potential number of triangles it could belong to. This is the smaller degree of the two adjacent nodes (this case node A), minus one (the edge examined). By all potential triangles we mean the case if all the neighbors of the A node would be a common neighbor with B, like c and d. So the ECC of the examined red edge is $\frac{2+1}{4} = \frac{3}{4}$. If we calculate the ECC of the highlighted edge on Fig. **b**, we get $\frac{1}{3}$. Even this simple example shows that the edges connecting clusters have a significantly smaller ECC, than the ones within clusters. When using the distance measure as a "length" we need the edges between clusters to be longer, so we use the ECC's inverse.

3.4 Identification of generator points

In addition to an appropriate distance measure, optimal choice of generator nodes can further improve the match between network communities and graph Voronoi cells. A principal goal is to select *one* central node in each cluster. In the rare case when one has a conjecture or a sample of the community structure, random nodes belonging to supposedly different communities can work as generator points (as illustrated in Fig. 3.1B). For the general case, however, no such information or meta-data is available. Here we offer a Voronoi seed selection method based on the relative local density of nodes [3], which is calculated as:

$$\rho_i = \frac{m}{m+k} \quad (3.2)$$

where m is the number of edges **inside** the subgraph consisting of the first neighbors of node i , and k is the number of edges going **out** of the neighborhood. The value of this density is highest for nodes within the center (dense parts) of communities, as illustrated in Fig. 3.2B.

As in Euclidean space, generator points on a graph will be chosen as the nodes with the highest local density in a region within radius r . On graphs r is measured using the above defined distances between nodes (Fig 3.4).

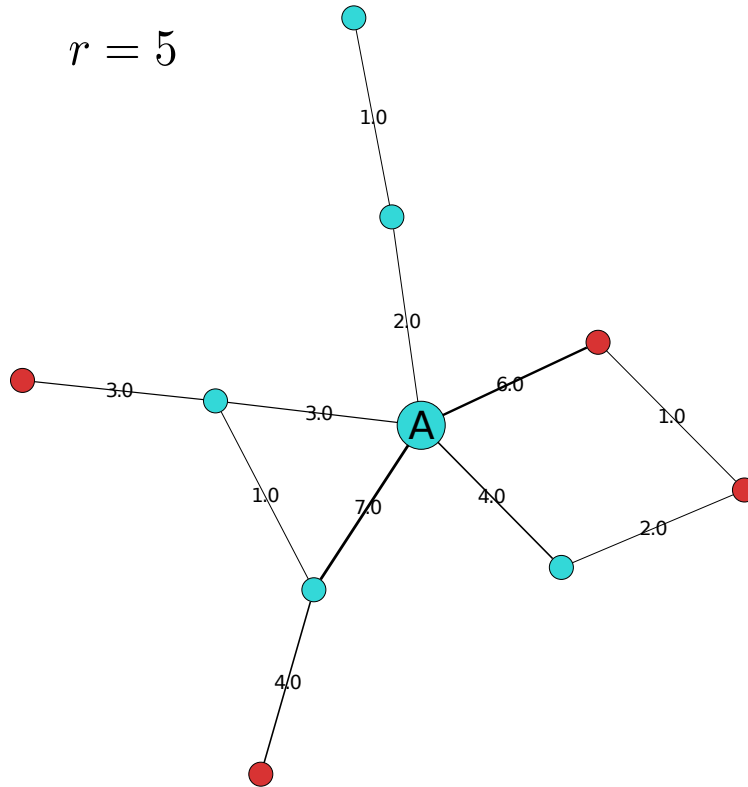


Figure 3.4: We examine the environment of the middle node marked with **A**. We are looking for the nodes, which are within r distance on the weighted graph, the distance being defined as the weighted shortest path. Considering the edge weights the nodes within the $r = 5$ radius are the ones marked with blue. If a node has the highest local density (Eq. 3.2) inside the radius (compared to the blue nodes), then we take it as a generator point.

This method for selecting community centers avoids the selection of multiple generator points within one cluster. In case two nodes have equal density, we choose one randomly. One could ask how frequently does this happen, does it introduce a significant stochastic effect to the method? We measured the relative local density values of all nodes on many graphs and we found that increasing the size of graphs this event becomes less and less frequent. This also holds when increasing the density. Even in those rare cases when it occurs, it affects nodes with small local

density (Fig. 3.5). Candidates for generator nodes are always the nodes with high local density, so the situation of choosing randomly between two local maxima is extremely rare.

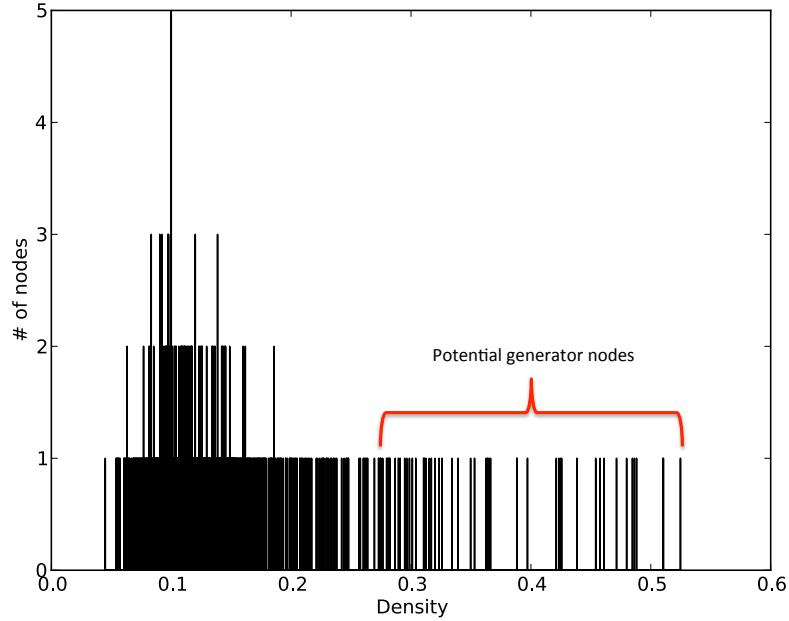


Figure 3.5: Distribution of the relative local density of nodes on a benchmark network with $N = 1000$ nodes, average degree $\langle k \rangle = 15$, mixing $\mu = 0.2$. The number of different density values is 995, meaning there are only 5 values which are not unique. The histogram bin size is extremely small $\simeq 3 \times 10^{-5}$. We can see that the probability of having equal or (almost equal) density values is very small, especially at high densities where potential generator nodes belong.

