CLICK and EXPANDER: A System for Clustering and Visualizing Gene Expression Data

Roded Sharan*[†] Adi Maron-Katz [‡] Ron Shamir [‡]

[†] International Computer Science Institute, 1947 Center St., Suite 600, Berkeley CA 94704-1198, USA. Email: roded@icsi.berkeley.edu.
[‡] School of Computer Science, Tel-Aviv University, Tel-Aviv 69978, Israel. Email: {adimaron,rshamir}@tau.ac.il.

Running head: EXPANDER: Clustering Gene Expression Data.

*To whom correspondence should be addressed.

Abstract

Motivation: Microarrays have become a central tool in biological research. Their applications range from functional annotation to tissue classification and genetic network inference. A key step in the analysis of gene expression data is the identification of groups of genes that manifest similar expression patterns. This translates to the algorithmic problem of clustering genes based on their expression patterns.

Results: We present a novel clustering algorithm, called CLICK, and its applications to gene expression analysis. The algorithm utilizes graph-theoretic and statistical techniques to identify tight groups (kernels) of highly similar elements, which are likely to belong to the same true cluster. Several heuristic procedures are then used to expand the kernels into the full clusters. We report on the application of CLICK to a variety of gene expression datasets. In all those applications it outperformed extant algorithms according to several common figures of merit. We also point out that CLICK can be successfully used for the identification of common regulatory motifs in the upstream regions of co-regulated genes. Furthermore, we demonstrate how CLICK can be used to accurately classify tissue samples into disease types, based on their expression profiles. Finally, we present a new java-based graphical tool, called EXPANDER, for gene expression analysis and visualization, which incorporates CLICK and several other popular clustering algorithms.

Availability: `http://www.cs.tau.ac.il/∼rshamir/expander/expander.html`.

Contact: `roded@icsi.berkeley.edu`.

1 Introduction

Microarray technology has become a central tool in biological and biomedical research. This technology provides a global, simultaneous view on the transcription levels of many or all genes of an organism under a range of conditions or processes. The information obtained by monitoring gene expression levels in different developmental stages, tissue types, clinical conditions and different organisms can help in understanding gene function and gene networks, assist in the diagnostic of disease conditions and reveal the effects of medical treatments.

A key step in the analysis of gene expression data is the identification of groups of genes that manifest similar expression patterns. This translates to the algorithmic problem of clustering gene expression data. A clustering problem usually consists of elements and a characteristic vector for each element. A measure of similarity is defined between pairs of such vectors. (In gene expression, elements are usually genes, the vector of each gene contains its expression levels under each of the monitored conditions, and similarity can be measured, for example, by the correlation coefficient between vectors.) The goal is to partition the elements into subsets, which are called *clusters*, so that two criteria are satisfied: *Homogeneity* - elements in the same cluster are highly similar to each other; and *separation* - elements from different clusters have low similarity to each other.

There is a very rich literature on cluster analysis going back over three decades (cf. (Hartigan, 1975; Everitt, 1993; Mirkin, 1996; Hansen & Jaumard, 1997)). Several algorithmic techniques were previously used in clustering gene expression data, including hierarchical clustering (Eisen *et al.*, 1998), self organizing maps (Tamayo *et al.*, 1999), K-means (Herwig *et al.*, 1999), simulated annealing (Alon *et al.*, 1999), and graph theoretic approaches: HCS (Hartuv & Shamir, 2000) and CAST (Ben-Dor *et al.*, 1999).

We have developed a novel clustering algorithm that we call CLICK (CLuster Identification via Connectivity Kernels). The algorithm does not make any prior assumptions on the number of clusters or their structure. At the heart of the algorithm is a process of recursively partitioning a weighted graph into components using minimum cut computations. The edge weights and the stopping criterion of the recursion are assigned probabilistic meaning, which gives the algorithm high accuracy. The speed of the algorithm is achieved by a variety of experimentally tested heuristic procedures that shortcut, prepend and append the main process.

CLICK was implemented and tested on a variety of biological datasets. On three large-scale gene expression datasets the algorithm outperformed previously published results, that utilized hierarchical clustering and self organizing maps. We also show the utility of CLICK in more advanced biological analyses: The identification of common regulatory motifs in the promoters of co-regulated genes, and the classification of samples into disease types based on their expression profiles. In the latter problem CLICK achieved success ratios of over 90% on two real datasets.

We present a new java-based graphical tool, called EXPANDER (EXPression ANalyzer and DisplayER), for gene expression analysis and visualization. This software contains several clustering methods including CLICK, K-Means, hierarchical clustering and self organizing maps, all controlled via a graphical user interface. It enables visualizing the raw expression data and the clustered data in several ways, as well as single-cluster and all-clusters evaluations via fitness scores and functional enrichment tests.

A preliminary version of this manuscript, containing an early version of CLICK and some initial tests, has appeared in (Sharan & Shamir, 2000).

