Analysis of Gene Expression Data **Spring Semester, 2002**

Lecture 6: May 2, 2002

Lecturer: Ron Shamir *Scribe: Giora Sternberg, and Ron Gabor*[1]

6.1 Clustering Gene Expression Data

6.1.1 Overview and Motivation

In any living cell, that undergoes a biological process, different subsets of its genes are expressed in different stages of the process. The particular genes expressed at a given stage and their relative abundance are crucial to the cell's proper function. Analysis of gene expression patterns can provide an insight into gene/function relationships, effects of medical treatment, biological processes and more.

Clustering techniques applied to gene expression data partition the genes into groups/clusters based on their expression patterns. Genes in the same cluster should have similar expression patterns, while genes in different clusters should have distinct well-separated expression patterns. We will see several examples of gene expression clustering in Section 6.1.4.

Gene expression data can be represented by a real-valued *expression matrix* I, where $I_{i,j}$ is the measured expression level of gene i in experiment j (see Figure 6.1). Experiments can be different time points, different body tissues or different strains of the organism. The i-th row of the expression matrix is called the *expression pattern* or *fingerprint* of gene i.

The similarities between expression patterns of any two genes are represented by a *similarity matrix* S, where $S_{i,j}$ is the similarity value between the fingerprints of gene i and gene j. The similarity matrix can be derived from the expression matrix by applying some similarity measure to the fingerprints of every pair of genes. Similarity measures can be application-dependent. General similarity measures include Euclidean distance and Pearson correlation.

The similarity matrix can be further transformed into a *similarity graph*, G_θ, where the vertices are genes, and there is an edge between two vertices if their similarity is above some threshold θ, that is, $(i,j) \in E(G_\theta)$ iff $S_{i,j} > \theta$.

6.1.2 The Corrupted Clique Graph Model

The clustering problem can be modeled by a corrupted clique graph. A *clique graph* is a graph consisting of disjoint cliques. The true clustering is represented by a clique graph

[1]Based on scribes by Ronny Morad and Tal Moran, January 8, 2001, and by Shachar Ofek and Elena Zotenko, January 15, 2001.

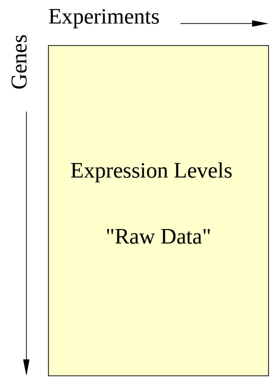


Figure 6.1: Gene expression matrix.

H (vertices are genes and cliques are clusters). Contamination errors introduced into gene expression data result in a similarity graph $C(H)$ which is not a clique graph. Under this model the problem of clustering is as follows: given $C(H)$, restore the original clique graph H and thus the true clustering.

Graph Theoretic Approach

A model for the clustering problem can be reduced to clique graph edge modification problems, stated as follows.

Problem 6.1 *Clique graph editing problem*
INPUT: $G(V, E)$ a graph.
OUTPUT: $Q(V, F)$ a clique graph which minimizes the size of the symmetrical difference between the two edge sets: $|E \Delta F|$.

Clique graph editing problem is NP-hard [14].

Problem 6.2 *Clique graph completion problem*
INPUT: $G(V, E)$ a graph.
OUTPUT: $Q(V, F)$ a clique graph with $E \subseteq F$ which minimizes $|F \setminus E|$.

The clique graph completion problem can be solved by finding all connected components of the input graph and adding all missing edges in each component. Thus the clique graph completion problem is polynomial.

Problem 6.3 *Clique graph deletion problem*
INPUT: $G(V, E)$ a graph.
OUTPUT: $Q(V, F)$ a clique graph with $F \subseteq E$ which minimizes $|E \setminus F|$.

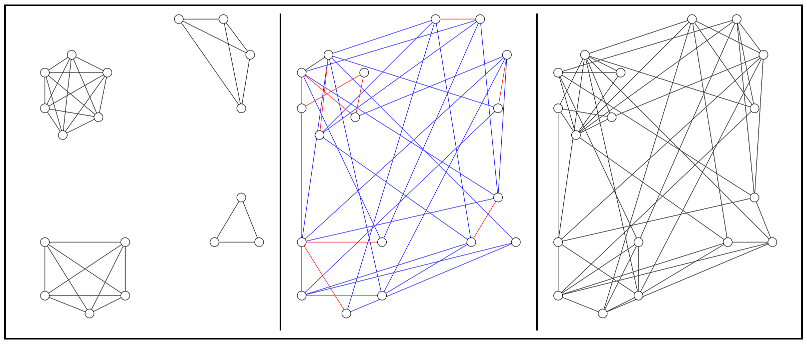


Figure 6.2: The randomly corrupted clique graph model. Left: The original Clustering H of 4 clusters, 18 elements. Middle: Random contamination (flip each edge with a probability $p < 0.5$), red edges denote edges that will be removed, blue edges denote added edges. Right: $G = C(H)$, the input (contaminated) graph.

The clique graph deletion problem is NP-hard [13]. Moreover, any constant factor approximation to the clique graph deletion problem is NP-hard as well [14].

Probabilistic Approach

Another approach is to build a probabilistic model of contamination errors and try to devise an algorithm which, given $C(H)$, reconstructs the original clique graph H with high probability.

One of the simplest probabilistic models for contamination errors is a *random corrupted clique graph*. The contamination errors are represented by randomly removing each edge in the original clique graph H, with probability $p < 0.5$, and adding each edge not in H with the same probability, p (see Figure 6.2). We will denote by $\Omega(H, p)$ the set of all corrupted clique graphs derived from H with contamination error fraction p using this model.

6.1.3 Clustering Algorithm

In this section we present a clustering algorithm of Ben-Dor et. al. [2], called Parallel Classification with Cores (PPC). In subsequent sections we will see the results of applying CAST, a derived practical heuristic, to synthetic data and to real biological data.

We begin with a few definitions.

Definition A *cluster structure* is a vector $(s_1, ..., s_d)$, where each $s_j > 0$ and $\sum s_j = 1$. An n-vertex clique graph has structure $(s_1, ..., s_d)$ if it consists of d disjoint cliques of sizes $ns_1, ..., ns_d$.

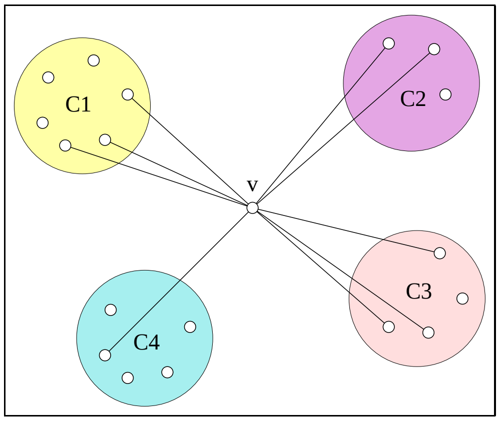


Figure 6.3: Relative Density - the highest relative density of v is with cluster C_3 (relative density of v with clusters C_1, C_2, C_3, C_4 is 1/2, 2/3, 3/4, 1/5, respectively).

Definition A clique graph $H(V,E)$ is called γ-*clustering* (has γ-cluster structure), if the size of each clique in H is at least $\gamma|V|$.

Definition For a fixed $0 < \gamma < 1$ we say that algorithm A *reconstructs* γ-*cluster structures w.h.p.* (with high probability), if for each $0 < \delta < 1$ there exists n_0 such that for each $n \geq n_0$ and for any graph $G \in \Omega(H,p)$, where H is a clique graph with n vertices and γ-cluster structure, $Prob(A(G) \neq H) < \delta$. Formally stated, $\forall\delta > 0\ \exists n_0$ such that $\forall n \geq n_0$ and for any γ-clustering over n vertices, H, we have $Prob(A(C(H,p)) \neq H) < \delta$, where $C(H,p)$ is the random contamination of H (with p probability of flipping), and $A(C(H,p))$ is the output of algorithm A when run on input $C(H,p)$.

Algorithm Idea

Assume that we have already clustered a subset S of vertices. Let us denote their clustering by $\{C_1, ..., C_m\}$. We will extend the clustering $\{C_1, ..., C_m\}$ to include the elements of another set S', by putting each vertex $v \in S'$ into the cluster C, to which it has the highest *relative density* (affinity), that is, the highest ratio between the number of edges connecting v to vertices in C, and the size of C (see Figure 6.3). Formally put, we choose C to be the cluster C_i which maximizes $\frac{|\{u|u\in C_i,(u,v)\in E\}|}{|C_i|}$.

After the extension $\{C_1, ..., C_m\}$ is the clustering of $S \cup S'$. Note that during the extension procedure we do not add new clusters, thus the number m of clusters is unchanged.