Varying the parameter r leads to partitioning with different number of clusters and can reveal hierarchical community structure within the graph. As illustrated in Euclidean space, the algorithm is not sensitive to r . Cluster memberships obtained at similar r values are near-perfectly matched (Figs. 4.3 A,B), with the exception of r values where cluster mergers take place in hierarchical networks (Figs. 4.3 C-E). As we will show below in benchmark and real networks, extremely small r values should not be considered, but relatively small r values already yield good partitioning, keeping the algorithm computationally efficient. We propose that the best strategy is to partition the graph with increasing values of r , while monitoring a quality function of the resulting clustering (e.g., modularity, separability). Optimum quality is usually achieved at small values of r , as shown below.

The resulting method is computationally efficient. The complexity of graph Voronoi partitioning is $O(N \log N)$ [20], where N is the number of nodes. Both the edge clustering coefficients and relative local densities can be calculated in $O(M \langle k \rangle^2)$ time, where M is the number of edges and

$\langle k \rangle$ is the average degree. Lastly, finding the Voronoi seeds can be achieved well below $O(Mg)$, where g denotes the number of communities (and thus the number of generator nodes). More on the computational efficiency in Section 4.3 Although we concentrated on partitioning graphs into non-overlapping communities, in Section 5 we offer an outlook of how the method could be generalized to fuzzy clustering.

Chapter 4

Results

4.1 Benchmark testing

We tested our method on benchmark networks with a large variety of parameter settings. Fig. 4.1 shows the results obtained with 12 of these settings. 100 graphs were considered for each case, with $N = 1000$, $k_{max} = 200$, $\gamma = -2$, $\beta = -1$, $s_{min} = 50$, $s_{max} = 500$ and varying average degrees $\langle k \rangle = 6, 10, 15, 20$ and mixing parameters $\mu = 0.1, 0.2, 0.3$. We performed several rounds of Voronoi partitioning on each graph with increasing r , and monitored quality functions that compare our partitioning to the pre-defined communities of the graph. We used the widely accepted *modularity* function [4] (red line) and the less frequent *separability* [31] (blue line). Depending on the concrete application, other metrics may be used. As the clusters in benchmark networks are predefined, we also computed the mutual information between our clustering and the original (ground-truth, GT) cluster membership [30] (green line). The horizontal line on Fig. 4.1 shows the modularity of the original clustering of the benchmark graph (ground truth modularity). Our method reaches this maximum level of modularity as the local radius increases. At this optimal value of r , the mutual information is typically close to one, indicating that our method identifies the original clusters almost perfectly. Separability also increases with r , reaching a plateau at the same r as modularity. As this quality function saturates at large r (occasionally showing a sudden increase), its usefulness as an indicator of optimal clustering is questionable. This phenomena follows from the definition of separability, which naturally increases at small cluster numbers (it reaches its maximal value 1 at a single cluster).

