

DENSITY-BASED CLUSTERING OF
INFORMATION NETWORKS BY
SUBSTRUCTURE OPTIMIZATION

BY

DUSTIN PHILIP BORTNER

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Computer Science
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2010

Urbana, Illinois

Adviser:

Professor Jiawei Han

ABSTRACT

Information networks, such as biological or social networks, contain groups of related entities, which can be identified by clustering. Density-based clustering (DBC) differs from vertex-partitioning methods in that some vertices are classified as noise. This approach is useful in practice to classify groups of related entities within noisy networks. The baseline DBC method involves constructing a maximal spanning forest (MSF) and deleting edges having weights below a threshold, leaving the connected components as clusters. In large networks, the data may contain large scale variances in the noise density level, which causes the baseline method to perform poorly. We investigate whether clustering within substructures of the MSF can improve the result. We present a new set of similarity measures for weighted networks that allow the MSF to connect vertices that are not connected in the original network. We convert the MSF to a hierarchy and merge it bottom-up by optimizing an objective function to produce a flat clustering. We present two new objective functions based on density and irregularity measures. Experiments show that the new similarity measures help to normalize noise density levels and outperform traditional set-based similarity measures; that the proposed substructure optimization method improves over the baseline in networks with many classes; and that DBC outperforms vertex-partitioning methods for classification in noisy networks. The results of the irregularity objective are promising, as it scales with the number of vertices, whereas objectives such as density or modularity scale with the number of edges.

ACKNOWLEDGEMENTS

I wish to thank my advisor professor Jiawei Han for his encouragement, financial and technological support, and invaluable advice in my research. I also thank my colleague Yintao Yu for providing formatted DBLP data and for his iNextCube Web site, from which I obtained the ranking scores of the conferences in each research area.

TABLE OF CONTENTS

Chapter 1	Introduction	1
Chapter 2	Related Work	4
Chapter 3	Clustering by Substructure Optimization	6
Chapter 4	Experiments and Analysis	14
Chapter 5	Conclusion and Future Work	28
Appendix		30
References		33

CHAPTER 1

Introduction

Information networks, such as biological or social networks, contain groups of related entities, which researchers desire to identify. For example, in *interactome mapping*, the goal is to create a complete diagram of all protein-protein interactions, for such uses as visualizing “disease mechanisms at a high level” [19]. Such networks are massive and complex, and only a fraction of the information may be of interest to a researcher in constructing part of the map.

Network clustering may be used to identify the groups of related entities, using link information. One approach to network clustering is *vertex partitioning*, in which all vertices of the network are partitioned into subsets. However, in very large real networks, it is likely that much of the network is irrelevant in identifying a particular function or pathway and that the network will contain much noise. Vertex partitioning methods such as [12] do not distinguish noise and give no ranking of cluster membership, but rather distribute noise throughout the clusters, which may be of limited use in noisy networks. *Density-based clustering* (DBC) is a distinct approach that can separate the interesting clusters from noise and can provide cluster membership ranking. The DBC approach may be practically better suited to the task of complex network clustering and classification.

In what we call the *baseline* DBC algorithm, a measure of similarity between network vertices is defined in terms of shared neighbors, a type of set-similarity measure. Using the similarities as edge weights, a maximal spanning forest (MSF) is constructed for the network. Edges having weight below some threshold are then deleted from the MSF, leaving the connected components as clusters. In constructing the MSF, the similarities are computed in a manner that relates them to network *density*. Unfortunately, complex networks may contain large-scale variances in the noise density level, which causes the baseline method to perform poorly. Figure 1.1 illustrates such a case for conferences in the DBLP Computer Science Bibliography network [15]. The bottom plot (*SCOT plot*) shows the similarities between adjacent vertices in the MSF, sorted by the structure-

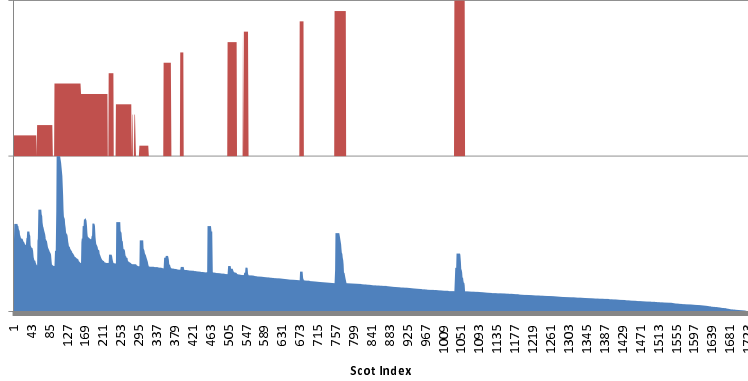


Figure 1.1: SCOT plot of the MSF (bottom) and clusters identified by substructure optimization of a local density objective (above). In contrast, the large scale sloping trend of the noise density level causes clusterings obtained by the baseline DBC algorithm to be of lesser quality.

connected order of traversal (SCOT) [5]. Clusters are visible as *dunes*, protrusions extending from a base of noise. The baseline DBC method is unable to produce a satisfactory clustering in this case, since the global density threshold must either miss some of the vertices in the dunes at lower “elevation” or must sacrifice distinguishing the clusters at higher “elevation” and include a great deal of noise in the clustering.

We investigate whether clustering within substructures of the MSF can improve the result. The outcome of our proposed substructure optimization methodology is illustrated by the clusters identified in the top plot of Figure 1.1. The cluster index for each vertex is plotted, demonstrating that substructure optimization is able to identify the clusters despite the variance of the noise density level over the MSF.

In [5], we present the hierarchical clustering algorithm HintClus. The hierarchical clustering provided by HintClus is useful for exploring the overall structure of the data, but it does not provide a flat clustering, which may be more useful in automating the network classification task. In this paper, we present a fresh substructure optimization approach to DBC, which produces a flat clustering. The distinct contribution of this paper as it relates to the network clustering area and other approaches is summarized in Table 1.

Specifically, our contributions in this paper are as follows:

- We present a set of new similarity measures for weighted networks that allow the MSF to connect vertices that are not connected in the original network. The new matrix-based

	Organizing all vertices of a network	Identifying dense sub-networks in noisy networks
Hierarchical	Agglomerative or divisive methods	HintClus
Flat	Vertex partitioning methods	Substructure Optimization

Table 1.1: The relation of the proposed method to other methods

similarity measures help to normalize noise density levels and outperform traditional set-based similarity measures.

- We study the relationship between modularity and density in network clustering.
- We present a novel algorithm for density-based clustering by substructure optimization. Experiments show that the proposed substructure optimization method improves over the baseline in networks with many classes, and that DBC outperforms vertex-partitioning methods for classification in noisy networks
- We define two new objective functions based on density and irregularity measures. The results of the irregularity objective are promising, as it scales with the number of vertices, whereas objectives such as density or modularity scale with the number of edges.

Chapter 2 outlines the relationship between the proposed methods and existing research. The substructure optimization DBC algorithm is discussed in detail in Chapter 3. Chapter 4 provides an extensive description and analysis of our findings, and Chapter 5 concludes with a summary and directions for future work.

Notation

We use `monospace font` in mathematical expressions to indicate operations defined in the Octave software [1] (similar to MATLAB). The expression E_{xy} denotes the (x, y) -th entry of the matrix-valued expression E , and $E_{xy}^k := (E^k)_{xy}$, not $(E_{xy})^k$.

CHAPTER 2

Related Work

The DBSCAN algorithm [10] produces a density-based clustering of points in a space with a distance measure (or with a point-to-point distance matrix). DBSCAN was first applied to unweighted undirected networks by Ertöz et al. [9], using a *shared nearest neighbor* (SNN) similarity to describe the similarity of adjacent vertices based on their common neighbors. This approach was subsequently duplicated by Xu et al. in SCAN [22] with the addition of hub and outlier identification and the use of a normalized SNN-based similarity, i.e., cosine similarity. (The formulas for the similarities of Ertöz et al. and Xu et al. are given in Section 3.1.) Analogous to OPTICS [2], Bortner et al. [5] present the *structure-connected order of traversal* (SCOT) algorithm. In addition, [5] proposes the algorithm HintClus to find the hierarchical density-based clustering that contains only the significantly distinct hierarchy nodes, as opposed to an agglomerative clustering that contains every binary merging, as in single-link clustering.