2 Preliminaries

Let $N = \{e_1, \ldots, e_n\}$ be a set of n elements, and let $\mathcal{C} = (C_1, \ldots, C_l)$ be a partition of N into subsets. Each subset is called a *cluster*, and $\mathcal{C}$ is called a *clustering solution*, or simply a *clustering*. Two elements e_i and e_j are called *mates with respect to* $\mathcal{C}$ if they are members of the same cluster in $\mathcal{C}$. In the gene expression context, the elements are the genes and we often assume that there exists some correct partition of the genes into "true" clusters. When $\mathcal{C}$ is the true clustering of N, elements that belong to the same true cluster are simply called *mates*.

The input data for a clustering problem is typically given in one of two forms: (1) *Fingerprint data* – each element is associated with a real-valued vector, called its *fingerprint*, or *pattern*, which contains p measurements on the element, e.g., expression levels of the gene's mRNA at different conditions (cf. (Eisen & Brown, 1999)). (2) *Similarity data* – pairwise similarity values between elements. These values can be computed from fingerprint data, e.g., by correlation between vectors. Alternatively, the data can represent pairwise dissimilarity, e.g., by computing distances. Fingerprints contain more information than similarity data, but the latter is completely generic and can be used to represent the input to clustering in any application. Note that there is also a practical consideration regarding the presentation: The fingerprint matrix is of order $n \times p$ while the similarity matrix is of order $n \times n$, and in gene expression applications often $n \gg p$.

The goal in a clustering problem is to partition the set of elements into homogeneous and well-separated clusters. That is, we require that elements from the same cluster will be highly similar to each other, while elements from different clusters will have low similarity to each other. Note that this formulation does not define a single optimization problem: Homogeneity and separation can be defined in various ways, leading to a variety of optimization problems (cf. (Hansen & Jaumard, 1997)). Even when the homogeneity and separation are precisely defined, those two objectives are typically conflicting: The higher the homogeneity – the lower the separation, and vice versa.

For two elements x and y, we denote the similarity of their fingerprints by $S(x, y)$. We say that a symmetric similarity function S is *linear* if for any three vectors u, v, and w, we have $S(u, v + w) = S(u, v) + S(u, w)$. For example, vector dot-product is a linear similarity function.

Judicious preprocessing of the raw data is key to meaningful clustering. This preprocessing is application dependent and must be chosen in view of the expression technology used and the biological questions asked. The goal of the preprocessing is to normalize the data and calculate the pairwise element (dis)similarity, if applicable. Common procedures for normalizing fingerprint data include transforming each fingerprint to have mean zero and variance one, a fixed norm or a fixed maximum entry. Statistically based methods for data normalization have also been developed recently (see, e.g., (Kerr *et al.*, 2000; Yang *et al.*, 2002)).

2.1 Assessment of Solutions

A key question in the design and analysis of clustering techniques is how to evaluate solutions. We present here figures of merit for measuring the quality of a clustering solution. Different measures are applicable in different situations, depending on whether a partial true solution is known or not, and whether the input is fingerprint or similarity data. For other possible figures of merit we refer the reader to (Everitt, 1993; Hansen & Jaumard, 1997; Yeung *et al.*, 2001).

Suppose at first that the true solution is known, and we wish to compare it to a suggested solution $\mathcal{C}$. The *Jaccard coefficient* (cf. (Everitt, 1993)) is defined as the proportion of correctly identified mates in $\mathcal{C}$ to the sum of correctly identified mates plus the total number of disagreements between $\mathcal{C}$ and the true solution (a disagreement is a pair which are mates in one solution and non-mates in the other). Hence, a perfect solution has score one, and the higher the score – the better the solution. This measure is a lower bound for both the sensitivity and the specificity of the suggested solution.

When the true solution is unknown, we evaluate the quality of a suggested solution by computing two figures of merit that measure its homogeneity and separation. We define the *homogeneity of a cluster* as the average similarity between its members, and the *homogeneity of a clustering* as the average similarity between mates (with respect to the clustering). Precisely, if $F(i)$ is the fingerprint of element i and the total number of mate pairs is M then:

$$H_{Ave} = \frac{1}{M} \sum_{i,j \text{ are mates}, i<j} S(F(i), F(j)) \ .$$

Similarly, we define the *separation of a clustering* as the average similarity between non-mates:

$$S_{Ave} = \frac{2}{n(n-1) - 2M} \sum_{i,j \text{ are non-mates}, i<j} S(F(i), F(j)) \ .$$

Related measures that take a worst case instead of average case approach are *minimum cluster homogeneity*:

$$H_{Min} = \min_{C \in \mathcal{C}} \frac{\sum_{i,j \in C, i<j} S(F(i), F(j))}{\binom{|C|}{2}}$$

and *maximum average similarity* between two clusters:

$$S_{Max} = \max_{C,C' \in \mathcal{C}} \frac{\sum_{i \in C, j \in C'} S(F(i), F(j))}{|C||C'|} \ .$$

Hence, a solution improves if H_{Ave} or H_{Min} increase, and if S_{Ave} or S_{Max} decrease. In computing all the above measures, singletons are considered as one-member clusters. Note that for fingerprint data and a linear similarity function, H_{Ave} and S_{Ave} can be computed in $O(np)$ time (Sharan, 2002).