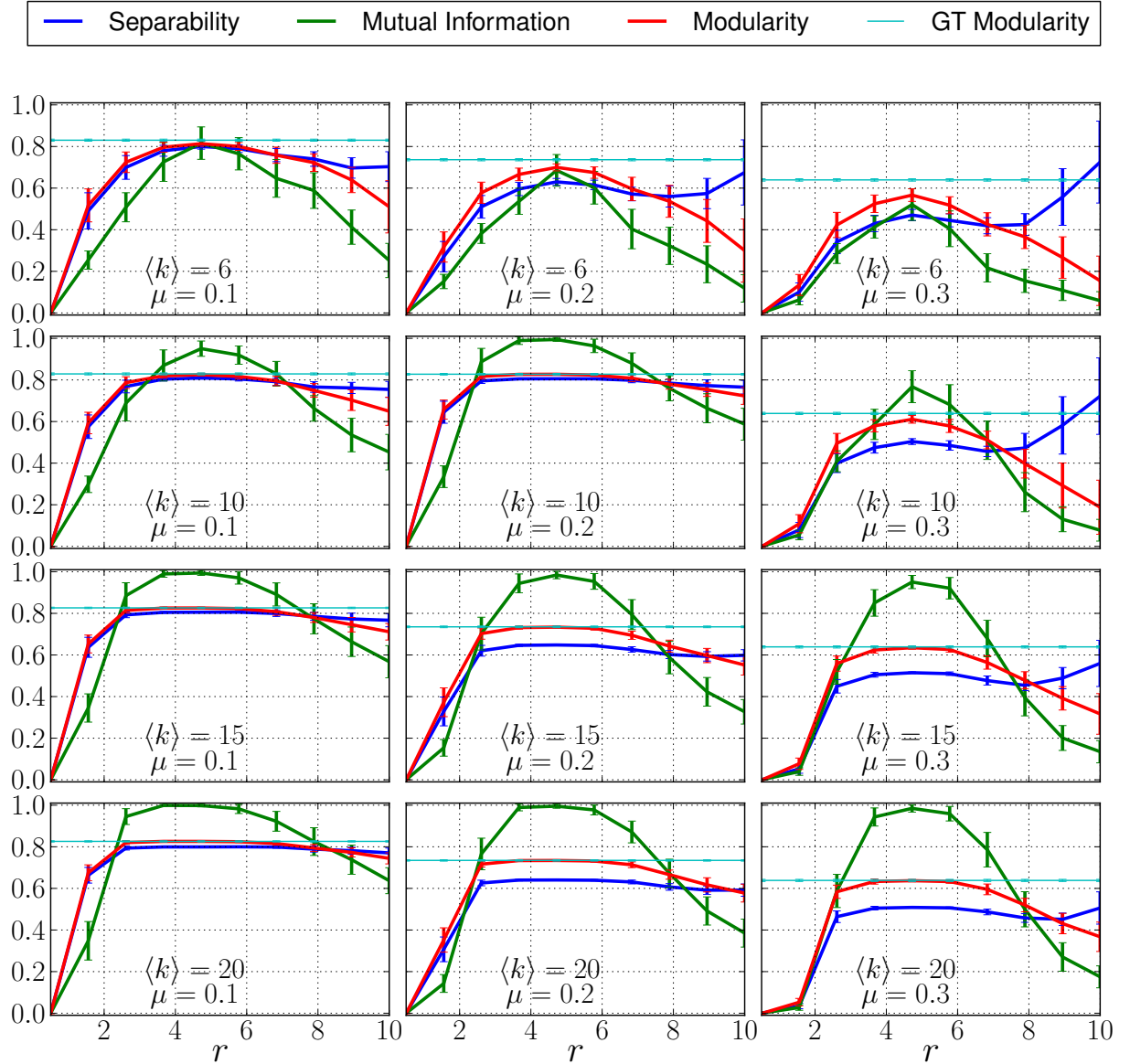


Figure 4.1: **Testing on benchmark networks.** The value of modularity (red), separability (blue) and mutual information of the Voronoi and ground-truth clustering (green) is shown as function of the radius parameter r . The ground-truth modularity is shown with turquoise line. Statistics was performed on 100 graphs at each different parameter setting with $\langle k \rangle = 6, 10, 15, 20$ and $\mu = 0.1, 0.2, 0.3$. Error bars indicate standard deviation.

Our algorithm achieves modularity values that are very close to their ground-truth value, even in very sparse graphs with large mixing ($\langle k \rangle = 6, \mu = 0.3$). Community structure in these graphs is not very clear (indicated by the smaller value of the GT modularity), there is no good ground-truth to detect. Indeed, on these graphs the mutual information between our partitioning and the pre-defined cluster membership is small, indicating that the GT-level modularity can be achieved even with clusters that are very different from the original benchmark communities.

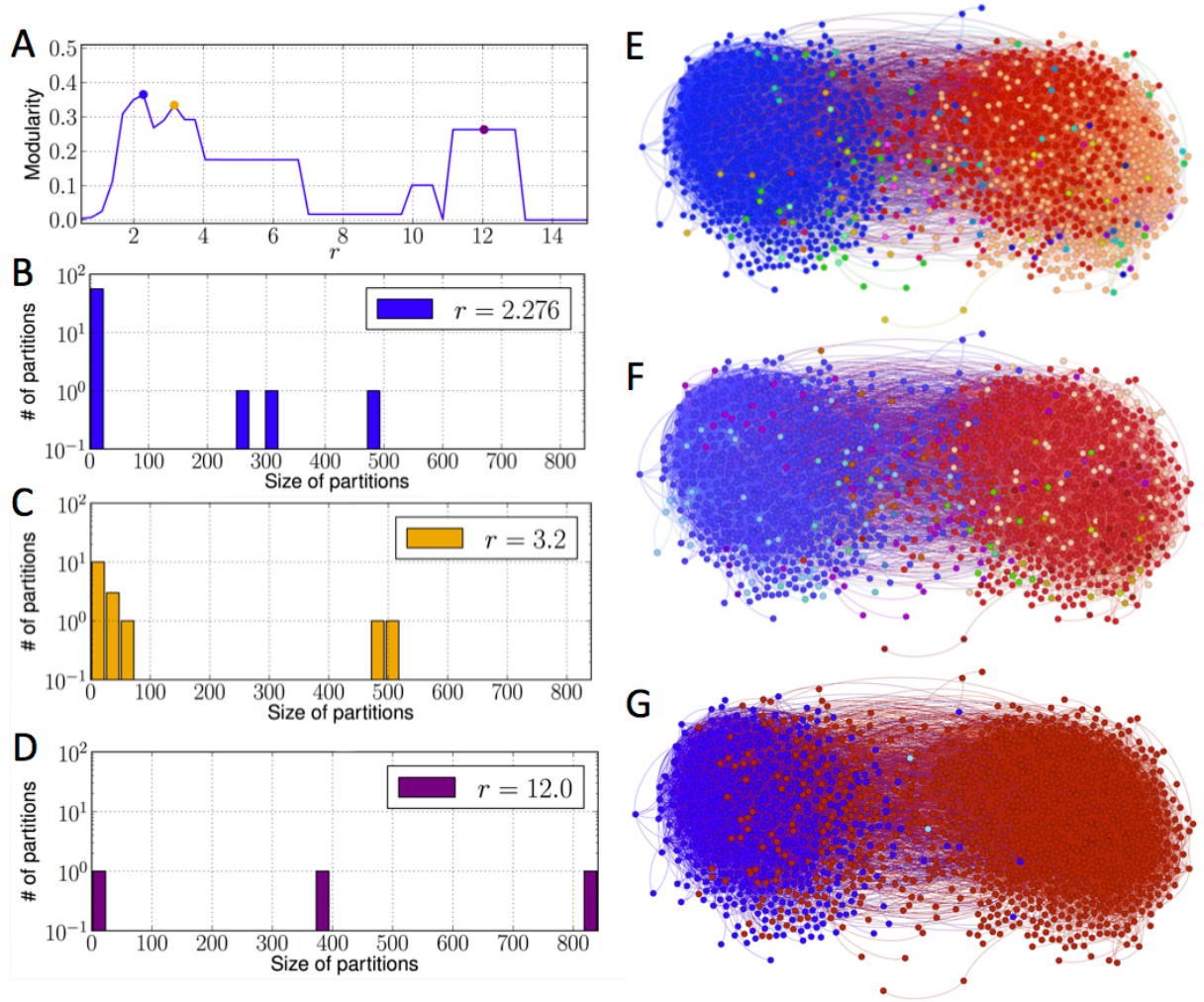


Figure 4.2: **Testing on real-world networks.** (A) Modularity of the Voronoi partitioning as function of r in the POLBLOGS network. At the three different values indicated by colored dots the size distribution of partitions is shown in (B) for $r = 2.276$, (C) $r = 3.2$, (D) $r = 12.0$. (E,F) The Gephi graph drawing software represents the network using the ForceAtlas2 layout algorithm [21]. Colors of nodes indicate the clusters detected with our algorithm at (E) $r = 2.276$, (F) $r = 3.2$ and (G) $r = 12.0$.