Density-based clustering relies upon the construction of a maximal spanning forest (MSF), which is a set of maximal spanning trees (MST's) corresponding to connected components. In *maximum spacing clustering*, the desired clustering is a partition that maximizes the single-link distance between each of the k clusters. The maximum spacing clustering can be obtained from the MST by deleting the $k - 1$ edges with the largest distances [14]. Similarly, the density-based clustering of SCAN is a collection of disjoint subsets obtained from the MSF by deleting the edges with similarity less than ε [5]. Single-link clustering recursively merges the clusters having maximum single-link similarity, the maximum similarity between vertices in distinct clusters. SCOT is exactly Prim's algorithm for a maximum spanning forest (MSF). So, both density-based and single-link clustering involve MSF construction, but the condition for the order of edge traversal is different.

We observe that in real, large networks, a global ε threshold is inappropriate because the data have an overall sloping trend. This results in the inability to obtain satisfactory clusters with any ε threshold. Our previous work [5] obtained a hierarchical clustering and used the heap

constructed from the SCOT sequence, which linearizes the MSF and hides some detail. In this paper, we seek a flat clustering that independently identifies clusters of differing base ε directly from the MSF. In addition, the prior density-based clustering work has been done on unweighted networks, while we address weighted networks. The presence of edge weights makes the selection of a similarity measure more important and difficult, which we also address.

Eigencluster, a *divide and merge* technique for clustering, is presented in [6]. Eigencluster first obtains a binary tree of subclusters, using a divisive spectral algorithm to approximate the minimum-conductance split, and then uses an agglomerative method on the binary tree to find the subtrees that optimize some objective function. The result is an optimal *tree-respecting* clustering, which is a partition of the data. Our work differs in that we do not use spectral methods, the analogous “divide phase” is performed by MSF construction, our trees are not binary, and the final clustering is not simply a partition of the data, but rather a collection of disjoint subsets that exclude some data as noise. We also propose wholly different and original objective functions.

CHAPTER 3

Clustering by Substructure Optimization

Our algorithm for density-based clustering by substructure optimization is outlined as follows:

1. Select a similarity measure;
2. Construct the maximal spanning forest (MSF);
3. Transform the MSF into a hierarchical clustering; and
4. Merge the hierarchical clustering bottom-up by optimizing an objective function to produce a flat clustering.

The subsequent sections discuss the steps in detail.

3.1 Similarity Measures

A number of similarity measures have been proposed in the literature. We have observed that different similarity measures can produce quite varied clustering results. We therefore view similarity measure selection as orthogonal to the rest of the density-based clustering process. In this section, we derive candidates for a measure of similarity between vertices in a weighted undirected network, using matrix formulas. This is a departure from the traditional set-based similarity measures.

For a homogeneous unweighted network, in which vertices represent objects of the same type, edge weights are typically assumed to be 1. This produces a binary adjacency matrix A , which is not directly amenable to density-based clustering. Consequently, Ertöz et al. [9, 21] propose the *shared nearest neighbor* (SNN) similarity measure. Define the *neighborhood* of a vertex x as the set of vertices adjacent to x (excluding x), denoted as $\Gamma(x)$. The SNN similarity of vertices x, y is

$$SNN(x, y) := |\Gamma(x) \cap \Gamma(y)| .$$

The normalized SNN similarity used by Xu et al. [22] is called *structural similarity*. Define the *structure* Γ' of a vertex x as the set of vertices adjacent to x plus x , that is, $\Gamma'(x) = \Gamma(x) \cup \{x\}$.

The *structural similarity* (SS) of vertices x, y is

$$SS(x, y) := \frac{|\Gamma'(x) \cap \Gamma'(y)|}{\sqrt{|\Gamma'(x)| |\Gamma'(y)|}} .$$

Both SNN and SS can be expressed in terms of matrices. Define D as the diagonal matrix containing the degrees of each vertex, that is, $D = \text{diag}(\text{sum}(A))$; then

$$SNN(x, y) = A_{xy}^2 ,$$

and

$$SS(x, y) = ((I + D)^{-1/2}(A + A^2)(I + D)^{-1/2})_{xy} .$$

The matrix forms of both similarity measures allow the similarity to be nonzero for nonadjacent vertices, unlike the definitions used in [9, 22].

SS is symmetric and normalized to the interval $[0, 1]$. Likewise, we wish to create a symmetric similarity measure $\sigma : V \times V \rightarrow [0, 1]$ based on the symmetric non-negative real matrix W . No particular reason for selecting the cosine similarity form of SS is given in [22]. We observe that the normalization in SS follows quite simply from the fact that A is a binary matrix. Since $0 \leq A_{xy} \leq 1$ ($\forall x, y \in V$), we have $(A + A^2)_{xy} \leq A_{xy}^2 + 1$ and $A_{xy}^2 \leq D_{xx}, D_{yy}$. Thus $(A + A^2)_{xy} \leq (I + D)_{xx}, (I + D)_{yy}$, so that $0 \leq SS(x, y) \leq 1$. Thus, we take some liberty to consider various candidates for a similarity measure σ .

We generate candidates for σ by combining different normalizations of the first and second powers of W , which correspond to paths of length 1 and 2, respectively.¹ Define the normalizations

$$\begin{aligned} N_d(X) &:= \text{Deg}(X)^{-1/2} X \text{Deg}(X)^{-1/2} , \text{ and} \\ N_m(X) &:= X / \max(X) , \end{aligned}$$

where $\text{Deg}(X) = \text{diag}(\text{sum}(X))$ (symmetric degree normalization) and $\max(X) = \max_{i,j} X_{ij}$, for a symmetric non-negative real matrix X that contains no all-zero columns (no isolated vertices). Our candidates for σ are listed in Table 3.1. The postfixes “-” and “+” refer to when the innermost normalization is applied relative to the exponentiation of W .

¹In this paper, we limit our study to paths of length 1 and 2.

Name	Formula
null	$N_m(W + W^2)$
null norm	$N_d(W + W^2)$
max-	$N_m(N_m(W) + N_m(W^2))$
max- norm	$N_d(N_m(W) + N_m(W^2))$
max+	$N_m(N_m(W) + N_m(W)^2)$
max+ norm	$N_d(N_m(W) + N_m(W)^2)$
deg-	$N_m(N_d(W) + N_d(W^2))$
deg- norm	$N_d(N_d(W) + N_d(W^2))$
deg+	$N_m(N_d(W) + N_d(W)^2)$
deg+ norm	$N_d(N_d(W) + N_d(W)^2)$
ss adj	$A \cdot (I + D)^{-1/2} (A + A^2) (I + D)^{-1/2}$
ss wgt	$A \cdot N_d(W + W^2)$

Table 3.1: Candidates for similarity measure σ

The reasoning behind the innermost normalization is that if W contains integers, such as the counts of occurrences, then W^2 will contain multiplicatively large values, such that the sum $W + W^2$ is dominated by W^2 . The application of innermost normalization is intended to equalize the relative importance of W and W^2 in the overall similarity. This set of candidates provides an indication of the behavior of reasonable similarity measures on the network, considering a maximum path length of 2. To impose the constraint that directly connected vertices be adjacent, we define $A := (W \neq 0)$, so that $\sigma' := A \cdot \sigma$, where $\cdot$ denotes the entrywise product.

Defining the similarity in terms of matrices has several benefits. By decoupling the calculation of the similarity measure from MST generation, we may easily experiment with different similarity measures. A modular approach increases the opportunity for parallelization, as efficient parallel implementations of arithmetical operations on sparse symmetric matrices already exist.

3.2 MSF and Hierarchical Clustering

The similarity measure induces a new network on the original one. From the new network's similarity matrix, we construct a maximal spanning forest (MSF), using the structure-connected order of traversal (SCOT) algorithm in [5]. However, this could also be done using any efficient implementation for constructing a MSF. The set of edges input to MSF construction is obtained by converting σ or σ' to its sparse edge-list representation. From the MST, it is simple to obtain the hierarchical clustering. Figure 3.1 lists the algorithm to perform the transformation. The algorithm is presented

Algorithm BuildHierarchy

Input: MSF $\mathcal{T} = (V, E)$, minimum similarity $\varepsilon \in [0, 1]$
Output: Hierarchical clustering $\mathcal{H}$

Sort E by descending weight;
Define array *membership* $[|V|]$;
Let $\mathcal{H} = \emptyset$ and $i = 1$;
While ($i \leq |E|$ and $E_i.weight \geq \varepsilon$) Do {
 Define new cluster c ;
 Let $c.weight = E_i.weight$;
 Let $mem_1 = membership[E_i.v_1]$;
 Let $mem_2 = membership[E_i.v_2]$;
 If ($mem_1 = \text{null}$) Then {
 Let $c.vertices \cup = \{E_i.v_1\}$;
 Let $membership[E_i.v_1] = c$;
 } Else {
 Let $r = \text{Root}(mem_1)$;
 If ($r.weight = E_i.weight$) Then {
 Let $c.vertices \cup = r.vertices$;
 Let $c.children \cup = r.children$;
 } Else {
 Let $c.children \cup = \{r\}$;
 Let $r.parent = c$;
 }
 $\mathcal{H}.Remove(r)$;
 }
 \\ ... similarly for mem_2 ...
 Let $\mathcal{H} \cup = \{c\}$;
 Let $i += 1$;
}
Define new cluster r ;
Let $r.weight = 0$;
Let $r.children = \mathcal{H}$;
For each $d \in r.children$, let $d.parent = r$;
Let $\mathcal{H} = r$;

Figure 3.1: Algorithm to convert MSF into hierarchical clustering

such that the while loop iterates until the threshold ε is reached. For $\varepsilon > 0$, this produces a partial hierarchical clustering that is used to obtain the baseline flat clusterings in our experiments. In Section 4.2, we analyze the candidates for σ based on the MSF's they generate.