Later in this section, we shall show that if the clustering of S is correct then with high probability S' is also clustered correctly.

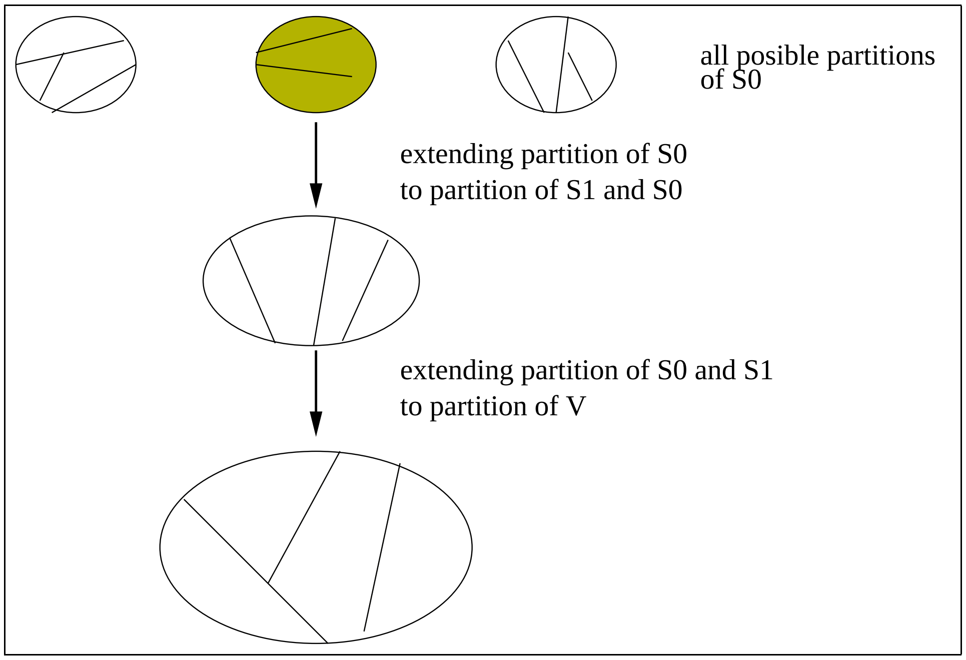

all posible partitions of S0
extending partition of S0 to partition of S1 and S0
extending partition of S0 and S1 to partition of V

Figure 6.4: PP Calgorithm steps shown schematically.

Algorithm Outline

Suppose we are given $G(V, E)$, a corrupted clique graph, that is $G \in \Omega(H, p)$ for some clique graph H with γ-cluster structure. Because H has γ-cluster structure the maximum number of cliques in H is $m = \lceil 1/\gamma \rceil$.

The PPC algorithm will perform the following steps (see Figure 6.4):

1. Uniformly draw $S_0 \subset V$, such that $|S_0| = O(\log\log(n))$;

2. Uniformly draw $S_1 \subset V \backslash S_0$, such that $|S_1| = O(\log(n))$;

3. For each clustering of S_0 into m clusters $\{C_1^0, ..., C_m^0\}$, perform:

   (a) Extend the clustering $\{C_1^0, ..., C_m^0\}$ of S_0 into clustering $\{C_1^1, ..., C_m^1\}$ of $S_0 \cup S_1$;

   (b) Extend the clustering $\{C_1^1, ..., C_m^1\}$ into a clustering $\{C_1, ..., C_m\}$ of V;

4. Each clustering $\{C_1, ..., C_m\}$ of V from the previous step determines a clique graph $C(V, E(C))$; from all such clusterings $\{C_1, ..., C_m\}$ of V, output the one which minimizes $|E \Delta E(C)|$;

Running Time

Running time of the algorithm is dominated by the execution of steps 3 and 4:

- Number of possible partitions of S_0 into m clusters is $m^{|S_0|} = m^{\log\log(n)} = \log^c(n)$;
- For each partition we perform $O(\log(n)\log\log(n))$ operations in step 3a and $O(n\log(n))$ operations in step 3b, thus the overall running time of step 3 is $O(n\log^{c_1}(n))$;
- For each partition we perform $O(n^2)$ operations in step 4, thus the overall running time in step 4 is $O(n^2\log^c(n))$;
- The overall running time of the algorithm is $O(n^2\log^c(n))$;

Algorithm Correctness

Here we will give an outline of the proof. For the details of the proof please refer to [2].

Theorem 6.1 *Chernoff 1952 [3]*
Let $X \sim$ Binomial(n,p). Let $a < p < b$, then:

$$P(X \geq bn) < \exp(-nD(b\|p))$$

$$P(X \leq an) < \exp(-nD(a\|p))$$

Where $D(a\|p)$ is the relative entropy distance *between $(p, 1-p)$ and $(a, 1-a)$, $0 < p, a < 1$, and is defined by:*

$$D(a\|p) = p\log(\frac{p}{a}) + (1-p)\log(\frac{1-p}{1-a})$$

Theorem 6.2 *Let H be a clique graph with γ-cluster structure. For the input graph $G(V,E) \in \Omega(H,p)$ the output $A(G)$ is H w.h.p.*

Proof: (outline)
Consider the partition $\overline{C}^0 = \{\overline{C}^0_1, ..., \overline{C}^0_m\}$ induced by H on S_0:

- Using a simple sampling lemma it can be shown that w.h.p. each cluster of H has at least $\frac{\gamma|S_0|}{2}$ representatives in S_0, thus each non empty cluster of $\overline{C}^0$ has at least $\frac{\gamma|S_0|}{2}$ elements;
- Using the Chernoff bound it can be shown that w.h.p. we extend $\overline{C}^0$ to $\overline{C}^1$ correctly, that is $\overline{C}^1$ is induced by H on $S_0 \cup S_1$;

- Using the sampling lemma it can be shown that w.h.p. each cluster of H has at least $\frac{\gamma|S_1|}{2}$ representatives in S_1, thus each non empty cluster of $\overline{C}^1$ has at least $\frac{\gamma|S_1|}{2}$ elements (assuming that extension from $\overline{C}^0$ to $\overline{C}^1$ was done correctly);
- Using the Chernoff bound it can be shown that w.h.p. we extend $\overline{C}^1$ to $\overline{C}$ correctly, that is $\overline{C}$ is induced by H on V;
- Thus in step 4 of the algorithm we have w.h.p. a correct clustering. To complete the proof we must show that w.h.p. for each other partition C that we have in step 4, $|E \Delta E(\overline{C})| < |E \Delta E(C)|$;

■

Below we will show in some detail a result, which proves the last statement of the above outline.

Theorem 6.3 *Let H be a clique graph with a γ-cluster structure. Then w.h.p. for any $G \in \Omega(H,p)$, H is the closest clique graph to G. That is, for any other clique graph C with a γ-cluster structure, the following holds w.h.p.: $|E(H)\Delta E(G)| < |E(C)\Delta E(G)|$.*

Proof: (outline)
First we will show that given some random clique graph C with a γ-cluster structure we have w.h.p. $|E(H)\Delta E(G)| < |E(C)\Delta E(G)|$:

- A key observation is that C is closer to G iff more than half of the edges in $E(H)\Delta E(C)$ were flipped when generating graph G (please refer to Figure 6.5 for a schematic proof);
- $P(|E(C)\Delta E(G)| < |E(H)\Delta E(G)|) = P(\text{number of edges flipped} \geq \frac{|E(H)\Delta E(C)|}{2})$;
- Using the Chernoff bound it can be shown that $P(|E(C)\Delta E(G)| < |E(H)\Delta E(G)|) < \exp(-|E(H)\Delta E(C)| D(0.5\|p))$;

Now we have to show that for any clique graph C with a γ-cluster structure we have w.h.p. $|E(H)\Delta E(G)| < |E(C)\Delta E(G)|$.