4.2 Real world networks.

Due to the method used to generate benchmark networks, they have well-defined structures with clearly separated communities (especially if the mixing parameter is relatively small). The structure of real-world networks is less obvious; detecting communities poses a greater challenge. Moreover, the performance of algorithms is harder to test, as the "ground-truth" community structure is not known. Consequently, new techniques are tested by comparing them to already accepted methods. Here we compare the clustering results of different algorithms through goodness (quality) metrics. Specifically, we have chosen *modularity*, proven to be one of the most reliable quality

indicators of community detection (defined in Section 1.1).

We tested our algorithm on several real-world networks frequently used in the literature, widely different in structure and origin, discussed in Section 2.2

First, we followed the evolution of the modularity function with increasing radius r on each of the five networks. Fig. 4.2A shows modularity of the POLBLOGS (see Fig. 4.3 for the other networks). Modularity reaches a maximum or a plateau at very small $r \in [2, 4]$ for each of the five networks. In some cases, such as the *C. elegans* neural network, modularity reaches a clear maximum at an optimal r , then starts to decrease (Fig. 4.3). In other cases, such as CONDMAT and YEAST, modularity converges to a maximum at low r , then stays at this plateau. The most intriguing behavior is seen in POLBLOGS (Fig. 4.2A), where modularity has several local maxima. Three distinct peaks appear at very different r values ($r = 2.276, 3.2, 12.0$), pointing to a hierarchical structure. Figs. 4.2B-D show the size distribution of partitions obtained at these r values (indicated by colored dots in Fig. 4.2A). In american politics we expect two large, crisp communities corresponding to the democratic (left) and republican (right) wings. Indeed the force-layout algorithm separates the graph in two clusters with almost equal size (Fig. 4.2E,F,G). Our algorithm agrees, but it discovers additional detail. For example, at $r = 2.276$ we have 3 large communities. One of these is almost twice as large as either of the other two, and it corresponds to the democratic left wing (blue cluster in Fig. 4.2E). The right wing, on the other hand, is cut into two distinct clusters (orange and red). At $r = 3.2$, our partitioning identifies the two, almost equal political party-based communities showcased by the force-layout algorithm (Fig. 4.2F). At $r = 12.0$, however, we find a third and much smaller modularity peak with the corresponding clustering shown in Fig. 4.2G. This suggests that a very cohesive but small moderate cluster was merged with the right-wing community. Owing to its high local density, it became the Voronoi seed for this larger-scale right wing cluster. It appears that about 20% of the left-wing blogs are closer to this centrist group than to their own community, resulting in their split.

Next, we compared our method with five widely used algorithms (for details see Section 1.1): 1) the Louvain method based on local modularity optimization [7]; 2) the label propagation algorithm (LPA) [12]; 3) GANXiS, also called SLPA for speaker-listener label propagation algorithm [32, 33]; 4) link-communitiy detection (LCD) [9] and 5) Infomap (IM) [34, 35]. Our goal was to use efficient algorithms which can be tested on large networks with thousands of nodes.

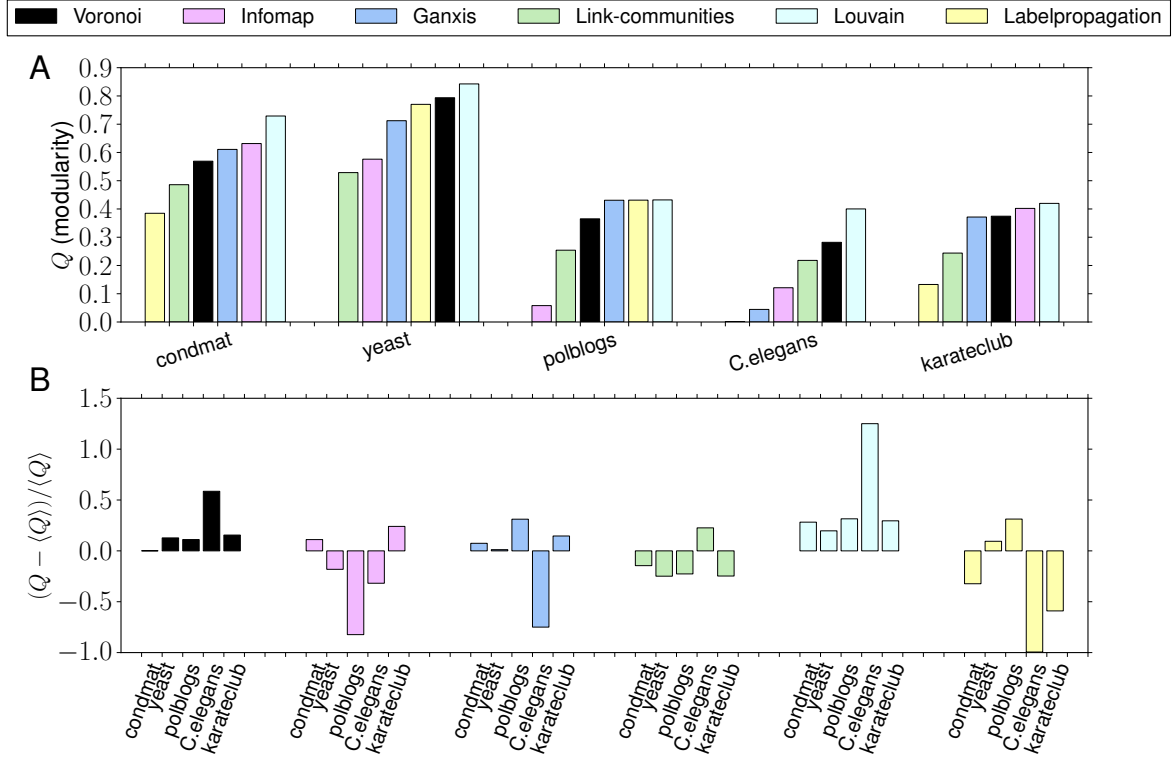


Figure 4.3: **Testing on real-world networks.** Modularities of the five real-world network clusterings obtained with different algorithms (see legend). (A) The modularity (Q) achieved by the algorithms in ascending order on each of the tested networks. (B) The relative error $(Q - \langle Q \rangle) / \langle Q \rangle$, where $\langle Q \rangle$ is the average over the results obtained with the six algorithms calculated separately for each network.