The two types of measures, intra-cluster homogeneity and inter-cluster separation, are inherently conflicting, as an improvement in one will typically correspond to worsening of the other. There are several approaches that address this difficulty. One approach is to fix the number of clusters and seek a solution with maximum homogeneity. This is done for example by the classical K-means algorithm. For methods to evaluate the number of clusters see, e.g., (Hartigan, 1975; Tibshirani *et al.*, 2000; Ben-Hur *et al.*, 2002; Pollard & van der Laan, 2002). Another approach is to present a curve of homogeneity vs. separation over a range of parameters for the clustering algorithm used (A. Ben-Dor, private communication). For yet another approach for comparing solutions across a range of parameters, see (Yeung *et al.*, 2001).

3 The CLICK Algorithm

In this section we present a novel clustering algorithm, which we call CLICK (CLuster Identification via Connectivity Kernels). The algorithm builds on the HCS algorithm of Hartuv and Shamir (2000) . It utilizes graph-theoretic and statistical techniques to identify tight groups (kernels) of highly similar elements, which are likely to belong to the same true cluster. Several heuristic procedures are then used to expand the kernels into the full clusters.

3.1 The Probabilistic Framework

A key modeling assumption in developing CLICK is that pairwise similarity values between elements are normally distributed: Similarity values between mates are normally distributed with mean μ_T and variance σ_T^2, and similarity values between non-mates are normally distributed with mean μ_F and variance σ_F^2, where $\mu_T > \mu_F$. This situation was observed on simulated and real data and can be theoretically justified under certain conditions by the Central Limit Theorem (Sharan, 2002). Another modeling parameter is p_{mates}, the probability that two randomly chosen elements are mates. We denote by $f(x|\mu_T, \sigma_T)$ the *mates probability density function*. We denote by $f(x|\mu_F, \sigma_F)$ the *non-mates probability density function*.

An initial step of the algorithm is estimating the parameters $\mu_T, \mu_F, \sigma_T, \sigma_F$ and p_{mates}, using one of two methods: (1) In many cases the true partition for a subset of the elements is known. This is the case, for example, if the clustering of some of the genes in a cDNA oligo-fingerprint experiment is found experimentally (see, e.g., (Hartuv *et al.*, 2000)), or more generally, if a subset of the elements has been analyzed using prior biological knowledge (see, e.g., (Spellman *et al.*, 1998)). Based on this partition one can compute the sample mean and sample variance for similarity values between mates and between non-mates, and use these as maximum likelihood estimates for the distribution parameters. The proportion of mates among all known pairs can serve as an estimate for p_{mates}, if the subset was randomly chosen. (2) In case no additional information is given, these parameters can be estimated using the EM algorithm (Sharan, 2002).

3.2 The Basic CLICK Algorithm

The CLICK algorithm works in two phases. In the first phase tightly homogeneous groups of elements, called kernels, are identified. In the second phase these kernels are expanded to the final clusters. In this section we describe the kernel identification step.

The input to this phase is a matrix S of similarity values, where S_{ij} is the similarity value between elements e_i and e_j. When the input is fingerprint data, a preprocessing step computes all pairwise similarity values between elements, using a given similarity function. The algorithm represents the input data as a weighted *similarity graph* $G = (V, E, w)$. In this graph vertices correspond to elements and edge weights are derived from the similarity values. Note that G is a complete graph. The weight w_{ij} of an edge (i, j) reflects the probability that i and j are mates, and is set to be

$$w_{ij} = \log \frac{Pr(i, j \text{ are mates}|S_{ij})}{Pr(i, j \text{ are non-mates}|S_{ij})} = \log \frac{p_{mates} f(S_{ij}|\mu_t, \sigma_T)}{(1 - p_{mates}) f(S_{ij}|\mu_F, \sigma_F)} \ .$$

Here $f(S_{ij}|i, j \text{ are mates}) = f(S_{ij}|\mu_T, \sigma_T)$ is the value of the mates probability density function at S_{ij}. Similarly, $f(S_{ij}|i, j \text{ are non-mates})$ is the value of the non-mates probability density function at S_{ij}. Hence,

$$w_{ij} = \log \frac{p_{mates}\sigma_F}{(1 - p_{mates})\sigma_T} + \frac{(S_{ij} - \mu_F)^2}{2\sigma_F^2} - \frac{(S_{ij} - \mu_T)^2}{2\sigma_T^2} \ .$$

The Basic-CLICK algorithm can be described recursively as follows: Assume temporarily that all edge weights are non-negative. Initially, all elements are active. In each step the algorithm handles some connected component of the subgraph induced by the active elements. If the component contains a single vertex, then this vertex is considered a *singleton* and is inactivated. Otherwise, a stopping criterion (which will be described later) is checked. If the component satisfies the criterion, it is declared a *kernel* and inactivated. Otherwise, the component is split according to a minimum weight cut (a set of edges of minimum total weight, whose removal would disconnect the graph). The algorithm outputs a list of kernels which serves as a basis for the eventual clusters, and a list of singletons.