- Consider all clique graphs C with a γ-cluster structure, such that $|E(C)\Delta E(H)| \geq \frac{n\log(n)}{D(0.5\|p)}$. The number of such graphs is m^n where $m = \lceil 1/\gamma \rceil$. Thus, using previous results, it can be easily shown that for any such C w.h.p. $|E(H)\Delta E(G)| < |E(C)\Delta E(G)|$;
- Consider all clique graphs C with a γ-cluster structure, such that $|E(C)\Delta E(H)| < \frac{n\log(n)}{D(0.5\|p)}$, a similar but more complicated argument shows that w.h.p. for any such C $|E(H)\Delta E(G)| < |E(C)\Delta E(G)|$;

■

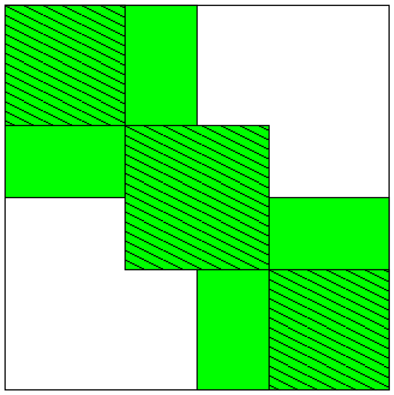


Figure 6.5: Schematic proof of the key observation in Theorem 6.3. This figure depicts a graph G (blank), an original clique graph H (green) and some other clique graph C (green with pattern). Consider $E(H)\Delta E(G)$ and $E(C)\Delta E(G)$. Each edge (u,v) which belongs to neither C nor to H, but belongs to G increments both $E(H)\Delta E(G)$ and $E(C)\Delta E(G)$. Each edge (u,v) which belongs to both C and H, but does not belong to G also increments both $E(H)\Delta E(G)$ and $E(C)\Delta E(G)$. Thus, the difference between $E(H)\Delta E(G)$ and $E(C)\Delta E(G)$ is due to patches that correspond to $E(H)\Delta E(C)$. Let us denote by Δ^* the symmetric difference constrained to edges in $E(C)\Delta E(H)$. Clearly $|E(C)\Delta^* E(G)| + |E(H)\Delta^* E(G)| = |E(H)\Delta E(C)|$. Thus $|E(C)\Delta^* E(G)| < |E(H)\Delta^* E(G)|$ iff $|E(H)\Delta^* E(G)| > \frac{|E(H)\Delta E(C)|}{2}$. Hence, $|E(C)\Delta E(G)| < |E(H)\Delta E(G)|$ iff $|E(H)\Delta^* E(G)| > \frac{|E(H)\Delta E(C)|}{2}$.

Practical Heuristic - CAST

Although the theoretical ideas presented in the previous sections show asymptotic running time complexity of $O(n^2 \log^c n)$, their implementation is still impractical (the constants, for instance, are very large, as in the computation of all possible partitions of S_o into m clusters in step 3). Therefore, based on ideas of the theoretical algorithm, CAST (Cluster Affinity Search Technique), a simple and practical heuristic, was developed. All the tests described in subsequent sections were performed using this practical implementation of the theoretical algorithm.

Let C be a cluster. Let $S_{i,j}$ be a similarity matrix and let $v \in V$ be a gene. We define the *affinity of* v *to cluster* C by $\frac{\sum_{u \in C} S_{u,v}}{|C|}$. Given an *affinity threshold* τ we will say that v is *a close gene to cluster* C if its affinity to C is above τ and we will say that v *is a weak gene in* C if its affinity to C is below τ.

Following are the steps of the practical implementation:

- Construct one cluster at a time and denote it by CC;

- At each step either:
  - Add a close gene to CC;
  - Remove a weak gene from CC;
  - Close CC when no addition or removal is possible;
- Repeat until all genes are clustered;

The main differences between the practical implementation and the theoretical algorithm are:

- In the theoretical algorithm several partitions are formed and then the "best" partition is chosen; in the practical implementation one partition is formed by building one cluster at a time.
- The theoretical algorithm considers the similarity graph, while the practical implementation processes the similarity matrix (the similarity value between any two genes can assume any real value).
- In the theoretical algorithm, the clusters in a partition are extended by adding new elements to them; the practical implementation also allows to remove a weak element from a cluster.

Although little can be proved about the running time and performance of the practical implementation, the test results described in the next sections show that it performs remarkably well, both on simulated data and on real biological data.

6.1.4 Clustering Using BioClust

BioClust is an implementation package of the CAST heuristic. The following section presents results of applying BioClust on both synthetic data and real gene expression data.

Synthetic Data

The simulation procedure is as follows (please refer to Figure 6.6 for visualization of the simulation procedure):

- Let H be the original clique graph.
- Generate G from H by independently removing each edge in H with probability p and adding each edge not in H with probability p.

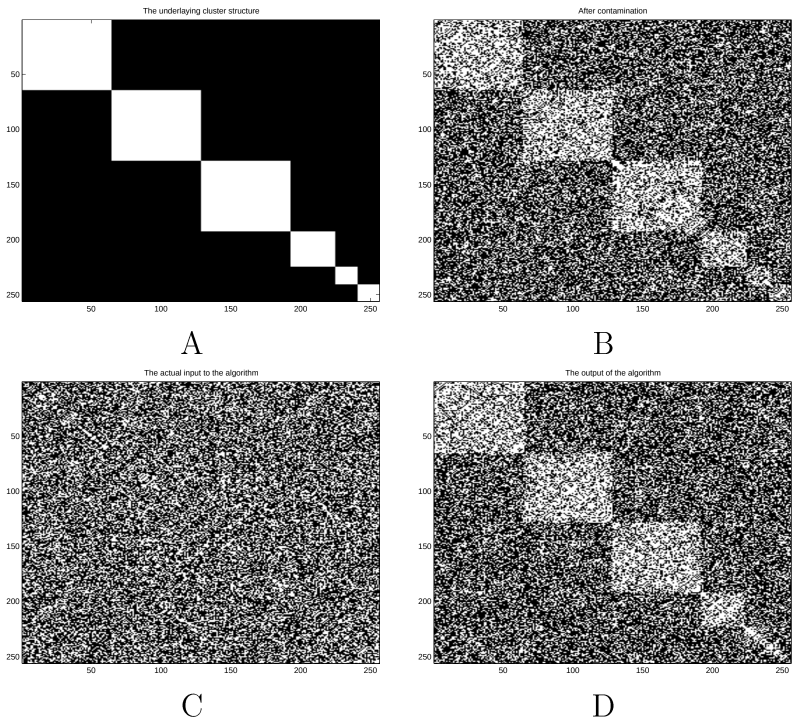


Figure 6.6: Source: [2]. A visualization of the simulation procedure. **A**: The adjacency matrix of the original clique graph H before introduction of errors. Position (i, j) is white if $(i, j) \in E(H)$, that is, if i and j belong to the same cluster. **B**: The same matrix after introduction of errors. Note that the cluster structure is still visible for all but the smallest clusters. **C**: The same as **B** but vertex order is randomly permuted. This is the actual input to the algorithm. **D**: Matrix **C** reordered according to solution produced by the algorithm. With the exception of perhaps the smallest clusters, the essential cluster structure is reconstructed.

- Randomly permute the order of vertices in G and run BioClust with affinity threshold $\tau = 0.5$.

- Compare BioClust's output to the original graph H.

There are several comparison criteria, which can be used to compare the algorithm's output to the original clique graph. Given two adjacency matrices A and B of two graphs of the same size, let N_{ij} be the number of entries on which A and B have values i and j, respectively. The *matching coefficient* is defined by $\frac{N_{00}+N_{11}}{N_{00}+N_{01}+N_{10}+N_{11}}$ that is total number of matching entries divided by total number of entries. The *Jaccard coefficient* is defined by $\frac{N_{11}}{N_{01}+N_{10}+N_{11}}$, which is similar to the matching coefficient, only with N_{00}, the number of entries which are zero in both matrices, removed. In sparse graphs N_{00} will be a dominant

cluster structure	n	p	matching coeff.	Jaccard coeff.
$\{0.4, 0.2, 0.1 \times 4\}$	500	0.2	1.0	1.0
$\{0.4, 0.2, 0.1 \times 4\}$	500	0.3	0.999	0.995
$\{0.4, 0.2, 0.1 \times 4\}$	500	0.4	0.939	0.775
$\{0.1 \times 10\}$	1000	0.3	1.0	1.0
$\{0.1 \times 10\}$	1000	0.35	0.994	0.943

Table 6.1: Performance of BioClust for different values of p and n. Mean values of matching coefficient and Jaccard coefficient are given.

factor, thus Jaccard coefficient is more sensitive when dealing with sparse graphs. With both coefficients, the higher the value, the closer the result is to the real clustering. Both coefficients have maximum value of 1, which implies perfect clustering.

Table 6.1 presents results of simulation for different values of contamination error p and/or number of cluster entities n. The values of the matching coefficient and the Jaccard coefficient are presented. It can be seen that the Jaccard coefficient is more sensitive. One can also observe the effect of p and n on the performance of the algorithm.

Figure 6.7 presents results of simulations for different values of n and p. It can be seen that the properties of the theoretical algorithm are preserved in its practical implementation. We get better performance when the number of clustered entities (vertices in H) increases.

Temporal Gene Expression Data

The gene expression data used in this experiment is from [6]. In this paper the authors study the relationship among expression patterns of genes involved in the rat Central Nervous System (CNS).

Gene expression patterns were measured for 112 genes along 9 different development time points. The gene expression data for each gene was augmented with derivative values to enhance the similarity for closely parallel but offset expression patterns, resulting in a 112×17 expression matrix. The similarity matrix was obtained using Euclidean distance. The execution of BioClust resulted in eight clusters. Since partitioning to clusters is known from [6] this experiment was done mainly for validation of the algorithm.