3.3 Merging by Optimal Substructure

In the merge step, an optimal flat clustering $\mathcal{F}^*$ is obtained from the hierarchical clustering $\mathcal{H}$. Each node in $\mathcal{H}$ covers a set of vertices, and the node's children partition this set. Thus, a *flat clustering of the hierarchy* is a subset $\mathcal{F}$ of nodes in $\mathcal{H}$ such that each vertex is covered by at most

one node in $\mathcal{F}$. Some vertices are not covered by any node in $\mathcal{F}$, and these vertices are regarded as noise.

Because of the recursive structure of $\mathcal{H}$, we can use dynamic programming to find the flat clustering that optimizes some objective function $f_C(P_1, \dots, P_k)$, where C is the set of vertices covered by a node, and $P_1, \dots, P_k$ are pairwise disjoint sets of vertices such that $P_1 \cup \dots \cup P_k \subseteq C$. Since $\mathcal{H}$ can be traversed given its root node, we define the optimal flat clustering $\mathcal{F}^* = \text{Opt}(\mathcal{H})$. Then $\text{Opt}(\cdot)$ is given by the recurrence relation

$$\text{Opt}(C) = \text{argmax} \{f_C(C), f_C(\text{Opt}(C_1), \dots, \text{Opt}(C_k))\}$$

for argmax over arguments of f_C , and where $C_1, \dots, C_k$ are the children of C . Candidates for the objective function f_C produce an acceptable approximation to $\mathcal{F}^*$ by bottom-up optimization on subtrees of $\mathcal{H}$. Since in each recursion the decision is whether to return a single cluster or multiple subclusters, the objective function cannot be strictly biased toward either possibility.

3.3.1 Local Density Objective

Many proposed measures of cluster quality are related to the internal and external densities, which are ratios of existing edges to possible edges [20]. These densities are traditionally defined on unweighted networks, but since we have scaled σ to within $[0, 1]$, the modification for weighted networks is straightforward. For a cluster C , the *internal density* is

$$\delta_{\text{int}}(C) = \frac{\sum_{x, y \in C} \sigma(x, y)}{|C|(|C| - 1)} .$$

For a clustering $\{C_1, \dots, C_k\}$ of a network G , the internal density is

$$\delta_{\text{int}}(G \mid C_1, \dots, C_k) = \frac{1}{k} \sum_{i=1}^k \delta_{\text{int}}(C_i) ,$$

and the *external density* is

$$\delta_{\text{ext}}(G \mid C_1, \dots, C_k) = \frac{\sum_{i,j=1}^k \sum_{\substack{x \in C_i \\ y \in C_j}} \begin{cases} \sigma(x, y), & \text{if } i \neq j \\ 0, & \text{if } i = j \end{cases}}{|V|(|V| - 1) - \sum_{\ell=1}^k |C_\ell|(|C_\ell| - 1)} .$$

Feasible flat clusterings have internal density greater than the density of the entire network and external density less than the density of the entire network. An optimal clustering will maximize the internal density and minimize the external density. Moreover, an optimal clustering should not discard too many vertices as noise to achieve these optimal densities. This makes our clustering problem different from those in which the clustering is a strict partition of the vertices. We define the local density objective function

$$f_C(P_1, \dots, P_k) := \delta_{\text{int}}(C \mid P_1, \dots, P_k) - \alpha \delta_{\text{ext}}(C \mid C', P_1 \cup \dots \cup P_k) ,$$

where $C' = C \setminus (P_1 \cup \dots \cup P_k)$, and $\alpha \geq 0$. The first term allows us to maximize the internal density of the clustering, while the second term accounts for the cost of discarding the vertices in C' as noise. The parameter α weights the cost, and influences the tendency of subclusters to merge. Higher α corresponds to fewer clusters in the final clustering. Since at each recursion, the universe for the external density is only $C \subseteq V$, the external density for the current node is $\delta_{\text{ext}}(C \mid C) = 0$. Thus, although maximizing only the internal density would strictly bias the objective function such that $\text{Opt}(\cdot)$ always returns multiple subclusters, the external density term counters this bias. The result is an objective function that balances the maximization of the internal density with the cost of discarding vertices as noise.

We considered other possible objective functions, including similarity-based modularity Q_s [11] and relative density ρ [20], defined as

$$Q_s(G \mid C_1, \dots, C_k) = \sum_{i=1}^k \left(\frac{IS(C_i)}{TS(G)} - \left(\frac{DS(C_i)}{TS(G)} \right)^2 \right) , \quad (3.1)$$

where $IS(\cdot)$ is the total internal similarity of a cluster, $DS(\cdot)$ is the total external similarity of a

cluster, and $TS(\cdot)$ is the total internal similarity of a network; and

$$\rho(G \mid C_1, \dots, C_k) = \frac{1}{k} \sum_{i=1}^k \frac{\deg_{\text{int}}(C_i)}{\deg_{\text{int}}(C_i) + 2\deg_{\text{ext}}(G \mid C_i)} ,$$

where $\deg_{\text{int}}(\cdot)$ and $\deg_{\text{ext}}(\cdot \mid \cdot)$ are the total internal and external similarities, respectively, of a cluster w.r.t. a network. However, it is easy to show that each of these functions is strictly biased. $Q_s(C \mid C) = 0 \leq Q_s(C \mid P_1, \dots, P_k)$, so that Q_s is biased toward multiple subclusters, and $\rho(C \mid C) = 1 \geq \rho(C \mid P_1, \dots, P_k)$, so that ρ is biased toward a single cluster. We have found experimentally that this behavior generally holds if we instead consider the entire set of vertices V as the universe for the external density in each recursion. For these reasons, we propose a new objective function constructed from the basic density functions.

3.3.2 Irregularity Objective

We define *irregularity* as a measure of variation in edge weights over a hierarchical structure. Irregularity takes into account the relative proximity of hierarchical substructures, rather than simply treating edges as an unordered collection and computing the statistical variance. It is defined as

$$\text{Irr}(C) := \sum_{i=1}^k \text{Irr}(C_i) + \begin{cases} G(C) \cdot H(C), & \text{if } k > 1 \\ 0, & \text{otherwise} \end{cases} , \quad (3.2)$$

where C_i are the child nodes of C .

$$G(C) := \frac{\sum_{i=1}^k n_i (\mu(C_i) - w)^2}{\sum_{i=1}^k n_i}$$

is the average squared difference of the weighted mean weight $\mu(C_i)$ to the weight w of C , where n_i is the count edges in C_i . The mean weight is given by

$$\mu(C) := \frac{\sum_{i=1}^k n_i \mu(C_i) + kw}{\sum_{i=1}^k n_i + k} .$$

We define the *uniformity* of the edge counts of C 's child nodes as

$$H(C) := \frac{\left(\prod_{i=1}^k n_i\right)^{1/k}}{\frac{1}{k} \sum_{i=1}^k n_i} . \quad (3.3)$$

Uniformity takes values between 0 and 1, where 1 represents complete uniformity and 0 represents total nonuniformity. Intuitively, when the edge counts are more uniform, each child is more likely to represent a distinct cluster. H is the ratio of the geometric mean to the arithmetic mean, and has the useful property of being an *algebraic aggregate function*, in the sense of [13]. We give a derivation of uniformity in appendix Section A.1.

We define the irregularity objective function

$$f_C(P_1, \dots, P_k) := \max\{\text{depth}(p) \mid p \in \{P_1, \dots, P_k\} \wedge \Delta\text{Irr}(p) \geq \gamma\} ,$$

where $\text{depth}(p)$ is the depth of node p in the hierarchy, and $\gamma = \alpha \text{Irr}(\mathcal{H})$, $0 \leq \alpha \leq 1$, is a fraction of the total irregularity of the hierarchical clustering. The expression $\Delta\text{Irr}(p)$ denotes the change in irregularity at p , given by the right-hand term in (3.2). Hence, this objective function will yield the deepest subclusters in the hierarchy that contribute at least a fraction α of the total irregularity.