The idea behind the algorithm is the following. Given a connected graph $G = (V, E)$, we would like to decide whether V is a subset of some true cluster, or V contains elements from at least two true clusters. In the former case we say that G is *pure*. In order to make this decision we test for each cut C in G the following two hypotheses:

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

We let $Pr(H_i^C|C)$ denote the posterior probability of H_i^C, for $i = 0, 1$. If G is pure then H_1^C is true for every cut C of G. On the other hand, if G is not pure then there exists at least one cut C for which H_0^C holds. We therefore determine that G is pure if and only if H_1^C is accepted for every cut C of G. In case we decide that G is pure, we declare it to be a kernel. Otherwise, we partition V into two disjoint subsets, according to a cut C in G, for which the posterior probability ratio $\frac{Pr(H_1^C|C)}{Pr(H_0^C|C)}$ is minimum. We call such a partition a *weakest bipartition* of G.

We first show how to find a weakest bipartition of G. To this end, we make a simplifying probabilistic assumption that for a cut C in G the random variables $\{S_{ij}\}_{(i,j)\in C}$ are pairwise independent given that the corresponding element pairs are all mates or all non-mates. We also assume that mate relations between pairs $(i, j) \in C$ are pairwise independent. We denote the weight of a cut C by $W(C)$ and its number of edges by $|C|$. We denote by $f(C|H_0^C)$ the likelihood that the edges of C connect only non-mates, and by $f(C|H_1^C)$ the likelihood that the edges of C connect only mates. We let $Pr(H_i^C)$ denote the prior probability of H_i^C, $i = 0, 1$.

Using Bayes Theorem we conclude that for a complete graph G and for any cut C in G:

$$\begin{aligned}\log \frac{Pr(H_1^C|C)}{Pr(H_0^C|C)} &= \log \frac{Pr(H_1^C)f(C|H_1^C)}{Pr(H_0^C)f(C|H_0^C)} \\ &= |C|log\frac{p_{mates}\sigma_F}{(1-p_{mates})\sigma_T} + \sum_{(i,j)\in C}\frac{(S_{ij}-\mu_F)^2}{2\sigma_F^2} - \sum_{(i,j)\in C}\frac{(S_{ij}-\mu_T)^2}{2\sigma_T^2} = W(C)\ .\end{aligned}$$

Thus, with our specific edge weight definition, a minimum weight cut of G induces a weakest bipartition of G. However, the computation of a minimum weight cut in a graph with negative edge weights is NP-hard. We give in the next section a heuristic procedure to compute a minimum weight cut for a graph with some negative edge weights.

It remains to show how to decide if G is pure or, equivalently, which stopping criterion to use. For a cut C, we accept H_1^C if and only if $Pr(H_1^C|C) > Pr(H_0^C|C)$. That is, we accept the hypothesis with higher posterior probability.

Let C be a minimum weight cut of G. For every other cut C' of G

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

Therefore, H_1^C is accepted for C if and only if $H_1^{C'}$ is accepted for every cut C' in G. Thus, we accept H_1^C and declare that G is a kernel if and only if $W(C) > 0$. In practice, we also require a kernel to have at least k elements, with a default value of $k = 15$.

3.3 Computing a Minimum Cut

The minimum weight cut problem can be solved efficiently on graphs with non-negative edge weights. Unfortunately, in the basic algorithm one must compute minimum weight cuts in graphs with negative edge weights. This problem is NP-hard even for a complete graph with edge weights 1 or -1 only (Shamir *et al.*, 2002). We overcome this problem using a two-phase process. In the first phase we split the input graph iteratively using a heuristic procedure for computing a minimum weight cut, which is based on a 2-approximation for the related maximum weight cut problem (MAX-CUT). In the second phase we filter from the resulting components all negative weight edges and then apply the basic CLICK algorithm.

Our heuristic for computing a minimum weight cut applies two steps:

- MAX-CUT approximation: Let w^* be the maximum weight in the input graph. Transform all weights using the transformation $f(w) = w^* - w + \epsilon$, for small $\epsilon > 0$, resulting in positive edge weights. Apply a 2-approximation for MAX-CUT (cf. (Hochbaum, 1997)) on the weight-transformed graph, and let (V_1, V_2) be the resulting cut.