Figures 6.8 and 6.9 present the clustering results. Note that all clusters, perhaps with the exception of cluster #1, manifest clear and distinct expression patterns. Moreover, the agreement with the prior biological classification is quite good.

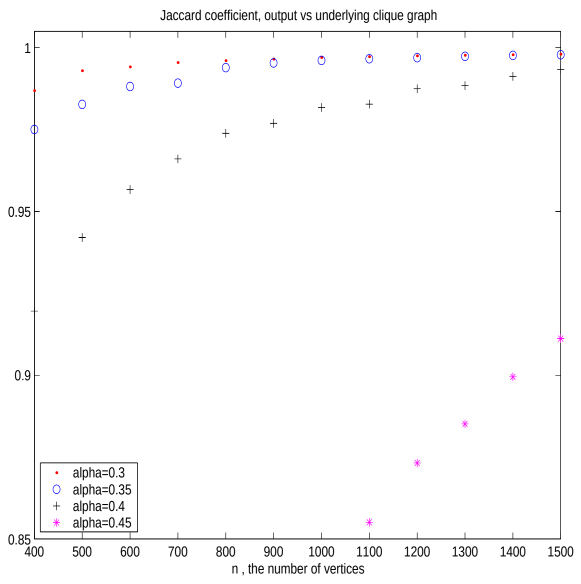


Figure 6.7: Source: [2]. Simulation results for H with cluster structure of $\{\frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \frac{1}{16}, \frac{1}{16}\}$. The x-axis is n, the number of vertices in H (clustered entities), and y-axis is the mean value of the Jaccard coefficient. Each curve corresponds to a specific probability $p = \alpha$ of contamination error.

C. Elegans Gene Expression Data

The gene expression data used in this analysis is from [11]. Kim et al. studied gene regulation mechanisms in the nematode C. Elegans. Gene expression patterns were measured for 1246 genes in 146 experiments, resulting in a 1246×146 expression matrix. The similarity matrix was obtained using Pearson correlation.

The algorithm found 40 clusters. Only very few genes out of the 1246 were classified into families by prior biological studies. The algorithm clustered these families quite well into few homogeneous clusters (see Figure 6.10).

One example of the potential use of clustering for analyzing gene expression patterns is shown in Figure 6.10. A six-gene cluster (cluster #24) contained two growth-related genes and four anonymous genes. This suggests the possibility that the other four genes are also growth-related, paving the way for future biological research.

Tissue Clustering

The gene expression data used in this experiment is from [1]. The authors describe an analysis of gene expression data obtained from 62 samples of colon tissue, 40 tumor and 22 normal tissues. Gene expression patterns were measured for 2000 genes in the 62 samples,

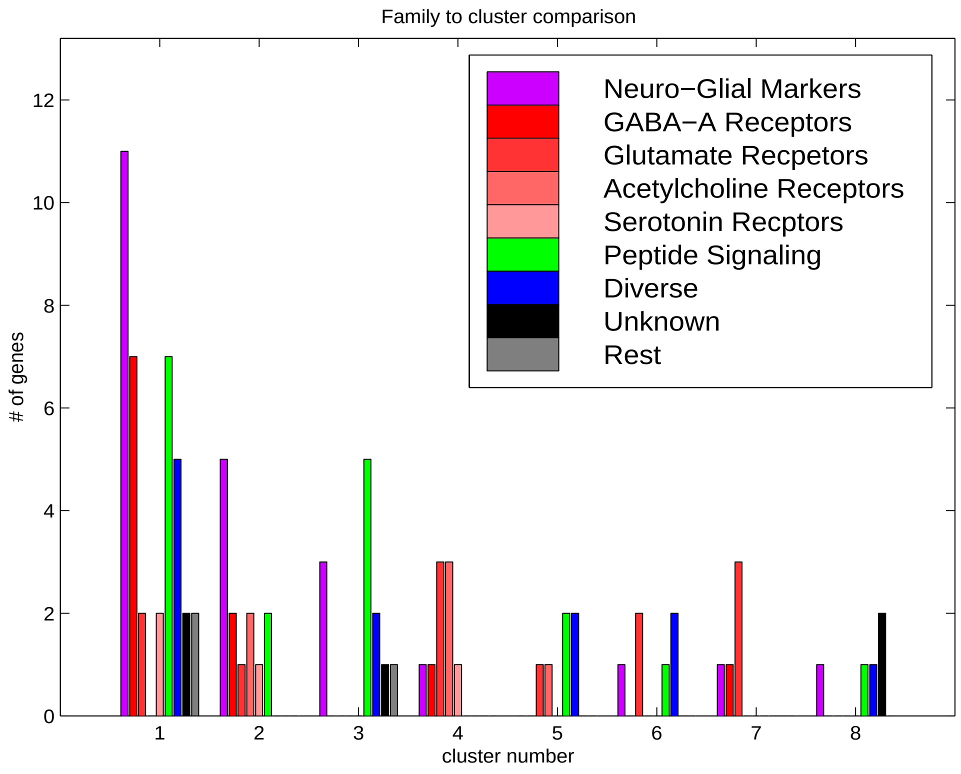


Figure 6.8: Source: [2]. Applying the algorithm to temporal gene expression data [6]. The solution generated by the algorithm is compared to the prior classification. For each cluster (x-axis), bars composition in terms of biologically defined families. The height of each bar (y-axis) represents the number of genes of a specific cluster family. Most clusters contain predominantly genes from one or two families.

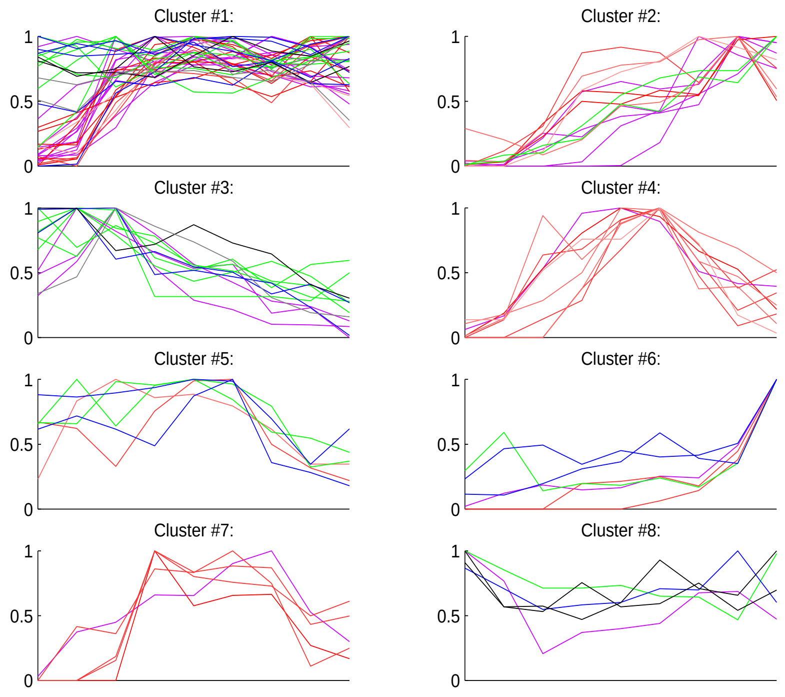
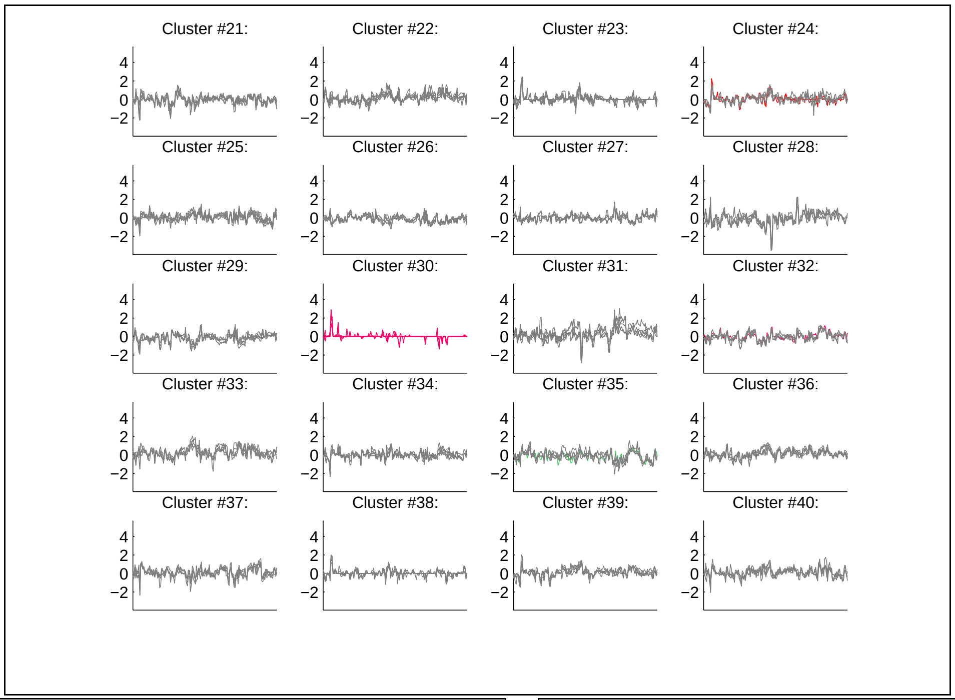
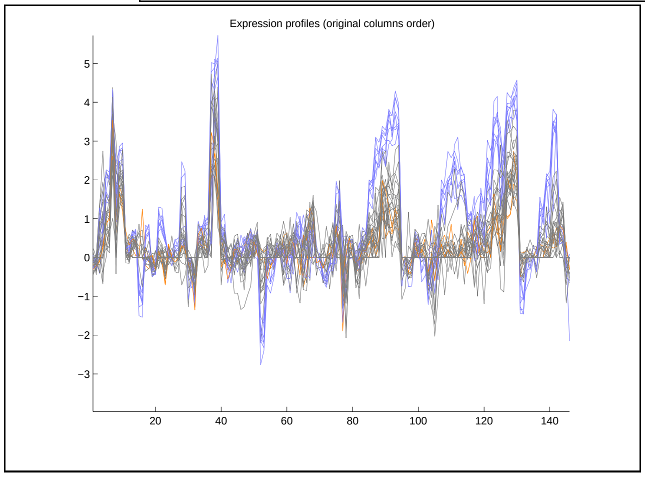