Fig. 4.3 shows a quality comparison between different methods, applied on the five real-world networks. Fig. 4.3A shows modularity of the communities detected by each algorithm in ascending order, sorted by networks type. Fig. 4.3B shows the modularity of individual methods relative to the average modularity across the different methods, sorted by method. Our algorithm provides the second best performance; the modularity it achieves is often close to maximum and always above average. We consider its reliability to be its most important feature. In contrast to our method, three other algorithms drastically fail on at least one network: the IM fails on POLBLOGS, while GANxis and LPA fail on CEL. LCD, on the other hand, is always around the average, but its performance is clearly weaker than that of our Voronoi partitioning method. The only algorithm which performs better is Louvain. This is expected, as Louvain algorithm is based on direct optimization of modularity, the very quality function we use to judge best performance.

In Figure 4.3 for each of the five networks we show the modularity function when increasing radius r and cluster size distributions at three different r values (highlighted with colored dots). It

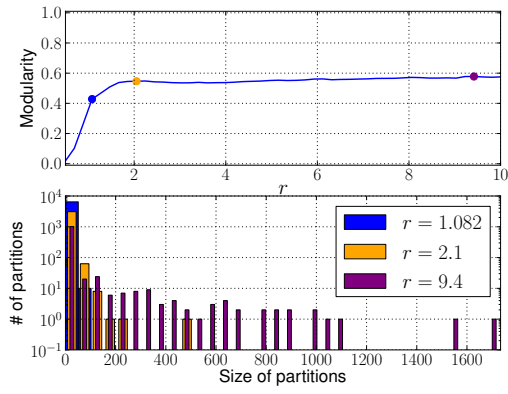
is natural that in all networks as r increases we have fewer but larger communities. The colormaps show mutual information comparison of partitioning with different radius r . A dot with coordinates (x, y) in the colormap indicates the mutual information of the two clusterings obtained with radius x and y . The mutual information increases from *blue* to *red* and the matrix is symmetric. All mutual information matrixes have the same resolution (50 points) per range, just the ranges are different depending on the size of the network. On the diagonal the mutual information is 1 because it is a self comparison, however the highest mutual information is reached above and beyond the diagonal, too. This happens when the clustering does not change in a certain interval of r .

We notice several distinct behaviors. In CONDMAT and YEAST the modularity seems to converge to a maximum, but the plateau is reached relatively early on. The colormaps indicate that the clustering itself changes smoothly: the mutual info is relatively large all along the plateau and small variations in r do not cause drastic changes in the community structure. In these networks the algorithm also detects many small communities, not only a few large ones.

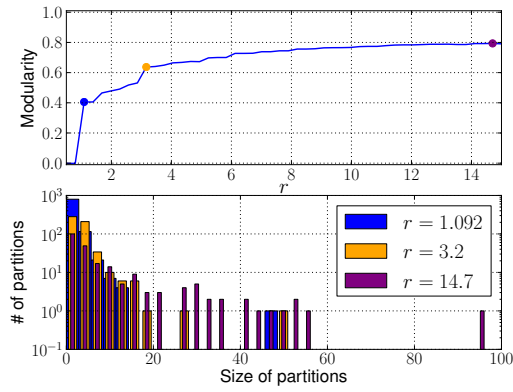
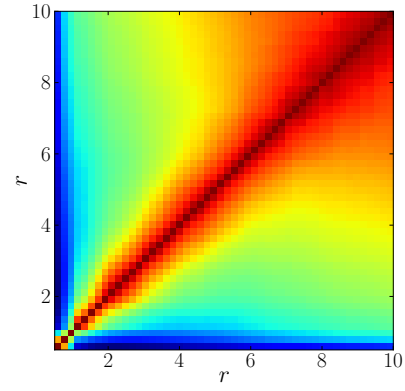
In POLBLOGS, C. Elegans and the Karate club the clustering changes more abruptly. Large red squares in the mutual information matrix indicate intervals where the clustering does not change at all. These correspond to plateaus in the modularity curve. When several such large squares exist (several local maxima or plateaus in the modularity curve) it indicates hierarchical structure.

4.3 Computational efficiency

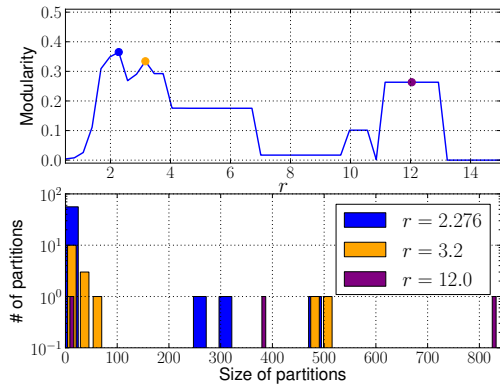
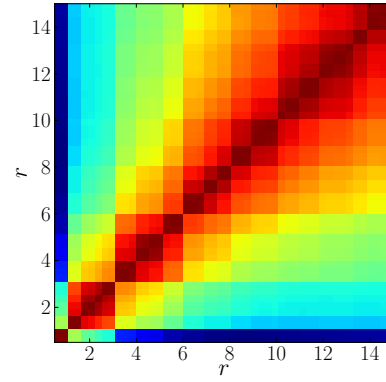
The algorithm has three main parts. i) First, we calculate the local density of each node and the ECC of each link, achievable in $O(N\langle k \rangle^2)$ and $O(M\langle k \rangle^2)$ steps, respectively. ii) Second, we identify the generator points by checking whether a node has the highest local density within a radius r . Breadth-first search covering the *whole* network from one node has complexity $O(M)$, but we find that search at depth is not necessary for optimal partitioning. Our results show that large values of modularity are achieved at small values of the radius parameter (typically between $r \in [2, 4]$), on both benchmark and real-world networks. This assures that breadth-first search must rarely go further than 1 – 3 steps from a node, keeping the algorithm efficient. We also note that breadth-first search starting from each node is not required. Once a node is identified to be a