- Greedy improvement: Starting from (V_1, V_2) greedily move vertices between sides so as to decrease the weight of the implied cut, using the original edge weights.

This heuristic is applied to the input graph recursively, and the recursion stops whenever the output partition for a component is the trivial one (all vertices are on one side of the partition). We then execute Basic-CLICK on each resulting component, after filtering negative weight edges from it. For testing if a certain component is a kernel, we find its minimum weight cut in the filtered graph, and evaluate the total weight of the edges connecting the two sides of the cut in the unfiltered graph. We declare the graph a kernel if the latter weight is positive.

In order to reduce the running time of the algorithm on large connected components, for which computing a minimum weight cut is very costly, we screen low weight vertices prior to the execution of Basic-CLICK. The screening is done as follows: We first compute the average vertex weight W in the component, and multiply it by a factor which is proportional to the logarithm of the size of the component. We denote the resulting threshold by W^*. We then remove vertices whose weight is below W^*, and continue to do so updating the weight of the remaining vertices, until the updated weight of every (remaining) vertex is greater than W^*. The removed vertices are marked as singletons and handled at a later stage.

3.4 The Full Algorithm

The Basic-CLICK algorithm produces kernels of clusters, which should be expanded to yield the full clusters. The expansion is done by considering the singletons found. We denote by $\mathcal{L}$ and R the current lists of kernels and singletons, respectively. Define the similarity between two sets as the average similarity between their elements. An *adoption step* repeatedly searches for a singleton v and a kernel K whose similarity is maximum among all pairs of singletons and kernels. If the value of this similarity exceeds some predefined threshold, then v is added to K and removed from R. Otherwise, the iterative process ends. For some theoretical justification of the adoption step see (Ben-Dor *et al.*, 1999). After the adoption step takes place, we start a recursive clustering process on the set R of remaining singletons. This is done by discarding all other vertices from the initial graph. We iterate that way until no change occurs.

At the end of the algorithm a *merging step* merges similar clusters. The merging is done iteratively, each time merging two kernels whose similarity is the highest and exceeds a predefined threshold. When two kernels are merged, they are replaced by a new kernel corresponding to their union. Finally, a last singleton adoption step is performed. The full CLICK algorithm is described in Figure 1.

In order to handle large datasets efficiently several enhancements were incorporated into CLICK. When the number of elements exceeds several thousands, memory requirements for storing all pairwise similarity values become a serious bottleneck. In this case we partition the set of elements into *super-components*, each having a limited size, by gradually increasing a weight threshold for the graph edges. For each super-component we evaluate the parameters of its similarity distributions and apply the full algorithm to it. The merge step and the last adoption step are performed later for the whole graph. We also devised a variant of CLICK for clustering fingerprint data using a linear similarity function. This variant uses properties of the similarity function to perform the adoption and merge steps considerably faster than otherwise possible. In addition, it allows the user to specify a *homogeneity parameter* with a default value of μ_T, which serves as a lower bound on the homogeneity of the resulting clustering. For full details on the formation of super-components and handling fingerprint data, see (Sharan, 2002).

4 Results

4.1 Implementation and Simulations

We have implemented the CLICK clustering algorithm in C++. Our implementation uses the algorithm of Hao and Orlin (1994) for minimum weight cut computations. This algorithm has theoretical running time of $O(n^2\sqrt{m})$ (for a graph with n vertices and m edges), and was shown to outperform other minimum cut algorithms in practice (Chekuri *et al.*, 1997). We measured the running times of CLICK on simulated datasets (described below) of various sizes containing 10 equal-size clusters. The running times, as measured on a Pentium III 600MHz, were approximately linear in the number of parameters, and ranged from few seconds for a dataset of 500 elements, to seven minutes for a dataset of 10,000 elements. Linearity (excluding the initial computation of all pairwise similarities) was observed on real datasets of up to 150,000 elements. This is a result of the time reduction heuristics incorporated into CLICK.

We have created an environment for simulating expression data and measuring CLICK's performance on the synthetic data. We use the following simulation setup: The *cluster structure*, i.e., the number and size of clusters, is pre-specified. Each cluster has an associated mean pattern, also called its centroid. Each coordinate of this centroid is drawn uniformly at random from $[0, R]$ for some R, independently from the other coordinates. Each element fingerprint is drawn at random according to a multivariate normal distribution around the corresponding mean pattern with the same standard deviation σ for each coordinate. Similar distribution models are used in other works that model gene expression data (see, e.g., (Ghosh & Chinnaiyan, 2002)).