Figure 6.9: Source: [2]. Applying CAST to temporal gene expression data [6]. Each graph presents expression patterns of genes in a specific cluster. The x-axis represents time, while the y-axis represents normalized expression level.





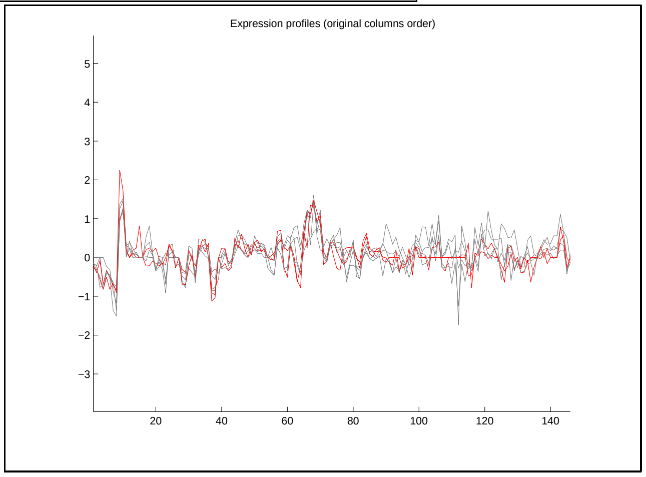


Figure 6.10: Source: [2]. Some results of the CAST algorithm applied to the nematode gene expression data of Kim et. al. Top: expression patterns for clusters #21 to #40. x axis: conditions (matrix columns) in arbitrary order. y axis: intensity level. Most of the genes' functions are unknown, so only few genes are color coded. Blue: sperm genes; red: yeast genes (control) ; gray: unknown. Note the homogeneity of cluster #30. Bottom Left: expression patterns of the genes in cluster #1, consisting of 31 genes. Bottom Right: Expression patterns of the six genes in cluster #24. This cluster contains two growth related genes, lin15 and E2F. This suggests the hypothesis that the other four members of this cluster have related functions.

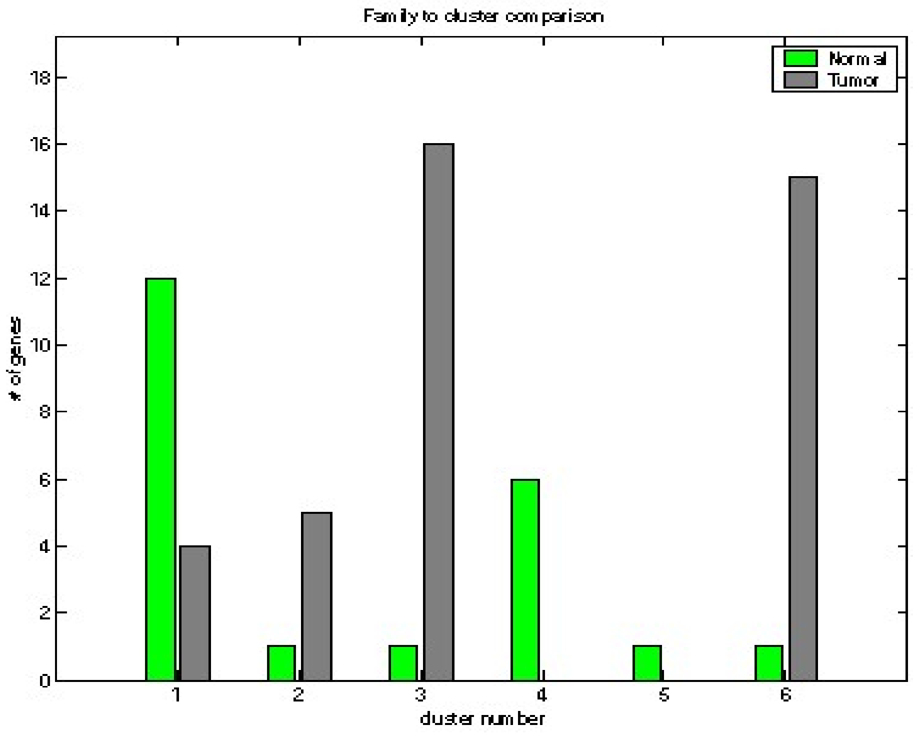


Figure 6.11: Source: [2]. Distribution of tumor and normal tissues in the six clusters produced by the CAST algorithm.

using an Affymetrix chip. The similarity between each two samples was measured using Pearson correlation. Note that here, the similarity is measured between tissues, not genes.

BioClust formed 6 clusters of the data. Figure 6.11 shows the distribution of tumor and normal tissues in the six clusters produced.

The main goal of clustering here is to achieve a separation of tumor and normal tissues. This experiment demonstrates the usefulness of clustering techniques in learning more about the relationship of expression profiles to tissue types.

Improved Theoretical Results

In research soon to be published [15], Tsur & Shamir have introduced a generalized random clique graph model with improved theoretical results, including reduction of the $\Omega(n)$ restriction on cluster sizes, and stronger results when cluster sizes are almost equal.

6.1.5 The CLICK Algorithm

Introduction

CLICK (CLuster Identification via Connectivity Kernels) is a new algorithm for clustering [16]. The input for CLICK is the gene expression matrix. Each row of this matrix is an "expression fingerprint" for a single gene. The columns are specific conditions under which gene expression is measured.

The CLICK algorithm attempts to find a partitioning of the set of elements into clusters, so that two criteria are satisfied: *Homogeneity* - fingerprints of elements from the same cluster, called *mates*, are highly similar to each other; and *Separation* - fingerprints of elements from different clusters, called *non-mates*, have low similarity to each other.

Probabilistic Assumptions

The CLICK algorithm makes the following assumptions:

1. Similarity values between mates are normally distributed with parameters μ_T, σ_T.
2. Similarity values between non-mates are normally distributed with parameters μ_F, σ_F.
3. $\mu_T > \mu_F$.

These assumptions are justified empirically, and in some cases theoretically (by the Central Limit Theorem).

The Basic CLICK Algorithm

The CLICK algorithm represents the input data as a weighted *similarity graph* $G = (V, E)$. In this graph vertices correspond to elements and edge weights are derived from the similarity values. The weight w_{ij} of an edge (i, j) reflects the probability that i and j are mates, and is set to be:

$$w_{ij} = \ln \frac{p_{\text{mates}} f(S_{ij}|i{,}j \text{ are mates})}{(1 - p_{\text{mates}}) f(S_{ij}|i{,}j \text{ are non-mates})}$$

where $f(S_{ij}|i{,}j \text{ are mates}) = f(S_{ij}|\mu_T, \sigma_T)$ is the value of the probability density function for mates at S_{ij}:

$$f(S_{ij}|i{,}j \text{ are mates}) = \frac{1}{\sqrt{2\pi}\sigma_T} e^{-\frac{(S_{ij}-\mu_T)^2}{2\sigma_T^2}}$$

Similarly, $f(S_{ij}|i{,}j$ are non-mates) is the value of the probability density function for non-mates.