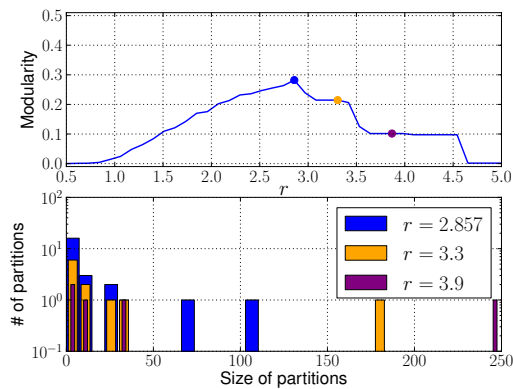
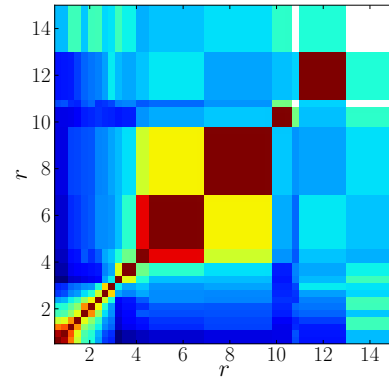
A. CONDMAT



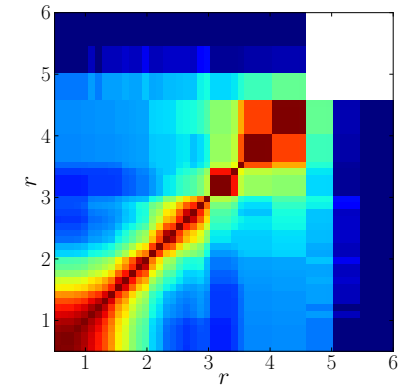
B. YEAST



C. POLBLOGS



D. CELEGANS



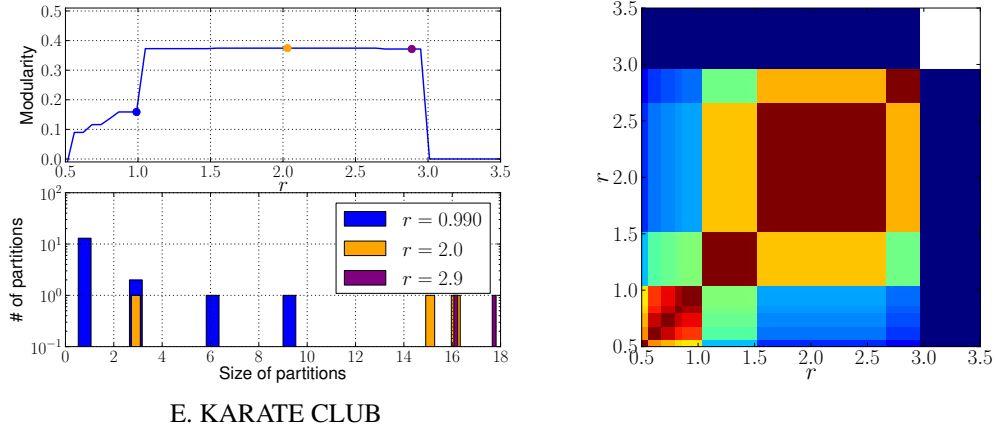


Figure 4.3: For the five real-world networks we show the modularity as function of radius and the community size distributions at some different highlighted values of r . The size of bins on the histograms are the same at each level of highlighted radius, it is varied only for visual clarity. The colormaps show mutual information comparison of partitioning with different radius r . A dot with coordinates (x, y) in the colormap indicates the mutual information of the two clusterings obtained with radius x and y . Highest mutual info is shown as red and lowest is blue. White indicates that the entire network is considered as one giant community and there is no clustering to be compared.

generator point, the other nodes in its r neighborhood no longer need to be considered as potential local maxima. Overall, the complexity of detecting generator nodes stays well below $O(Mg)$, where g is the number of communities. iii) Third, we partition the graph into Voronoi cells, a process with complexity of $O(N \log N)$ [15].

Chapter 5

Discussion and outlook

Voronoi diagrams are widely used for partitioning metric spaces. Their clear and simple definition is easily generalized to graphs. Here we showed that graph Voronoi diagrams are well suited for community detection on complex networks. We surmounted two major challenges: *First*, we defined a suitable distance measure between nodes. To this end, we chose the inverse of the edge clustering coefficient, $1/\text{ECC}$. ECC is frequently used to characterize the probability that two connected nodes belong to different clusters. *Second*, we provided a method to identify the Voronoi seeds or generator nodes that mark community centers. To this end we leveraged another simple measure, the relative local density. Nodes were chosen as Voronoi seeds if their relative local density was maximal in their neighborhood of radius r . Using these two measures, we showed that our method outperforms all but one of five commonly used method capable of partitioning large networks, only falling behind a method which directly maximizes the quality function used for our comparisons.

Distance measures on graphs can be defined in a variety of ways, all of which aim to quantify the probability that two nodes belong to the same community (high probability reflected by a small distance; for extensive review see [3]). Some of these measures are computed directly from network structure, such as the number of edge- (or vertex-) independent paths between vertices [37] or structural equivalence [36]. Other methods define distance based on random walks on networks, such as the commute-time between vertices [38] or the time of average first passage [39, 11]. Our choice of the ECC [10] as distance measure was motivated by its simplicity and computational efficiency. It is possible that employing more complex measures will further improve the mapping

between Voronoi partitions and network communities. We expect the trade-off between accuracy and efficiency to yield a whole spectrum of future Voronoi type methods for different application domains.

The most important parameter of our method is the radius r , which defines the range in which local density maxima are chosen as seeds. As the same r is used for each seed, we need to address whether the method can detect clusters with strongly varying sizes. The distribution of cluster sizes is a power-law in many real-world networks. Thus, large clusters are often accompanied by many small communities. Can our algorithm deal with these types of networks? Our choice of the edge clustering coefficient as the base of our distance measure guarantees that it can. When one imagines a complex network in space, it is natural to think that large clusters (with many nodes) "occupy more space". Using our distance measure, however, this is not so in the metric space we define. As ECC values tend to be much larger inside large, very dense clusters, they are compressed in metric space and their diameter is not typically larger than that of small clusters (sometimes quite the opposite). Moreover, distances between separate clusters, i.e., the length of edges connecting different clusters, do not cover multiple orders of magnitude. Rather, cluster placement in our metric space is relatively homogenous, even if the number of nodes in different clusters shows strong variations. This assures that constructing Voronoi cells can detect small communities next to large ones. This is in fact illustrated by the size distribution of clusters detected in CONDMAT and YEAST (Fig. 4.3).