In our simulations we measured the performance of the algorithm as a function of the cluster structure and the distance Δ in standard deviation units between μ_T and μ_F (due to the nature of the simulations, $\sigma_T \approx \sigma_F$). This distance can be controlled by changing R. Table 1 presents CLICK's results for several simulation setups as measured by the average Jaccard coefficient over 20 runs. The simulated fingerprints in all cases were of length 200, and we used $\sigma = 5$ for all coordinates. It can be seen that CLICK performs well (Jaccard coefficient above 0.8) on all cluster structures even for distances as low as one standard deviation and, as expected, performance worsens when the mate and non-mate distributions get closer.

4.2 The EXPANDER Clustering and Visualization Tool

We have developed a java-based graphical tool, called EXPANDER (EXPression ANalyzer and DisplayER), for gene expression analysis and visualization. This software provides graphical user interface to several clustering methods, including CLICK. It enables visualizing the raw expression data and the clustered data in several ways. In the following we outline the visualization options and demonstrate them on part of the yeast cell-cycle dataset of (Spellman *et al.*, 1998), clustered using CLICK. The original dataset contains samples from yeast cultures synchronized by four independent methods, as well as separate experiments in which some cyclins were induced. Spellman *et al.* (1998) identified in this data 800 genes that are cell-cycle regulated. The dataset that we used contains the expression levels of 698 out of those 800 genes, which have up to three missing entries, over the 72 conditions that cover the α factor, cdc28, cdc15, and elutriation experiments. (As in (Tamayo *et al.*, 1999), the 90 minutes data point was omitted from the cdc15 experiment.) Each row of the 698×72 matrix was normalized to have mean 0 and variance 1. CLICK's solution for this data contains six clusters and 23 singletons. (For a systematic comparison of CLICK, K-means, SOM and CAST on this dataset, see (Shamir & Sharan, 2002).)

Clustering Methods: EXPANDER implements several clustering algorithms including CLICK, K-means, hierarchical clustering and self organizing map (SOM). The user can specify the parameters of each algorithm: Homogeneity parameter for CLICK, number of clusters for K-means, type of linkage (single, average or complete) for hierarchical clustering, and grid dimensions for SOM. In addition, the user can upload an external clustering solution.

Matrix Visualization: EXPANDER can create images for the expression matrix and the similarity matrix (Figure 2). In these images the matrices are represented graphically by coloring each cell according to its content. Cells with neutral values are colored black, increasingly positive values with red of increasing intensity, and increasingly negative values with green of increasing intensity. Each matrix can be shown in two ways: (1) In its raw form; and (2) after reordering the rows of the matrix so that elements from the same cluster appear consecutively. (The columns are also reordered in the similarity matrix.) Ideally, in the reordered expression matrix we expect to see a distinct pattern for each cluster, while the reordered similarity matrix should be composed of red squares, each corresponding to a cluster, in black/green background.

When using hierarchical clustering, the solution dendrogram is displayed alongside with the expression matrix, in which the genes are reordered according to the dendrogram.

Clustering Visualization: EXPANDER provides several visual images of a clustering solution. A graphical overview of the solution is produced by showing for each cluster its mean expression pattern along with error bars indicating the standard deviation in each condition. For an example see Section 4.3. Alternatively, for each single cluster a superposition of all the patterns of its members can be shown.

Another data visualization method provided in EXPANDER is principal component analysis (cf. (Johnson & Wichern, 1982)). This is a method for reducing data dimensionality by projecting high-dimensional data into a low-dimensional space spanned by the vectors that capture maximum variance of the data. In EXPANDER we reduce the data dimension to 2, by computing the two axes that capture maximum variance of the data. The projected data is visualized as points in the plane. Given a clustering solution the points are colored according to their assigned clusters.

As a simple aid for the interpretation of clustering results using biological knowledge, EXPANDER can present and quantify the enrichment of gene functions in a clustering solution (Figure 3). Given a functional annotation (an assignment of an attribute, such as functional category) of the genes in an input dataset, the abundant functional categories in each cluster are shown in a pie chart. For each such category we compute its enrichment in the cluster by computing a hypergeometric p-value, as suggested in (Tavazoie *et al.*, 1999).

4.3 Application to Yeast Cell-Cycle Data

CLICK was first tested on the yeast cell cycle dataset of Cho *et al.* (1998) . That study monitored the expression levels of 6,218 *S. cerevisiae* putative gene transcripts (ORFs) measured at 10-minutes intervals over two cell cycles (160 minutes). We compared CLICK's results to those of GeneCluster (Tamayo *et al.*, 1999). To this end, we applied the same filtering and data normalization procedures of (Tamayo *et al.*, 1999). The filtering removes genes that do not change significantly across samples, leaving a set of 826 genes. The data preprocessing includes the removal of the 90-minutes time-point and normalizing the expression levels of each gene to have mean zero and variance one within each of the two cell-cycles.