The basic CLICK algorithm is described in Figure 6.12 and exemplified in Figure 6.13. The idea behind the algorithm is the following: given a connected graph G, we would like

```
Basic-CLICK($G(V, E)$)
    **if** ($V(G) = \{v\}$) **then**
        move $v$ to the singleton set $R$
    **elseif** ($G$ is a kernel) **then**
        Output $V(G)$
    **else**
        ($H$,$\bar{H}$, $cut$) $\leftarrow$ MinWeightCut($G$)
        Basic-CLICK($H$)
        Basic-CLICK($\bar{H}$)
    **end if**
**end**
```

Figure 6.12: The Basic-CLICK algorithm

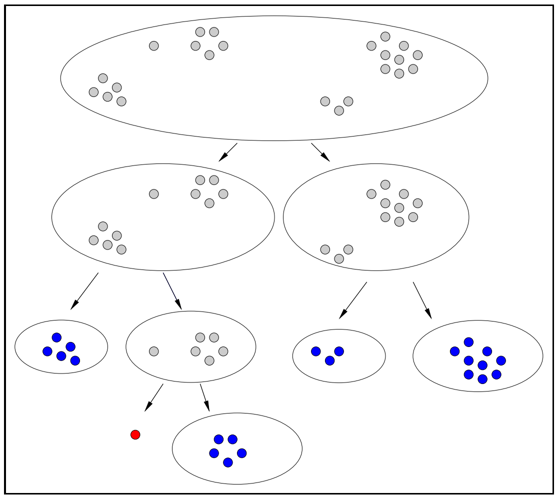


Figure 6.13: Basic scheme of the CLICK algorithm. Split subsets of G, that contain elements from two kernels.

to decide whether $V(G)$ is a subset of some true cluster, or $V(G)$ contains elements from at least two true clusters. In the first case we say that G is *pure*. In order to make this decision, we test the following two hypotheses for each cut C in G:

- H_0^C: C contains only edges between non-mates.
- H_1^C: C contains only edges between mates.

G is declared a *kernel* if H_1 is more probable for all cuts. The decision of whether G is a kernel relies on the following lemma:

Lemma 6.4 *G is a kernel iff $MinWeightCut(G) > 0$.*

Proof: Using Bayes' Theorem, it can be shown that for any cut C in G

$$W(C) = \log \frac{Pr(H_1^C|C)}{Pr(H_0^C|C)}$$

Obviously, $W(C) > 0$ iff $Pr(H_1^C|C) > Pr(H_0^C|C)$. If the minimum cut is positive, then obviously so are all the cuts. Conversely, if the minimum cut is non-positive, then for that cut $Pr(H_1^C|C) \leq Pr(H_0^C|C)$, therefore G is not a kernel. ■

Removing Negative Weight Edges The MIN-CUT problem for a weighted graph with both positive and negative edges is NP-Complete[2]. In order to use the efficient MIN-CUT algorithms we must remove the negative edges. By modifying the algorithm slightly, we can still use the new graph to find kernels in the original graph.

Refinements

The Basic-CLICK algorithm divides the graph into kernels and singletons. These kernels are expanded to the full clustering, using several refinements:

Adoption Step In practice, "true" clusters are usually larger than just the kernel. To accommodate this, in the refined algorithm, kernels "adopt" singletons to create larger clusters. This is done by searching for a singleton v and a kernel K, whose pairwise fingerprint similarity is maximum among all pairs of singletons and kernels. The refined algorithm iteratively applies the adoption step and then the Basic-CLICK algorithm on the remaining singletons, stopping when there are no more changes.

Merge Step In this step we merge clusters whose fingerprints are similar (the justification for this is that, in practice, clusters can contain multiple kernels). The merging is done iteratively, each time merging two clusters whose fingerprint similarity is the highest (provided that the similarity exceeds a predefined threshold).

[2]MIN-CUT can be proved to be NP-Complete by reduction from MAX-CUT [7, page 210].

Quality Assessment

When the "correct" solution for the clustering problem is known, we can evaluate the algorithm's performance using comparison criteria as we have done with BioClust. The criteria used here are the Jaccard coefficient (as defined in the previous sections) and the Minkowski coefficient. The latter is defined by $\sqrt{\frac{n_{01}+n_{10}}{n_{11}+n_{10}}}$. Note that unlike the Jaccard and Matching coefficients, the Minkowski coefficient improves as it decreases, with optimal value at 0.

Unfortunately, in most cases the "correct" solution for the clustering problems is unknown. In these cases we evaluate the quality of the solution by computing two figures of merit to measure the *homogeneity* and *separation* of the produced clusters. For fingerprint data, homogeneity is evaluated by the average and minimum correlation coefficient between the fingerprint of an element and the fingerprint of its corresponding cluster. Separation is evaluated by the weighted average and the maximum correlation coefficient between cluster fingerprints. Formally:

Definition Homogeneity and Separation measures are defined as follows:
We define the fingerprint of a set of elements to be the mean vector of the fingerprints of the members of the set. Let $X_1, ..., X_t$ be clusters, $C(u)$ be the cluster of vertex u, $F(X)$ and $F(u)$ be the fingerprints of a cluster X and of element u respectively, and let $S(x, y)$ denote the similarity between fingerprints x and y, then:

Average Homogeneity

$$H_{Ave} = \frac{1}{|N|} \sum_{u \in N} S(F(u), F(C(u)))$$

Minimum Homogeneity

$$H_{Min} = \min_{u \in N} S(F(u), F(C(u)))$$

Average Separation

$$S_{Ave} = \frac{1}{\sum_{i \neq j} |X_i||X_j|} \sum_{i \neq j} |X_i||X_j| S(F(X_i), F(X_j))$$

Maximum Separation

$$S_{Max} = \max_{i \neq j} S(F(X_i), F(X_j))$$

Logically, a clustering improves when H_{Ave} and H_{Min} increase, and when S_{Ave} and S_{Max} decrease.

Another method of quality assessment is setting a certain similarity threshold and measuring the fraction of mates and non-mates above that threshold. Good clustering is expected to yield higher values of similarity between mates (indicating homogeneity) and lower values between non-mates (indicating separation).

Program (algorithm)	# Clusters	Homogeneity		Separation	
		H_{Ave}	H_{Min}	S_{Ave}	S_{Max}
CLICK	30	0.8	-0.19	-0.07	0.65
GENECLUSTER	30	0.74	-0.88	-0.02	0.97

Table 6.2: A comparison between CLICK and GENECLUSTER [18] on the yeast cell-cycle dataset [4]. Expression levels of 6,218 S. cerevisiae genes, measured at 17 time points over two cell cycles.

6.1.6 Algorithm Performance Comparisons

This section contains examples of comparisons between CLICK and other clustering algorithms, in various problems, including expression data, oligo-fingerprinting data and protein similarity data (Tables 6.2, 6.3, 6.4, 6.5, 6.6 and Figures 6.14, 6.15, 6.16). Analysis of the results (see Table 6.7) shows that CLICK outperforms all the compared algorithms in terms of quality. In addition, CLICK is very fast, allowing clustering of thousands of elements in minutes, and over 100,000 elements in a couple of hours on a regular workstation. Figure 6.17 shows the result of a comparison in which the authors of each clustering algorithm were allowed to run the test on their own. The graph shows a tradeoff between the homogeneity and separation scores; The further the algorithm is from the origin the "better" its overall performance.

In addition, CLICK was tested in simulations which included varying cluster structures and different distribution parameters. Similarity values for mates and non-mates were distributed normally: for each cluster structure, standard deviation σ was set at 5 for both mates and non-mates, while the difference between the means of mates μ_T and non-mates μ_F was set at $t \times \sigma$ for $t = 2, 1, 0.8, 0.6$. Results are shown in Table 6.8, evaluated using the Jaccard coefficient. As expected, the larger the distance between the means of mates and non-mates, the better the performance of the algorithm. It also seems that better results are obtained when cluster sizes are larger.