CHAPTER 4

Experiments and Analysis

To examine the clustering quality of the proposed substructure optimization algorithm, we use homogeneous networks with known class labels. The homogeneous networks are derived from the Wisconsin prognostic breast cancer (“WPBC”) and Wine recognition (“Wine”) datasets [3] and from the Digital Bibliography and Library Project (“DBLP”) database [15]. The WPBC data have two classes, benign and malignant, while the Wine data have three classes, the cultivars of the grapes. To obtain classifications for the conferences in the DBLP network, we select the top 50 conferences for each of the eight research areas, according to the iNextCube project [23]. Since some conferences appear in the top-50 of more than one research area, we use the ranking scores to derive a probability for each classification.

For each dataset, the network used for the study is produced after several steps of normalization, homogeneous network reduction and sparsification. The data preprocessing steps are summarized in Tables 4.1 and 4.2.

The networks thus obtained contain no noise, since each vertex has a class label. To study the effect of the existence of noise, we create two additional networks by adding noise to the Wine and DBLP networks. Table 4.3 lists the properties of the five networks used in the study. The details of the network preparation are presented in the following section.

Dataset	Raw Data	Homogeneous Network
WPBC	z -normalized	none
Wine	z -normalized	none
DBLP	none	symmetric degree normalized

Table 4.1: Summary of data standardization

Dataset	Data Type	Reduction	Sparsification
WPBC	vector	cosine	$w \geq 0.25$
Wine	vector	Manhattan	top-10
DBLP	heterogeneous network	matrix inner product	top-10

Table 4.2: Summary of homogeneous network reduction

Network	Vertices	Edges	Classes
WPBC	683	99451	2
Wine	178	1216	3
Wine Noisy	356	2949	3 + noise
DBLP	346	2793	8
DBLP Noisy	692	6208	8 + noise

Table 4.3: Network properties

4.1 Network Preparation

4.1.1 Noiseless Networks

In the WPBC and Wine datasets, each object is represented as a numeric feature vector. Such data can be considered a bipartite network consisting of two vertex types, object and feature, with edge weights given by the feature vectors. To reduce a vector-based bipartite network to a homogeneous network of objects, two similarity functions are considered: cosine similarity and similarity under a p -norm.

Cosine similarity is based on the angle between two vectors in Euclidean space and is defined as

$$\sigma_{\cos}(x, y) := \frac{x \cdot y}{\|x\| \|y\|} ,$$

where x, y are feature vectors. Cosine similarity takes values between -1 and 1, where 1 means identical, -1 means opposite, and 0 means independence. Since we deal with homogeneous networks with positive edge weights, the cosine similarity is transformed as

$$\sigma'_{\cos}(x, y) = \frac{1 + \sigma_{\cos}(x, y)}{2} .$$

The p -norm is defined as

$$\|x\|_p := \left(\sum_{i=1}^n |x_i|^p \right)^{1/p} .$$

For $p = 1$, we have the Manhattan or cityblock norm, and for $p = 2$, we have the Euclidean norm.

The Gaussian function

$$g_{\beta}(x, y) = \exp \left(-\frac{\|x - y\|^2}{\beta^2} \right)$$

has been used to induce a similarity matrix on a set of vectors for spectral clustering [17, 4]. We extend this to the p -norm as

$$g_{\beta,p}(x, y) = \exp \left(-\frac{\|x - y\|_p^p}{\beta^p} \right) .$$

We call $g_{1,1}$ the *Manhattan similarity*.

Prior to reduction, the feature vectors are standardized by the z -normalization

$$z_{\mu,\sigma}(x) = \left(\frac{x_1 - \mu_1}{\sigma_1}, \dots, \frac{x_n - \mu_n}{\sigma_n} \right) ,$$

where $x \in \mathbb{R}^n$ is a feature vector, and μ_i and σ_i are the mean and standard deviation, respectively, of the i th component over the entire dataset.

The heterogeneous DBLP network consists of two relations, paper-conference and paper-author. To reduce it to a homogeneous network of conferences, we represent each relation as a matrix and compute a matrix inner product. Let R_{PC} be the occurrence matrix with papers on rows and conferences on columns, where each entry contains 1 if the paper-conference combination occurred; otherwise, 0. Similarly, let R_{PA} be the occurrence matrix with papers on rows and authors on columns. Then a similarity matrix for the conference network is given by

$$R_{CC} = R_{PC}^T R_{PA} R_{PA}^T R_{PC} .$$

This is equivalent to the SQL query in Figure 4.1.

```

SELECT  r_pc1.cID, r_pc2.cID, COUNT(*)
FROM    r_pc AS r_pc1, r_pa AS r_pa1,
        r_pa AS r_pa2, r_pc AS r_pc2
WHERE   r_pc1.pID = r_pa1.pID
        AND r_pa1.aID = r_pa2.aID
        AND r_pa2.pID = r_pc2.pID
GROUP BY r_pc1.cID, r_pc2.cID

```

Figure 4.1: SQL query for DBLP network reduction

The edge weights of the conference network are then degree normalized, using N_d . Without the degree normalization step, the deg+ norm similarity would produce the best MSF; however, with this step, the max+ norm similarity produces the best MSF. Since the WPBC and Wine datasets produce the best MSF's with the max+ and max+ norm similarities and to reduce the possibility that our particular reduction method would bias the comparison with other algorithms, we perform the degree normalization step.

Finally, we desire to sparsify the resulting homogeneous networks by keeping only the edges of the k nearest neighbors for each vertex, as described in [9]. For the DBLP and Wine datasets, for each vertex, we keep the adjacent edges with the top 10 weights. However, for the WPBC network we proceed differently. Sparsification is only useful when it does not affect the MSF. For example, if a vertex x is connected to vertices $y_1, \dots, y_{20}$ and those edge weights are the highest of any edges adjacent to $y_1, \dots, y_{20}$, then keeping only the edges of the 10 nearest neighbors of x will affect the MSF. Indeed, this is a concern for the WPBC network, where top- k sparsification produces nearly-isolated subnetworks, i.e., creates false clusters. Thus, edges with weight $w \geq 0.25$ are kept.

4.1.2 Noisy Networks

The Wine Noisy and DBLP Noisy networks are created from the Wine and DBLP networks used in the study. To the 178 vertices in the Wine network, 178 noise vertices are added. Each noise vertex is assigned a distinct class label, which is not one of the original class labels. This guarantees that inclusion of noise vertices in the clustering can only have a negative affect on the MVC accuracy, which is discussed in Section 4.3.1. Each noise vertex is connected to 10 vertices, which are uniformly selected at random from the 178 original vertices. No noise vertices are directly connected. The weights of the noise edges are selected from a Gaussian distribution having the mean and standard deviation of the original edge weights. For the DBLP Noisy network, 346 noise vertices are added to the original 346 vertices. The connections, edge weights, and class labels are created similarly to the Wine Noisy network.

4.2 Similarity Measure Evaluation

The first part of our study is the evaluation of the similarity measure candidates in generating the MSF’s for each network. In contrast to the straightforward simplicity with which the SSN and SS similarities are presented in [9, 22], we find that the MSF and the clustering obtained therewith depend strongly on the similarity measure used, and the results varied greatly over our set of similarity measure candidates. Our observations of the similarity candidates are summarized in Table 4.4. In general, we found that using the binary-valued adjacency matrix produced extremely poor results compared to those obtain with a real-valued similarity matrix. Similarly, constraining the similarity measures by disallowing nonzero similarity for nonadjacent vertices produced relatively poor results. This is a significant observation, for it changes the construction MSF from using only edges that appear in the original network to using edges that appear in $W + W^2$, i.e., paths of length one and two in the original network. We first discuss the relationship between MSF the and the SCOT plot, and we then discuss the ability to judge a similarity measure from the MSF and SCOT plot.

For each dataset, the similarity matrix is computed and the MSF is constructed. The SCOT plot is a linear representation of an MSF, generated by plotting the weights of edges traversed in

Similarity	General Behavior
cos, ss adj, ss wgt	Produces one large bushy component in the MSF with few small relatively dense components. SCOT plot shows one large, rounded dune with small sparse irregularities. Yields poor clustering results, classifying most vertices as noise.
max+, max+ norm	Produces bushy components that correspond closely with classes. Yields the best clustering results.
null, null norm, max-, max- norm	Similar to max+ and max+ norm but with somewhat less pronounced dunes. Yields similar clustering results, but has a greater tendency to merge clusters.
deg-, deg-norm, deg+, deg+ norm	Produces stringy, thin MSF’s with longest average path lengths of all similarities. SCOT plot shows a very irregular surface with many small dunes. Destroys cluster structure and amplifies noise, producing many small clusters. An exception is if the network is produced by matrix inner product reduction of a heterogeneous network; it then behaves like “max+” family.