CLICK clustered the genes into 18 clusters and left no singletons. These clusters are shown in Figure 4. A summary of the homogeneity and separation parameters for the solutions produced by CLICK and GeneCluster is shown in Table 2. CLICK obtained better results in all four measured parameters. A putative true solution for a subset of the genes was obtained through manual inspection by Cho *et al.* (1998) . Cho *et al.* identified 416 genes that have periodic patterns and partitioned 383 of them into five cell-cycle phases according to their peak time. We calculated Jaccard coefficients for the two solutions based on 250 of these genes that passed the variation filtering. The results are shown in Table 2. It can be seen that CLICK's solution is much more aligned with the putative true solution.

4.4 Human Fibroblasts

We analyzed the dataset of Iyer *et al.* (1999) , that studied the response of several human fibroblasts to serum. It contains expression levels of 8,613 human genes obtained as follows: Human fibroblasts were deprived of serum for 48 hours and then stimulated by addition of serum. Expression levels of genes were measured at 12 time-points after the stimulation. An additional data-point was obtained from a separate unsynchronized sample. A subset of 517 genes whose expression levels changed substantially across samples was analyzed by the hierarchical clustering method of Eisen *et al.* (1998) . The data was first normalized by dividing each entry by the expression level at time zero, and taking a logarithm of the result. For ease of manipulation, we also transformed each fingerprint to have norm 1. The similarity function used was dot-product, giving values identical to those used in (Eisen *et al.*, 1998). CLICK clustered the genes into 6 clusters with no singletons. Table 3 presents a comparison between the clustering quality of CLICK and the hierarchical clustering of (Eisen *et al.*, 1998) on this dataset. The two solutions are incomparable since CLICK's solution has better separation while the other solution has better average homogeneity. In order to directly compare the two algorithms we reclustered the data using CLICK with homogeneity parameter 0.76 (instead of the default value $\mu_T = 0.65$, see Section 3.4), since this value is the average homogeneity of the hierarchical solution. CLICK produced 6 new clusters and 28 singletons. The solution parameters are given in Table 3. Note that CLICK performs better in all parameters.

4.5 Human Cell Cycle

We next studied the gene expression dataset of Whitfield *et al.* (2002) . This dataset contains the expression profiles of synchronized HeLa cells in five independent experiments using three synchronization methods. Whitfield *et al.* (2002) identified in the data 874 genes that are cell-cycle regulated. The experiments were done using two kinds of arrays. We chose to focus on three experiments (76 conditions) that used the larger array, which represents about 29,600 genes, as in the experiments with the smaller array only approximately half the genes were monitored. As we later performed also regulatory motif analysis, we focused on 497 of the genes, that were identified as cell-cycle regulated, were represented on the larger array and had promoter sequences in the public database (NCBI Reference Sequence project (Maglott *et al.*, 2000), release of June 2001). We applied the same data processing methods as described in (Whitfield *et al.*, 2002) to this reduced dataset of expression levels of 497 genes over 76 conditions. CLICK's solution for this data consisted of 9 clusters and 35 singletons. For comparison, we used the partition of the 497 genes according to their cell cycle phases, as provided in (Whitfield *et al.*, 2002).

In order to assess the clusters according to their biological relevance we also retrieved GO annotations (Consortium, 2000) for the genes, and checked the enrichment of each GO category in each of the clusters using a hypergeometric score. In total, 226 genes had known annotations. The homogeneity and separation parameters for the two clustering solutions, along with enrichment p-values that are below 0.01, are shown in Table 4. Notably, CLICK's solution is superior in all parameters and is more aligned with the biological annotations.

4.6 Identifying Regulatory Motifs

In a previous work (Sharan *et al.*, 2002) we have shown the utility of CLICK in identifying regulatory sequence motifs. We outline the method and results below.

Several studies have shown that co-expressed genes tend to share common regulatory elements in their promoter regions (Tavazoie *et al.*, 1999; Zhang, 1999; Brazma & Vilo, 2000). This motivates the following two-step approach for detecting regulatory motifs: (I) Cluster the genes into groups sharing similar expression patterns. (II) In each cluster search for sequence patterns that are over-represented in upstream regions of the sequences of cluster members.

We analyzed the dataset published by Jelinsky *et al.* (2000). In that experiment, expression levels of all 6,200 ORFs of the yeast *S. Cerevisiae* were measured over 26 biological conditions in order to study the cellular response to DNA damage. 2,610 genes that changed by a factor of 3 or more in at least one condition were subjected to cluster analysis. The clustering reported in (Jelinsky *et al.*, 2000) consists of 18 clusters, obtained by GeneCluster (Tamayo *et al.*, 1999). In comparison, CLICK identified 33 clusters with more than 10 members. We then applied the AlignACE motif finding algorithm (Roth *et al.*, 1998; Hughes *et al.*, 2000) to promoter regions (500 bases upstream of the translation start sites) of the genes in each cluster. In total, 26 significant motifs were identified using CLICK's clusters, and 30 such motifs were identified using GeneCluster's. The identified motifs were matched against the SCPD database of experimentally verified yeast transcription factor binding sites (Zhu & Zhang, 1999). Of CLICK's 26 motifs, 17 (65%) were verified by SCPD. Of GeneCluster's 30 motifs, 19 (63%) were verified. 17 of the motifs were common and 13 of these (76%) were verified. This demonstrates the utility of the approach and also the advantage of combining results from different clustering algorithms.