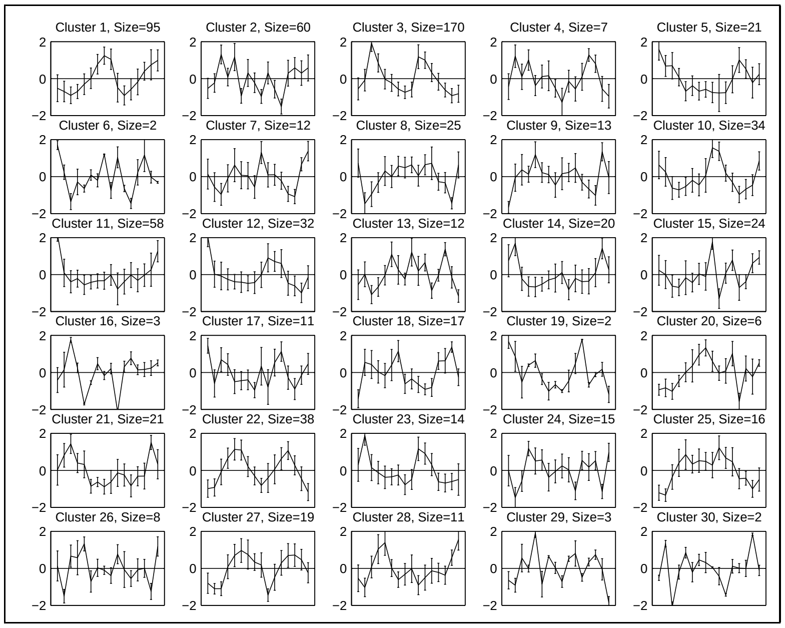


Figure 6.14: Source: [16]. CLICK's clustering of the yeast cell-cycle data [4]. x-axis: time points 0-80, 100-160 at 10-minute intervals. y-axis: normalized expression levels. The solid line in each sub-figure plots the average pattern for that cluster. Error bars display the measured standard deviation. The cluster size is printed above each plot.

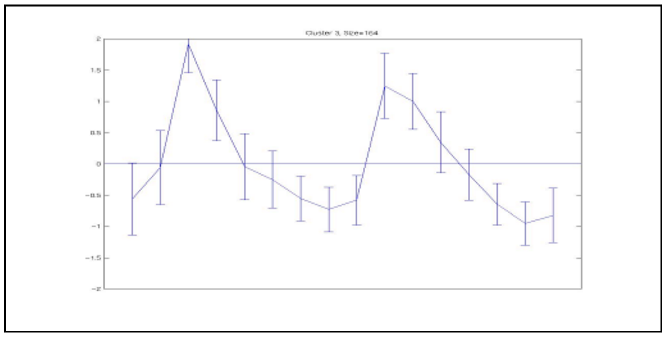


Figure 6.15: Yeast Cell Cycle: late G1 Cluster (cluster 3 from Figure 6.14). The cluster found by CLICK contains 91% of the late G1-peaking genes. In contrast, in GeneCluster 87% are contained in 3 clusters.

Program	#Clusters	#Singletons	Minkowski	Jaccard	Time(min)
CLICK	31	46	0.57	0.7	0.8
HCS	16	206	0.71	0.55	43

Table 6.3: A comparison between CLICK and HCS on the blood monocytes cDNA dataset [8]. 2,329 cDNAs purified from peripheral blood monocytes, fingerprinted with 139 oligos. Correct clustering is known from back hybridization with long oligos.

Program	#Clusters	#Singletons	Minkowski	Jaccard	Time(min)
CLICK	2,952	1,295	0.59	0.69	32.5
K-Means	3,486	2,473	0.79	0.4	–

Table 6.4: A comparison between CLICK and K-Means [9] on the sea urchin cDNA dataset. $20,275$ cDNAs purified from sea urchin eggs, and fingerprinted with 217 oligos. Correct clustering of 1,811 cDNAs is known from back hybridizations.

Program	#Clusters	Homogeneity		Separation	
		H_{Ave}	H_{Min}	S_{Ave}	S_{Max}
CLICK	10	0.88	0.13	-0.34	0.65
Hierarchical	10	0.87	-0.75	-0.13	0.9

Table 6.5: A comparison between CLICK and Hierarchical [5] clustering on the dataset of response of human fibroblasts to serum [10]. Human fibroblast cells starved for 48 hours, then stimulated by serum. Expression levels of 8,613 genes measured at 13 time points.

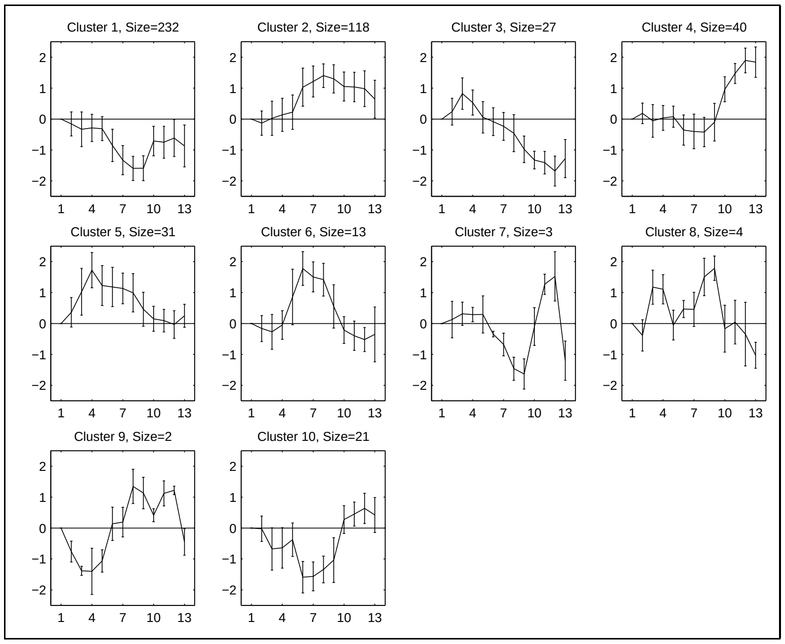


Figure 6.16: Source: [16]. CLICK's clustering of the fibroblasts serum response data [10]. x-axis: 1-12: synchronized time-points. 13: unsynchronized point. y-axis: normalized expression levels. The solid line in each sub-figure plots the average pattern for that cluster. Error bars display the measured standard deviation. The cluster size is printed above each plot.

Program	#Clusters	#Singletons	Homogeneity	Separation	Time(min)
CLICK	9,429	17,119	0.24	0.03	126.3
SYSTERS	10,891	28,300	0.14	0.03	–

Table 6.6: A comparison between CLICK and SYSTERS on a dataset of 117,835 proteins [12]. Measures based on similarity when no correct solution is known: For a fixed threshold t, homogeneity is the fraction of mates with similarity above t, and separation is the fraction of non-mates with similarity above t.

Elements	Problem	Compared to	Improvement	Time(min)
517	Gene Expression Fibroblasts	Cluster [5]	Yes	0.5
826	Gene Expression Yeast cell cycle	GeneCluster [18]	Yes	0.2
2,329	cDNA OFP Blood Monocytes	HCS [8]	Yes	0.8
20,275	cDNA OFP Sea urchin eggs	K-Means [9]	Yes	32.5
72,623	Protein similarity	ProtoMap [19]	Minor	53
117,835	Protein similarity	SYSTERS [12]	Yes	126.3

Table 6.7: A Summary of the time performance of CLICK on the above mentioned datasets. CLICK was executed on an SGI ORIGIN200 machine utilizing one IP27 processor. The time does not include preprocessing time. The "Improvement" column describes whether the solution of the CLICK algorithm was better than the compared algorithm.

Ataxia Telangiectasia

The following experiment shows how clustering methods can aid our understanding of biological processes. Its aim was to study the expression patterns of a genetic disease, Ataxia Telangiectasia (A-T).

A-T is a rare autosomal recessive disorder, characterized by cerebellar and thymic degeneration and predisposition to cancer. The gene found to be responsible for A-T is ATM - an important mediator of cell responses to DNA damage, in particular those that control

Structure	2	1	0.8	0.6
6×60	1	1	0.98	0.85
10×30	1	0.96	0.71	0.1
$10, ..., 80$	1	1	0.97	0.83

Table 6.8: CLICK simulation results (mean Jaccard score over 20 runs). The test included various cluster structures (rows) and distances between μ_T and μ_F (in each column, the distance appearing in the title was used as a factor of the standard deviation σ. The first column denotes a distance of 2σ, etc.).