Table 4.4: Observed similarity measure behavior

SCOT order to construct the MSF. Figure 4.2(a) shows the SCOT plot for the WPBC network, which corresponds to the MST in Figure 4.2(b). The max+ similarity is used. The two distinct dunes in the plot correspond to the two bushy components of the MST. In fact, each dune corresponds to a class: the benign class on the left and the malignant class on the right. The null similarity produced a similar but less pronounced pair of dunes. For the Wine dataset, a similar correlation between the MST and the SCOT plot is observed, as shown in Figure 4.3. Again, the three classes are clearly visible in the SCOT plot, produced with max+ norm similarity.

We observe that the best clustering results are produced when the SCOT plot is closely correlated with the class labels. Figure 4.4 shows the class labels plotted above the SCOT plot for the DBLP network. In general, similarity measures that yield a good clustering produce a MSF that has a moderate average path length and is moderately irregular. Average path length can be

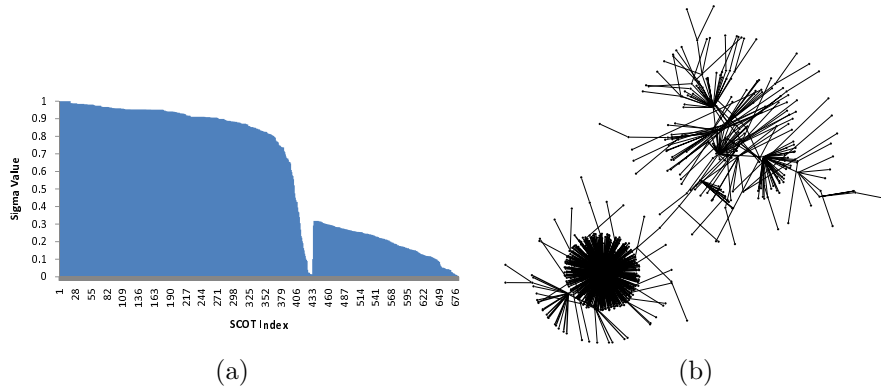


Figure 4.2: SCOT plot (a) and MST (b) of WPBC network

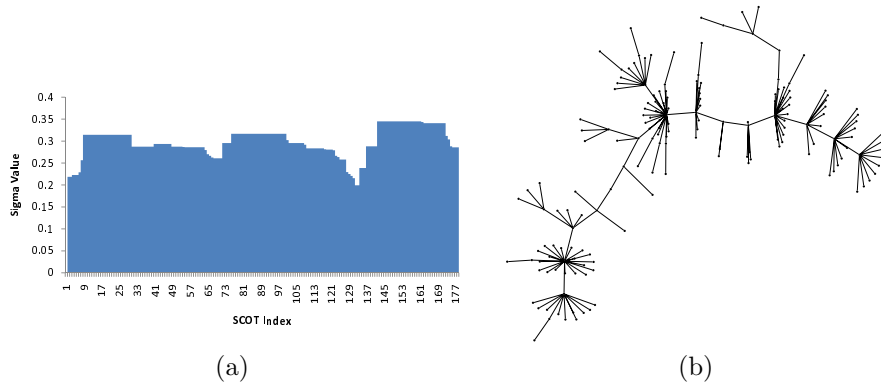


Figure 4.3: SCOT plot (a) and MST (b) of Wine network

calculated by

$$APL = \frac{\sum_{p=1}^m p \sum_{i,j} T_{ij}^p}{\sum_{p=1}^m \sum_{i,j} T_{ij}^p},$$

where T is the upper-triangular adjacency matrix of the MST (a directed graph), and m is the greatest integer for which some entry in T^m is nonzero. After observing a number of SCOT plots for networks with known classification, one is able to discern a promising similarity measure from the SCOT plot without performing the actual clustering. Based on the similarity measure observations, the best-performing similarity measure for each network is selected for the remainder of the study. The selected similarity measures are listed in Table 4.5.

4.3 Classification Accuracy

To study the classification accuracy of the proposed algorithm, we compare it against several other clustering algorithms. The baseline clustering is given by the density-based clustering obtained

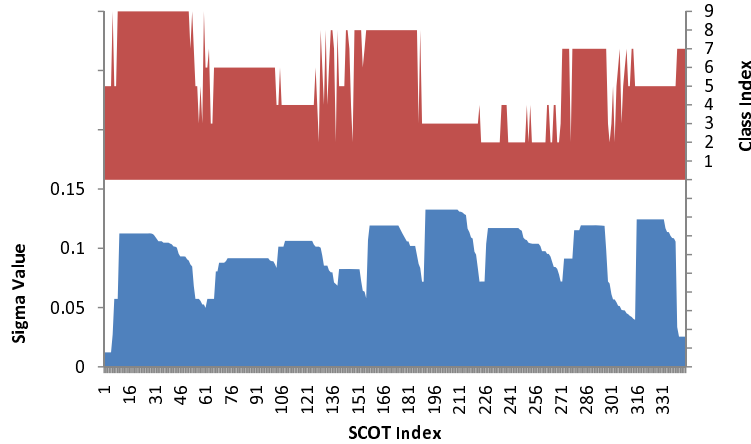


Figure 4.4: SCOT plot (bottom) and corresponding class indices (above) of DBLP network

Network	Sigma
WPBC	max+
Wine	max+ norm
Wine Noisy	max+ norm
DBLP	max+ norm
DBLP Noisy	max+ norm

Table 4.5: Selected similarity measures for each network

from the connected components remaining after deleting edges with similarity less than ϵ from the MSF, as described in [5]. We also compare it against the vertex-partitioning network clustering algorithms FastCommunity by Clauset et al. [7] and Graclus by Dhillon et al. [8]. To make a reasonable comparison between the density-based flat clusterings that classify some points as noise and the vertex-partitioning clustering, we use the information theoretic mutual information (MI) measure and a majority-vote classification accuracy (MVC) measure. The measures used to compare the clusterings are detailed in the following section.

4.3.1 Cluster Comparison Measures

Mutual Information

The *mutual information* (MI) measures the dependence of two random variables. For clustering comparison, it can be defined as

$$MI(K; K') = \sum_{k \in K} \sum_{k' \in K'} p(k, k') \log \left(\frac{p(k, k')}{p(k)p'(k')} \right),$$

where K and K' are two clusterings of a set of objects, $p(\cdot, \cdot)$ is the joint probability distribution function of K and K' , and $p(\cdot)$ and $p'(\cdot)$ are the marginal probability distribution functions of K and K' , respectively. The details of the joint probability distribution function are given in the appendix Section A.2.

$MI(K; K')$ is also known as *information gain*, since it represents the reduction of uncertainty about the class K when the attribute K' is known. Information gain is popularly used in selecting the partitioning attribute in building a decision tree, but it is well known to be biased toward clusterings with high selectivity [18]. Rectifications have been proposed, but at most MI can provide only an indication of clustering quality. Since in our results, the clustering results all have similar numbers of clusters, we use MI with the caveat that it alone is not sufficient to judge clustering quality. Thus, we also define the majority-vote classification accuracy (MVC) measure.

Majority-Vote Classification Accuracy

The *majority-vote classification accuracy* (MVC) is the fraction of correctly classified objects if the most frequently occurring class in each cluster is used as the predicted class label. In this sense, the class label of each cluster is determined by majority vote. MVC is defined as

$$MVC(K; K') = \frac{\sum_{k' \in K'} \max_{k \in K} |\{x \in k' \mid x \in k\}|}{\sum_{k' \in K'} |k'|},$$

where K' is the clustering and K is the true classification. Assuming that the probability of a researcher determining the true class label of each object is uniformly distributed, the MVC measures the accuracy of classification by a researcher who uses this method to classify a cluster using a sample. While MVC can also be biased toward clustering with many clusters, we observe that for similar numbers of clusters in each clustering being compared, this bias is not a concern. In our results, we present both the MVC calculated by excluding the vertices classified as noise by the algorithm (“MVC excl. noise”) and the MVC calculated by including these vertices as another cluster (“MVC incl. noise”).