We propose that the best strategy for using our method is to perform the Voronoi partitioning at increasing values of r . This procedure can reveal interesting details in the community structure of networks at different hierarchical levels. When a network has no specific hierarchical structure, quality functions such as modularity reach a large value at small r , after which the partitioning changes very slowly. The method is thus not sensitive to the parameter r and allows us to identify the correct number of communities whenever there is no further structure hidden within. The presence of such structure, or hierarchy, is signaled by the fact that the quality function has several local maxima (e.g., POLBLOGS). Between these maxima lie critical r values at which the detected community structure undergoes drastic changes (see Fig 4.3). These changes often correspond to community mergers into higher-scale clusters. The methods for automated detection of a network's structural hierarchy based on these mergers needs to be addressed in future research. Moreover,

the relationship between hierarchies detected in this way and those found by other hierarchical community detection methods (e.g., multi-resolution methods [41] and hierarchical methods [42]) also needs to be addressed.

Here we focused on graph partitioning problems where nodes are assigned a strict community membership. An important future goal of our research is to extend our Voronoi diagram-based method to overlapping communities. We envision two approaches to tackle this. First, we envision a fuzzy clustering method [3] that uses the distance between individual nodes and Voronoi seeds as the measure of their membership in each community. Nodes that fall close to Voronoi cell borders are prime candidates for regions of overlap. Second, tracking changes in single-node community membership with small perturbations of the radius r (in radius value range where the overall community structure is largely unchanged) may also serve to pinpoint network regions where communities overlap.

A few well-known hierarchical and spectral methods of data clustering have already been adapted to graphs [17, 18], albeit often via mapping the graph itself onto Euclidean space. Adapting the large set of centroid- and density-based methods available in data mining could further enrich community detection on complex networks, potentially customizing them for concrete applications. Our graph Voronoi diagram-based method combines some key features of these algorithms, and we expect it to be further extendable. For example, centroid-based methods such as k -means clustering concentrate on finding optimal cluster centers (Voronoi cell seeds in Euclidean space) in order to minimize the overall distance of all points from these seeds [43]. Our current approach, on the other hand, selects Voronoi seeds based on node density. We expect that methods to further optimize the position of seeds nodes would be promising for improving community detection. Along different lines, our distance measure opens the possibility of defining the concept of *density-reachability* on complex networks, a core concept of density-based clustering methods. (Two points in Euclidean space are density-reachable if they are closer than a threshold distance, and the density in their neighborhood is large enough; for precise description see [44, 45].) In summary, our method shows that an expediently chosen topology based metric on graphs can open new ways to approach graph community detection, narrowing the gap between the relatively young field of community detection and the well explored realm of cluster analysis in data mining.

Our results have been recently published in the New Journal of Physics [46] and presented in a

contributed talk on the NetSci conference in Copenhagen, 2013.

Publications and conferences

Publications

- **D. Deritei**, Zs. Lazar, I. Papp, F. Jarai-Szabo, R. Sumi, L. Varga, E. Regan, M. Ercsey-Ravasz, *Community detection by graph Voronoi diagrams*, New Journal of Physics 16, 063007 (2014) (impact factor: 4.06)
- I. Papp, M. Ercsey-Ravasz, **D. Deritei**, R. Sumi, F. Jarai-Szabo, R.V. Florian, A.I. Cabuz, Zs. Lazar, *The PH-Index: Hirsch Index of Individual Publications*, Proc. of ISSI 2013, Vol. 2, pp 2086-2088 (2013).

Conferences

- **D. Deritei**, Zs. Lazar, I. Papp, F. Jarai-Szabo, R. Sumi, M. Ercsey-Ravasz, Community detection by graph Voronoi diagrams, International School and Conference on Network Science, NetSci2013, Copenhagen, Denmark, June 3-7 2013, **oral presentation**
- I. Papp, M. Ercsey-Ravasz, **D. Deritei**, R. Sumi, F. Jarai-Szabo, R.V. Florian, A.I. Cabuz, Zs. Lazar, *The PH-Index: Hirsch Index of Individual Publications*, 14th International Society of Scientometrics and Informetrics Conference, ISSI2013, Vienna, Austria, July 15-19, 2013, **poster**
- Levente Varga, Ferenc J  rai-Szab  , **D  vid Deritei**, Zsolt I. L  z  r, Istv  n Papp, M  ria Ercsey-Ravasz and Razvan Valentin Florian, *Normalized cross-disciplinary article impact evaluation*, Global TechMining Conference GTM2014, Leiden, Netherlands, 2014.

Bibliography

- [1] Newman, MEJ. *Networks: An Introduction*. *Oxford University Press*. (2010).
- [2] Barabasi, AL. *Network Science*. (2012).
- [3] Fortunato, S. Community detection in graphs. *Phys. Rep.* **486** 75-174 (2010).
- [4] Newman, MEJ. Modularity and community structure in networks. **103** (23) 8577-8582 (2006).
- [5] Newman, MEJ and Girvan, M. Finding and evaluating community structure in networks *Phys. Rev. E* **69** (2) 026113 (2004).
- [6] Reichardt, J and Bornholdt, S. Statistical mechanics of community detection *Phys. Rev. E* **74** (1) 016110 (2006).
- [7] Blondel, VD. Guillaume, JL. Lambiotte, R. Lefebvre, E. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*. No. **10**. P10008 Vol.(2008).
- [8] Girvan, M and Newman, MEJ. Community structure in social and biological networks. *Proc. Natl. Acad. Sci. USA*. **99** 7821-7826 (2002).
- [9] Ahn, YY. Bagrow, JP and Lehmann, S. Link communities reveal multiscale complexity in networks. *Nature* **466** 761-764 (2010).
- [10] Radicchi, F. Castellano, C. Cecconi, F. Loreto, V. Parisi, D. Defining and identifying communities in networks. *Proc. Natl. Acad. Sci. USA* . **101** 2658 (2004).
- [11] Zhou, H. Distance, dissimilarity index, and network community structure. *Physical Review E*. **67** (6) 061901 (2003).