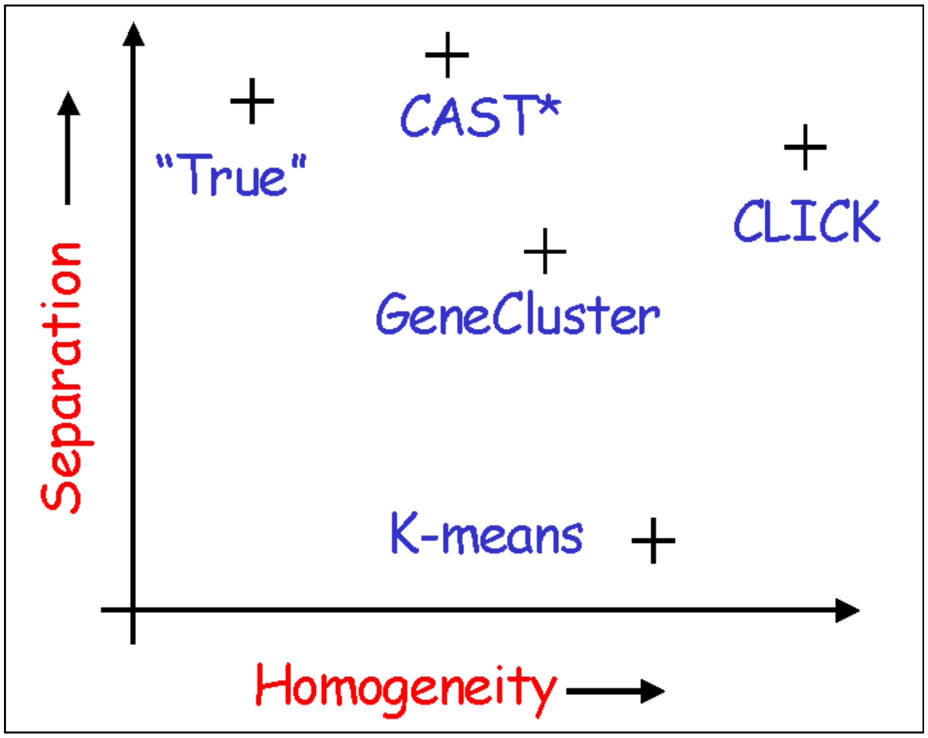


Figure 6.17: Comparison of clustering algorithms using homogeneity and separation criteria. The data consisted of 698 genes, 72 conditions [17]. Each algorithm was run by its authors in a "blind" test.

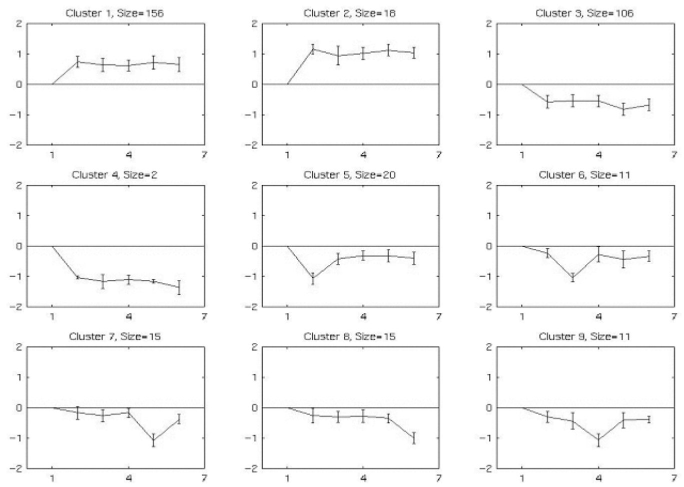


Figure 6.18: Clusters generated by the CLICK algorithm for the A-T experiment. y-axis: normalized expression levels. x-axis: 1-3: normal mice at times 0, 30, 120 (minutes); 4-6: ATM-deficient mice at times 0, 30, 120. Note that in cluster #1 we see constitutive expression of the genes in ATM-deficient mice at all three time-points, whereas in normal mice expression rises to about that level only after irradiation.

progression through the cell cycle. The study of A-T is facilitated by the existence of a mouse homologue of ATM.

In this experiment, DNA damage was induced both in normal mice and in ATM-deficient mice via irradiation. Tissues were then obtained in 3 time-points - 0, 30 and 120 minutes after irradiation - in order to check for differences in the cellular response. cDNA's from thymus, cerebellum and brain were hybridized to microarrays representing 8,000 transcripts. Each gene was sampled at 18 combinations of tissue, genotype and time-point. The resulting data was then processed by the CLICK algorithm.

The results are shown in Figure 6.18. Interestingly, some of the clusters showed differences between the normal and ATM-deficient samples. These differences included constitutive expression of genes in ATM-deficient mice: whereas in normal mice a rise in the expression level of these genes was noted in the time-points after 0, in ATM-deficient mice the level was high already at 0 time, and did not change significantly afterwards. These results may suggest that these genes are involved in the response to the induced DNA damage, which is hampered in ATM-deficient mice.

6.1.7 Clustering - Future Research

Clustering pops up in many questions and problems in various fields of interest, not only gene clustering. In many practical situations, the complexity of the clustering is the dominant part of the complexity of the whole solution. In many cases, though, clustering heuristics may not perform well and produce poor clustering results.

Therefore there is still room for improvement, both in theoretical clustering algorithms (and proofs), and in practical algorithms and heuristics. Some of the possible avenues for progress are indicated below.

Theoretical Clustering

- Reducing the restrictions on cluster sizes.
- Novel approaches and better algorithms, improving on both quality of results and time-complexity.

Practical Clustering

- Determining which algorithms and heuristics are best suited for particular clustering jobs.
- Improving on the optimization criteria.
- Better control over the tradeoff of accuracy versus speed, for instance using plausible assumptions about the clustered entities in each specific implementation.
- Devising new clustering/viewing tools, which would allow interactive setting of parameters and modifying algorithm behavior, while viewing the results.

Bibliography

[1] U. Alon, N. Barkai, D. A. Notterman, G. Gish, S. Ybarra, D. Mack, and A. J. Levine. Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. *PNAS*, 96:6745–6750, June 1999.

[2] A. Ben-Dor, R. Shamir, and Z. Yakhini. Clustering gene expression patterns. *Journal of Computational Biology*, 6(3/4):281–297, 1999.

[3] H. Chernoff. A measure of the asymptotic efficiency for tests of a hypothesis based on the sum of observations. *Annals of Mathematical Statistics*, 23:493–509, 1952.

[4] RJ. Cho et al. A genome-wide transcriptional analysis of the mitotic cell cycle. *Mol Cell*, 2:65–73, 1998.

[5] M. B. Eisen, P. T. Spellman, P. O. Brown, and D. Botstein. Cluster analysis and display of genome-wide expression patterns. *PNAS*, 95:14863–14868, 1998.

[6] X. Wen et. al. Large-scale temporal gene expression mapping of central nervous system development. *PNAS*, 95:334 – 339, 1998.

[7] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman and Co., San Francisco, 1979.

[8] E. Hartuv, A. Schmitt, J. Lange, S. Meier-Ewert, H. Lehrach, and R. Shamir. An algorithm for clustering cDNA fingerprints. *Genomics*, 66(3):249–256, 2000.

[9] R. Herwig, A.J. Poustka, C. Muller, C. Bull, H. Lehrach, and J. O'Brien. Large-scale clustering of cDNA-fingerprinting data. *Genome Research*, 9:1093–1105, 1999.

[10] V.R. Iyer, M.B. Eisen, D.T. Ross, G. Schuler, T. Moore, J.C.F. Lee, J.M. Trent, L.M. Staudt, J. Hudson Jr., M.S. Boguski, D. Lashkari, D. Shalon, D. Botstein, and P.O. Brown. The transcriptional program in the response of human fibroblasts to serum. *Science*, 283 (1), 1999.

[11] S. Kim. Department of Developmental Biology, Stanform University, `http://cmgm.stanford.edu/∼kimlab/`.

[12] A. Krause, J. Stoye, and M. Vingron. The systers protein sequence cluster set. *Nucleic Acids Research*, 28(1):270–272, 2000.

[13] A. Natanzon. Complexity and approximation of some graph modification problems. Master's thesis, Department of Computer Science, Tel Aviv University, 1999.

[14] R. Sharan R. Shamir and D. Tsur. Clustering graph modification problems. In *28th International Workshop on Graph Theoretical Concepts in Computer Science*, 2002. To appear.

[15] R. Shamir and D. Tsur. Improved algorithms for the random cluster graph model. In *8th Scandinavian Workshop on Algorithm Theory.*, 2002. To appear.

[16] R. Sharan and R. Shamir. Click: A clustering algorithm with applications to gene expression analysis. In *Proceedings of the 8th Annual International Conference on Intelligent Systems for Molecular Biology, (ISMB '00)*, pages 307–316. AAAI Press, 2000.

[17] P. T. Spellman, G. Sherlock, et al. Comprehensive identification of cell cycle-regulated genes of the yeast Saccharomyces cerevisiae by microarray hybridization. *Mol. Biol. Cell*, 9:3273–3297, 1998.

[18] P. Tamayo, D. Slonim, J. Mesirov, Q. Zhu, S. Kitareewan, E. Dmitrovsky, E. S. Lander, and T.R. Golub. Interpreting patterns of gene expression with self-organizing maps: Methods and application to hematopoietic differentiation. *PNAS*, 96:2907–2912, 1999.

[19] G. Yona, N. Linial, and M. Linial. Protomap: Automatic classification of protein sequences, a hierarchy of protein families, and local maps of the protein space. *Proteins: Structure, Function, and Genetics*, 37:360–378, 1999.