Coverage

Since the density-based clusterings classify some vertices as noise, it is useful to measure the *coverage*, the fraction of non-noise vertices that are classified as non-noise by the algorithm. In the noiseless networks, this is simply the number of vertices in the flat clustering. In the noisy networks, this is the fraction of vertices that are non-noise and that are present in the flat clustering. In a noisy network, it is possible to attain a high coverage while including many noise vertices in the clustering. Indeed, the vertex-partitioning algorithms always include 100% of the vertices in the clustering. Since we have assigned distinct class labels to each noise node, the penalty for including many noise nodes is reflected by the MVC. It is highly improbable for a cluster to be given a noise label by majority vote, especially since the minimum number of vertices in a density-based cluster is determined by the parameter μ so that clusterings cannot be too highly selective.

Effective MVC

Given that a only a fraction of non-noise vertices may be covered and that, of those, only a fraction are correctly classified by MVC, we define the *effective MVC* as the product of the coverage and the MVC excluding noise. This is interpreted as the probability of a vertex being identified as non-noise and being classified correctly using random sampling. This provides a reasonable measure of the overall usefulness of the clustering algorithm to a researcher in performing classification.

4.3.2 Clustering Evaluation

Each of the density-based clustering algorithms is executed over a mesh of parameter combinations. The clustering procedure is outlined in Figure 4.5. The FastCommunity (FC) algorithm is run in two passes, in the first pass, the maximum modularity¹ MaxQ is determined, and in the second pass, the clustering is produced using the MaxQ value. The Graclus (GC) algorithm requires the number of clusters k as a parameter. Both the FC and GC algorithms are run once per network, since the parameter selection is obvious. The density-based clusterings with the highest effective MVC are selected for comparison, and the clustering results are summarized in Table 4.6.

MI has some correlation with effective MVC for the non-noisy networks, where the top-2 algorithms by MI correspond to the top-2 algorithms by effective MVC for two out of three

¹This modularity differs from the structural-similarity based modularity Q_s .

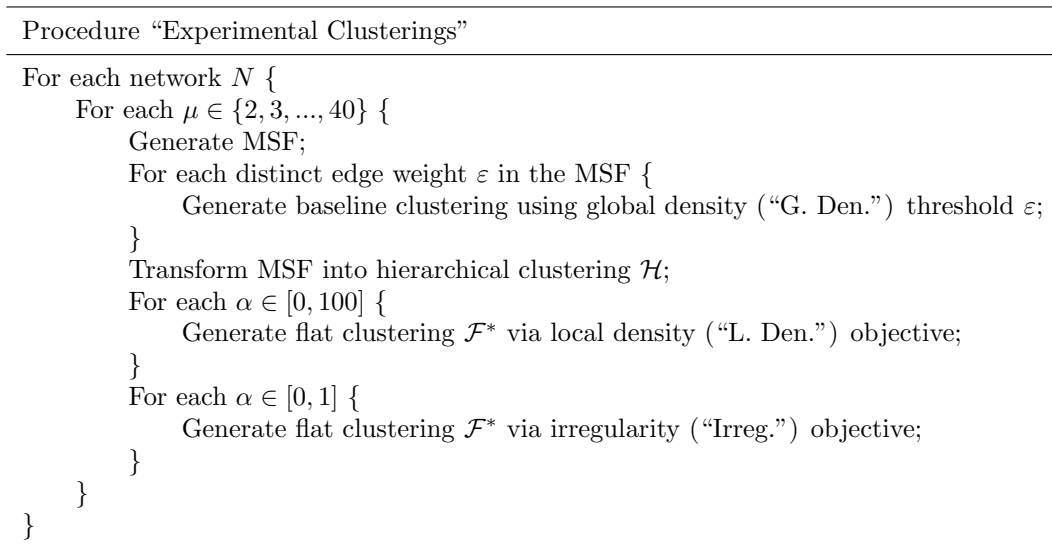


Figure 4.5: Procedure to generate the experimental clusterings

Method	Parameters	Clusters	MI	Cover- age (%)	MVC Excl. Noise (%)	Effec. MVC (%)	MVC Incl. Noise (%)
WPBC (2 classes)							
G. Den.	μ : 10, ε : 0.01538	2	0.7699	99.12	97.49	<i>96.63</i>	97.07
L. Den.	μ : 10, α : [0, 20]	2	0.7699	99.12	97.49	<i>96.63</i>	97.07
Irreg.	μ : 10, α : [0, 1]	2	0.7699	99.12	97.49	<i>96.63</i>	97.07
FC	MaxQ: 0.2346	2	0.7587	100.00	96.78	96.78	96.78
GC	k : 2	2	0.7762	100.00	97.36	97.36	97.36
Wine (3 classes)							
G. Den.	μ : 27, ε : 0.2155	4	1.2696	94.94	93.49	<i>88.76</i>	93.26
L. Den.	μ : 20, α : [7.5, 50]	3	1.3136	90.45	96.27	87.08	94.38
Irreg.	μ : 27, α : [0.05, 0.90]	3	1.1088	94.94	90.53	87.08	88.76
FC	MaxQ: 0.6032	3	1.3612	100.00	96.63	96.63	96.63
GC	k : 3	3	1.4274	100.00	97.75	97.75	97.75
DBLP (8 classes)							
G. Den.	μ : 20, ε : 0.07196	12	1.8200	76.88	86.21	66.27	69.50
L. Den.	μ : 19, α : [45, 80]	9	1.8681	89.88	78.06	<i>70.17</i>	73.00
Irreg.	μ : 19, α : [0.8, 0.9]	10	1.8313	90.46	76.52	69.22	74.00
FC	MaxQ: 0.6915	8	2.0956	100.00	75.89	75.89	75.89
GC	k : 8	8	2.2252	100.00	79.31	79.31	79.31
Wine Noisy (3 classes + noise)							
G. Den.	μ : 28, ε : 0.2440	2	0.9398	33.15	85.59	56.74	41.85
L. Den.	μ : 28, α : [0, 5]	4	1.4602	79.21	74.86	52.99	42.98
Irreg.	μ : 28, α : 0	4	1.4602	79.21	74.86	52.99	42.98
FC	MaxQ: 0.3116	4	1.4900	100.00	42.70	42.70	42.70
GC	k : 3	3	1.4310	100.00	47.75	47.75	47.75
DBLP Noisy (8 classes + noise)							
G. Den.	μ : 28, ε : 0.1047	9	1.8691	81.79	70.03	57.28	38.07
L. Den.	μ : 29, α : [0, 11.25]	10	2.2329	90.17	51.84	46.74	38.47
Irreg.	μ : 28, α : [0.6, 0.7]	10	2.3692	91.62	52.90	48.46	40.35
FC	MaxQ: 0.4507	9	3.1003	100.00	2.59	2.59	2.59
GC	k : 8	8	2.6800	100.00	2.31	2.31	2.31

Table 4.6: Classification results. Bold font indicates maximum effective MVC; italic font indicates maximum effective MVC among the density-based clusterings. The notation $\alpha : [c, d]$ indicates that the same results are observed for $c \leq \alpha \leq d$.

networks. However, for the noisy networks, the top MI algorithms are the worst effective MVC algorithms. This may be due to the fact that the noise vertices are assigned distinct labels, making them highly selective, given the bias of information gain. The effective MVC provides a better quantification of classification usefulness. Within the non-noisy networks, the vertex-partitioning algorithms outperform the density-based algorithms in terms of effective MVC. GC yields the top effective MVC, which might be expected considering that the number of clusters is specified. Among the density-based algorithms, we observe similar performance for each of the WPBC and Wine datasets, with the baseline algorithm slightly ahead for Wine. However, for the DBLP

network we see better performance by the substructure optimization algorithms, with L. Den. first among the three. Considering the structure of the MSF for the DBLP network, as reflected by the SCOT plot in Figure 4.4, this is an instance in which a single global ε threshold cannot produce the best possible clusters, and the substructure optimization algorithms successfully improve over the baseline.

For the noisy networks, we see a significant but expected change in the effective MVC of the vertex-partitioning algorithms compared to the density-based algorithms. Although FC identified an extra class in for both Wine Noisy and DBLP Noisy, inspection of the results reveals that the noise clusters are not grouped into the extra cluster, but rather are evenly distributed among all clusters. Similar behavior is observed for GC, even when we increase the value of parameter k . This demonstrates that vertex clustering algorithms are not as useful in the classification task with noise, since there is no indication of which vertices are more likely to be noise, whereas the density-based algorithms can provide a ranking by similarity within the clusters, and can be adjusted to be more or less inclusive of noise.