- [12] Raghavan, UN. Albert, R. Kumara, S. Near linear time algorithm to detect community structures in large-scale networks. *Phys. Rev. E*. **76** 036106 (2007).
- [13] Theodoridis, S. Koutroumbas, K. *Pattern Recognition*. Academic Press. (2008).
- [14] Han, J. Kamber, M. Pei, J. *Data Mining: Concepts and Techniques*. Morgan Kaufmann. (2011).
- [15] Aurenhammer, F. Voronoi diagrams - a survey of a fundamental geometric data structure. *ACM Computing Surveys*. Volume **23**, Issue 3, 345-405 (1991).
- [16] Jain, AK. Murty, MN. Flynn, PJ. Data Clustering: A Review. *ACM Computing Surveys*. Vol. **31**, No. 3. (1999).
- [17] Nishikawa, T and Motter, AE. Discovering Network Structure Beyond Communities. *Scientific Reports I*. **151** (2011).
- [18] Jin, H. Wang, S. Lib, C. Community detection in complex networks by density-based clustering. *Physica A*. **392** 4606-4618 (2013).
- [19] Chartrand, G and Zhang, P.A *First Course in Graph Theory*, Dover Publications. (2012).
- [20] Erwig, M. The graph Voronoi diagram with applications. *Networks*. **36** 156-163 (2000).
- [21] Bastian, M. Heymann, S. Jacomy, M. Gephi: an open source software for exploring and manipulating networks. *ICWSM* (2009).
- [22] Newman, MEJ. The structure of scientific collaboration networks. *Proc. Natl. Acad. Sci. USA* **98** 404-409 (2001).
- [23] Jeong, H. Mason, S. Barabási, AL and Oltvai, ZN. Lethality and centrality in protein networks. *Nature* **411** 41-42 (2001).
- [24] Adamic, LA and Glance, N. The political blogosphere and the 2004 US Election. *Proceedings of the WWW-2005 Workshop on the Weblogging Ecosystem*. (2005).
- [25] White, JG. Southgate, E. Thompson, JN. and Brenner, S. The Structure of the Nervous System of the Nematode *Caenorhabditis elegans*. *Phil. Trans. R. Soc. London*. **314** 1-340 (1986).

- [26] Watts, DJ and Strogatz, SH. Collective dynamics of ‘small-world’ networks. *Nature*. **393** 440-442 (1998).
- [27] Zachary WW. An information flow model for conflict and fission in small groups. *Journal of Anthropological Research*. **33** 452-473 (1977).
- [28] Lancichinetti, A. Fortunato, S. Radicchi F. Benchmarks graphs for testing community detection algorithms. *Phys. Rev. E*. **78** 046110 (2008).
- [29] Lancichinetti, A. Fortunato, S. Benchmarks for testing community detection algorithms on directed and weighted graphs with overlapping communities. *Phys. Rev. E*. **80** 016118 (2009).
- [30] Danon, L. Díaz-Guilera, A. Duch, J. and Arenas, A. Comparing community structure identification. *Journal of Statistical Mechanics: Theory and Experiment*. Issue **09**, (2005).
- [31] Yang, J. Leskovec, J. Defining and evaluating network communities based on ground-truth. *MDS '12 Proceedings of the ACM SIGKDD Workshop on Mining Data Semantics*. Article No. **3** (2012).
- [32] Xie, J and Szymanski, B. Towards Linear Time Overlapping Community Detection in Social Networks. *PAKDD* **7302**, 25-36 (2012).
- [33] Xie, J. Szymanski, B and Liu, X. SLPA: Uncovering Overlapping Communities in Social Networks via A Speaker-listener Interaction Dynamic Process. *IEEE ICDM workshop on DMCCI*. 344-349 (2011).
- [34] Rosvall, M and Bergstrom, CT. Maps of random walks on complex networks reveal community structure. **105** 1118-1123 (2008).
- [35] Rosvall, M. Axelsson, D. and Bergstrom, CT. The map equation. *Eur. Phys. J. Special Topics*. **178** 13-23 (2009).
- [36] Lorraine, F. White, HJ. Structural equivalence of individuals in social networks, *Journal of Math. Sociol.* **1** 49-80 (1971).
- [37] Ahuja, RK. Magnanti, TL. and Orlin, JB. *Network Flows: Theory, Algorithms, and Applications*, Prentice Hall (1993).

- [38] Fouss, F. Renders, JM. Saerens, M. Pirotte, A. Random-Walk Computation of Similarities between Nodes of a Graph with Application to Collaborative Recommendation. *IEEE. Trans. on Knowl. and Data Eng.* **19** (3) 355-369 (2007).
- [39] White, S. Smyth, P. A spectral clustering approach to finding communities in graphs. *Proc. of SIAM Int. Conf. on Data Minig.* 76-84 (2005).
- [40] Csardi, G and Nepusz, T. The igraph software package for complex network research. *Inter-Journal. vol. Complex Systems.* **1695**, (2006).
- [41] Arenas, A. Fernández, A. Gómez, S. Analysis of the structure of complex networks at different resolution level. *New J. Phys.* **9**, 176 (2008).
- [42] Sales-Pardo, M. Guimera, R. Moreira, A.A. Amaral, L.A.N. Extracting the hierarchical organisation of complex systems. *PNAS* **104**, 15224 (2007).
- [43] Lloyd, S. Least squares quantization in PCM. *IEEE Transactions on Information Theory* **28** (2), 129-137 (1982).
- [44] Ester, M. Kriegel, HP. Sander, J. Xiaowei, X. A density-based algorithm for discovering clusters in large spatial databases with noise. *AAAI Press*, 226-231 (1996).
- [45] Ankerst, M. Breunig, M. Kriegel, HP. Sander, J. OPTICS: Ordering Points To Identify the Clustering Structure. *ACM SIGMOD international conference on Management of data.* **28** (2), 49-60 (1999).
- [46] D. Deritei, Zs. Lazar, I. Papp, F. Jarai-Szabo, R. Sumi, L. Varga, E. Regan, M. Ercsey-Ravasz, Community detection by graph Voronoi diagrams, *New Journal of Physics* **16**, 063007 (2014)