The baseline density-based algorithm outperforms the substructure optimization algorithms for the noisy networks, significantly for DBLP Noisy. Several undesirable behaviors of the substructure optimization algorithms are observed for the noisy networks. First, although the noise vertices have a lower similarity than the non-noise vertices, they smoothly blend into the dunes in the SCOT plot, and hence the MSF's, making the noise difficult to identify. Second, the tendency of clusters to merge is not well-balanced, causing some clusters to merge too much and include the surrounding noise, while others do not merge enough. The combination of these behaviors caused the drop in performance relative to the baseline.

It is interesting to note that although the local density and irregularity objective functions are quite different, the behavior of the substructure optimization algorithm for both objectives is similar for all networks in the study. This fact is encouraging, for the irregularity objective uses only the MSF, which contains at most $n - 1$ edges, while the local density objective and other objectives, such as modularity Q_s , use the original network. Thus, the irregularity objective scales with the number of vertices rather than the number of edges. Ideally, the irregularity objective provides a speedup in substructure optimization of 1 to 2 orders of magnitude over density and modularity objectives, for the networks we examine.

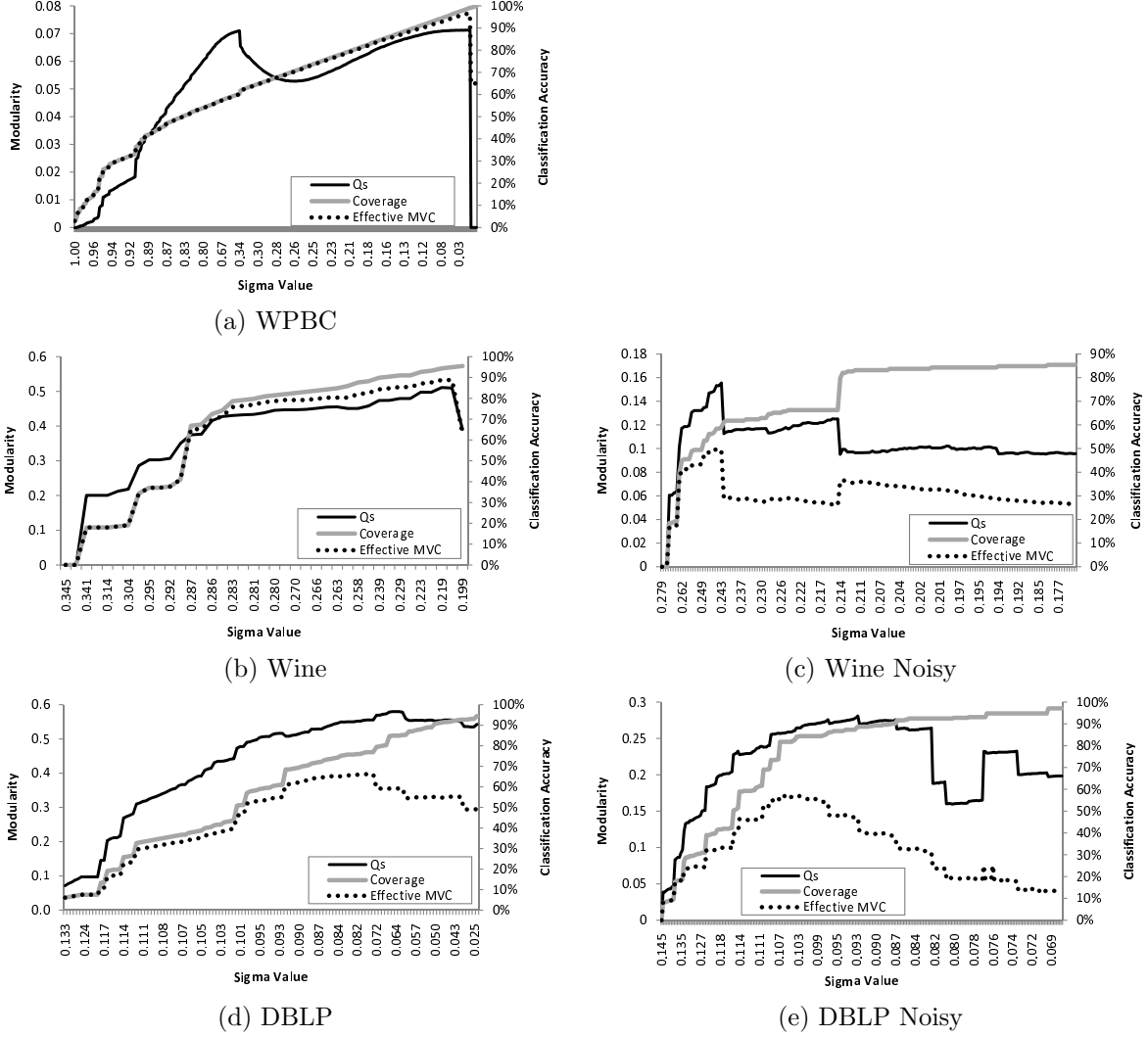


Figure 4.6: Modularity of baseline clustering. The modularity (solid black line) is plotted using the left vertical axis, while the coverage (grey line) and effective MVC (dotted line) are plotted using the right vertical axis.

4.4 Modularity Analysis

Motivated by the positive performance of the baseline algorithm on the noisy networks, we investigate whether the similarity-based modularity function Q_s in (3.1) can be used as an objective function in conjunction with the baseline algorithm. For the μ values reported in Table 4.6, we generate the baseline (G. Den.) clusterings for all ε values and compute the associated modularity. The results are presented in Figure 4.6.

Although the modularity curve is convex for the Wine and DBLP networks, it has concave

regions for the other networks. Since the SCOT plot of the WPBC network exhibits a significantly larger dune for the benign class than for the malignant class, the sudden drop in the modularity near $\varepsilon = 0.34$ is caused when the malignant vertices are included in the clustering. For the Wine noisy network, the first drop is caused by the inclusion of noise nodes, as indicated by the corresponding drop in effective MVP, while the second drop, in conjunction with a sudden increase in coverage, is caused by the inclusion of non-noise nodes after many of the noise nodes are already included. The oscillation of the modularity curve of the DBLP Noisy network is caused by the alternating inclusion of noise and non-noise nodes in separate bushes of the MSF, in which the maximum similarities of the nodes in each class varies noticeably.

Despite the concave portions of the modularity curve for some networks, we ask whether it is possible to use either a global or local maximum modularity to determine the best clustering. Certainly for the WPBC network, this is the case for neither the global maximum nor the first local maximum, since we know that the best clustering is obtained when most of the vertices are covered. In the Wine and Wine Noisy networks, the globally maximum modularity corresponds to the maximum effective MVC. However, for the DBLP and DBLP Noisy networks, the globally maximum modularity is close to the maximum MVC in terms of the change of coverage, yet a significant decrease in effective MVC occurs between the two maxima.

In summary, in two cases the maximum modularity coincides with the maximum effective MVC, in two cases it is somewhat close but is offset enough to cause a significant decrease in effective MVC, and in one case it is not at all close. Overall, the modularity curve seems to take a convex shape if the peak similarity of all the clusters is similar. However, the modularity is somewhat fragile if clusters have different density characteristics or if noise is present. Based on this result, we conclude that that modularity is not likely to provide a reliable objective function to obtain a best clustering in conjunction with the baseline algorithm.

CHAPTER 5

Conclusion and Future Work

Our goal in this paper is to improve over the baseline density-based clustering (DCB) method in cases where no global density threshold produces satisfactory results. We propose a set of new similarity measures based on matrix formulas, which consider paths of length 1 and 2 in the original network. Significantly, the similarity measures induce a new network, which allows the MSF to connect vertices that are not connected in the original network. In our evaluation, we find that some of the unsatisfactory performance of the baseline is improved by our new similarity measures, which amplify the cohesiveness of clusters to classify fewer vertices as noise, enhance the distinction between clusters, and reduce variance of the noise density level. Through these findings we submit that similarity selection should receive more attention in the DBC process, and be treated as an orthogonal component from the clustering algorithm itself.

Our experiments show that the proposed methods perform similarly to the baseline for networks with a few (2–3) classes, while they outperform the baseline for networks with many classes (8). The local density and irregularity objectives show similar performance, despite that the functions are quite different. The performance of the irregularity objective is encouraging, since it scales with the number of vertices and promises a 1 to 2 order-of-magnitude speedup in substructure optimization over the density or modularity objectives, which scale with the number of edges. Further, the α parameter of the irregularity objective is bounded and, as a fraction of the total irregularity, is more intuitive than the parameter for the local density objective.

In the noisy network experiments, the substructure optimization algorithms demonstrated two negative behaviors that can be resolved in future work. We would further like to develop the irregularity objective because of its efficiency. Finally, considering the improvement of the new similarity measures that consider paths longer than one edge in the original network and considering results on multiscale clustering such as in [4], we would like to study similarity measures over longer paths. This is particularly useful in developing similarity measures or reductions for heterogeneous

networks. Matrix computations in heterogeneous graphs offer other areas of improvement, since we only need to calculate the block of the similarity matrix corresponding to vertices of the type being clustered.

APPENDIX

A.1 Derivation of Uniformity

Consider the case in which C has two child clusters with edge counts $n_1, n_2 \geq 1$. We wish to define a function H such that $H(C) = 1$ if $n_1 = n_2$, and $H(C) \rightarrow 0$ as $|n_1 - n_2| \rightarrow \infty$. Since $0 < n_2/n_1 < \infty$, we can interpret the ratio as the slope of a line, such that $\tan(\theta) = n_2/n_1$. Now, since $0 < \theta = \arctan(n_2/n_1) < \pi/2$, then $\sin(2\theta)$ is a feasible function for H . It follows that

$$\begin{aligned} \sin(2\theta) &= \frac{\tan(\theta)}{1 + \tan^2(\theta)} \\ &= 2 \frac{n_1 n_2}{n_1^2 + n_2^2}, \end{aligned}$$

so we can generalize a candidate H' as

$$H'(C) := k \frac{\prod_{i=1}^k n_i}{\sum_{i=1}^k n_i^k}.$$

However, H' is a *holistic aggregate function* because of the variable exponent in the summation, so we modify it to arrive at (3.3).

A.2 Derivation of Joint Probability

The joint probability of two clusterings K and K' is defined in [16] as

$$p(k, k') := \frac{|k \cap k'|}{n},$$

where $k \in K$ and $k' \in K'$ are clusters (sets of vertices), and n is the total number of vertices. This joint probability distribution function can equivalently be defined in terms of a product of cluster

membership matrices

$$P_{KK'|X} := \frac{P_{K|X}^\top P_{K'|X}}{\sum_{k,k'} (P_{K|X}^\top P_{K'|X})_{kk'}} ,$$

where $P_{K|X}$ is the membership matrix with objects in X on rows and clusters in K on columns, where each entry contains 1 if the object is a member of the cluster; otherwise, 0. Since each object is a member of exactly one class, the membership matrix is actually a conditional probability distribution.

We can easily show that $p(\cdot, \cdot)$ is a valid probability distribution function. By definition,

$$\sum_{k,k'} (P_{KK'|X})_{kk'} = 1 .$$

Since an object x belongs to exactly one cluster,

$$\begin{aligned} (P_{K|X}^\top P_{K'|X})_{kk'} &= \sum_{x \in X} p(k | x) p'(k' | x) \\ &= |k \cap k'| . \end{aligned}$$

By total probability,

$$\sum_{k \in K} p(k | x) = 1 = \sum_{k' \in K'} p'(k' | x) ,$$

so that

$$\begin{aligned} \sum_{k,k'} (P_{K|X}^\top P_{K'|X})_{kk'} &= \sum_{k \in K} \sum_{k' \in K'} \sum_{x \in X} p(k | x) p'(k' | x) \\ &= \sum_{x \in X} \sum_{k \in K} p(k | x) \sum_{k' \in K'} p'(k' | x) \\ &= \sum_{x \in X} \sum_{k \in K} p(k | x) \\ &= \sum_{x \in X} 1 \\ &= n . \end{aligned}$$

In this formulation of the joint probability distribution function, the probability of each object x has been normalized so that each occurs with equal probability. In our case, for the DBLP network, we have probabilities $p(x | k)$, where x is a conference and k is a research area. Assuming

that research areas occur with equal probability, we have that

$$p(x, k) = \frac{1}{8}p(x \mid k) .$$

Thus, we can instead define a non-conditional joint probability distribution function

$$p(k, k') := (P_{KK'})_{kk'} ,$$

where

$$P_{KK'} = \frac{P_{XK}^\top P_{XK'}}{\sum_{k, k'} (P_{XK}^\top P_{XK'})_{kk'}}$$

such that $(P_{XK})_{xk} = p(x, k)$, which is known.

REFERENCES

- [1] <http://www.octave.org>.
- [2] Mihael Ankerst, Markus M. Breunig, Hans peter Kriegel, and Jörg S, *Optics: Ordering points to identify the clustering structure*, ACM Press, 1999, pp. 49–60.
- [3] A. Asuncion and D.J. Newman, *UCI machine learning repository*, <http://www.ics.uci.edu/~mllearn/MLRepository.html>, 2007.
- [4] Arik Azran and Zoubin Ghahramani, *Spectral methods for automatic multiscale data clustering*, IEEE Conference on Computer Vision and Pattern Recognition, 2006.
- [5] Dustin Bortner and Jiawei Han, *Progressive clustering of networks using structure-connected order of traversal*, Proc. 2010 Int. Conf. on Data Engineering (ICDE’10), March 2010.
- [6] David Cheng, Ravi Kannan, Santosh Vempala, and Grant Wang, *A divide-and-merge methodology for clustering*, ACM Trans. Database Syst. **31** (2006), no. 4, 1499–1525.
- [7] Aaron Clauset, M. E. J. Newman, and Cristopher Moore, *Finding community structure in very large networks*, (2004).
- [8] Inderjit S. Dhillon, Yuqiang Guan, and Brian Kulis, *Weighted graph cuts without eigenvectors a multilevel approach*, IEEE Trans. Pattern Anal. Mach. Intell. **29** (2007), no. 11, 1944–1957.
- [9] Levent Ertöz, Michael Steinbach, and Vipin Kumar, *Finding clusters of different sizes, shapes, and densities in noisy, high dimensional data*, In Proceedings of Second SIAM International Conference on Data Mining, 2003.
- [10] Martin Ester, Hans peter Kriegel, Jörg S, and Xiaowei Xu, *A density-based algorithm for discovering clusters in large spatial databases with noise*, AAAI Press, 1996, pp. 226–231.

- [11] Zhidan Feng, Xiaowei Xu, Nurcan Yuruk, and Thomas Schweiger, *A novel similarity-based modularity function for graph partitioning*, Data Warehousing and Knowledge Discovery (2007), 385–396.
- [12] M. Girvan and M. E. J. Newman, *Community structure in social and biological networks*, Proceedings of the National Academy of Sciences of the United States of America **99** (2002), no. 12, 7821–7826.
- [13] Jiawei Han, *Data mining: Concepts and techniques*, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2005.
- [14] Jon Kleinberg and Eva Tardos, *Algorithm design*, Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2005.
- [15] Michael Ley, *DBLP*, <http://www.informatik.uni-trier.de/~ley/db/index.html>.
- [16] Marina Meila, *Comparing clusterings by the variation of information*, COLT, 2003, pp. 173–187.
- [17] Andrew Y. Ng, Michael I. Jordan, and Yair Weiss, *On spectral clustering: Analysis and an algorithm*, Advances in Neural Information Processing Systems 14, MIT Press, 2001, pp. 849–856.
- [18] J. Ross Quinlan, *C4.5: programs for machine learning*, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1993.
- [19] J Rual, K Venkatesan, T Hao, T Hirozane-Kishikawa, A Dricot, N Li, G Berriz, F Gibbons, M Dreze, N Ayivi-Guedehoussou, N Klitgord, C Simon, M Boxem, S Milstein, J Rosenberg, D Goldberg, L Zhang, S Wong, G Franklin, S Li, J Albala, J Lim, C Fraughton, E Llamosas, S Cevik, C Bex, P Lamesch, R Sikorski, J Vandenhaute, H Zoghbi, A Smolyar, S Bosak, R Sequerra, L DoucetteStamm, M Cusick, D Hill, F Roth, and M Vidal, *Towards a proteome-scale map of the human protein-protein interaction network*, Nature **437** (2005), no. 7062, 1173–8.
- [20] S. Schaeffer, *Graph clustering*, Computer Science Review **1** (2007), no. 1, 27–64.

- [21] Pang-Ning Tan, Michael Steinbach, and Vipin Kumar, *Introduction to data mining*, 1 ed., Addison Wesley, May 2005.
- [22] Xiaowei Xu, Nurcan Yuruk, Zhidan Feng, and Thomas A. J. Schweiger, *Scan: a structural clustering algorithm for networks*, KDD '07: Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining (New York, NY, USA), ACM, 2007, pp. 824–833.
- [23] Yintao Yu, Cindy X. Lin, Yizhou Sun, Chen Chen, Jiawei Han, Binbin Liao, Tianyi Wu, ChengXiang Zhai, Duo Zhang, and Bo Zhao, *iNextCube: information network-enhanced text cube*, Proc. VLDB Endow. **2** (2009), no. 2, 1622–1625.