4.7 Tissue classification

An important application of gene expression analysis is the classification of tissue types according to their gene expression profiles. Recent studies (Alon *et al.*, 1999; Golub *et al.*, 1999; Alizadeh *et al.*, 2000; van't Veer *et al.*, 2002) have demonstrated that gene expression data can be used in distinguishing between similar cancer types, thereby allowing more accurate diagnosis and treatment.

In these studies the data consist of expression levels of thousands of genes in several tissues. The tissues originate from two or more known classes, e.g., normal and tumor. The analysis aims at studying the typical expression profile of each class and predicting the classification of new unlabeled tissues. Classification methods employ supervised learning techniques, i.e., the known classifications of the tissues are used to guide the algorithm in building a classifier. These include support vector machines (Ben-Dor *et al.*, 2000; Furey *et al.*, 2000), boosting (Ben-Dor *et al.*, 2000), clustering (Ben-Dor *et al.*, 2000), discriminant analysis (Xiong *et al.*, 2000) and weighted correlation (Golub *et al.*, 1999). Classification can be improved by first limiting the dataset to genes that are informative for the required distinction. Several methods have been suggested to choose subsets of informative genes (Ben-Dor *et al.*, 2000; Dudoit *et al.*, 2002; Furey *et al.*, 2000; Xiong *et al.*, 2000; Xing & Karp, 2001).

Ben-Dor *et al.* (2000) were the first to demonstrate the strength of clustering in cancer classification problems. Key to their method is combining the labeling (known classification) information in the clustering process. Suppose we use a clustering algorithm with at least one free parameter. Given an unlabeled tissue, the clustering algorithm is applied repeatedly with different parameter values on the set of all tissues (known and unknown). Each solution is scored by its level of compatibility with the labeling information, and the best solution is chosen. Each unlabeled tissue is then assigned to the most represented class among the known tissues in its cluster.

The compatibility score for a clustering solution used by Ben-Dor *et al.* is simply the number of tissue pairs that are mates or non-mates in both the true labeling and the clustering solution. The clustering algorithm used in (Ben-Dor *et al.*, 2000) was CAST with Pearson correlation as the similarity function.

We have classified two datasets using CLICK. The first dataset of Alon *et al.* (1999) contains 62 samples of colon epithelial cells, collected from colon-cancer patients. They are divided into 40 ’tumor’ samples collected from tumors, and 22 ’normal’ samples collected from normal colon tissues of the same patients. Of the $\sim$6,000 genes represented in the experiment, 2,000 genes were selected based on the confidence in the measured expression levels. The second dataset of Golub *et al.* (1999) contains 72 leukemia samples. These samples are divided into 25 samples of acute myeloid leukemia (AML) and 47 samples of acute lymphoblastic leukemia (ALL). Of the $\sim$7,000 genes represented in the experiment, 3,549 were chosen based on their variability in the dataset.

The application of CLICK to classify these datasets enumerates several homogeneity parameters for CLICK, and chooses the solution which is most compatible with the given labels. We used the same similarity function and compatibility score as in (Ben-Dor *et al.*, 2000). A sample is not classified if it is either a singleton in the clustering obtained, or no class has a majority in the cluster assigned to that sample. In order to assess the performance of CLICK we employed the leave one out cross validation (LOOCV) technique, as done in (Ben-Dor *et al.*, 2000). According to this technique, one trial is performed for each tissue in the dataset. In the i-th trial, the algorithm tries to classify the i-th sample based on the known classifications of the rest of the samples. The average classification accuracy over all trials is computed. Table 5 presents a comparison between the classification based on CLICK and that of CAST, as reported in (Ben-Dor *et al.*, 2000). The results are comparable, with CAST performing slightly better on the colon dataset, and CLICK performing better on the leukemia dataset.

Next, we tested CLICK’s utility in differentiating between two very similar types of cancer. We concentrated on part of the leukemia dataset composed of the 47 ALL samples only. For these samples an additional sub-classification into either T-cell or B-cell, is provided. An application of CLICK to this dataset resulted in an almost perfect classification (see Table 6).

Finally we examined the influence of feature selection on the classification accuracy. To this end, we sorted the genes in each dataset according to the ratio of their between-sum-of-squares and within-sum-of-squares values, as suggested in (Dudoit *et al.*, 2002). This ratio is computed by the following formula:

$$\frac{BSS(g)}{WSS(g)} = \frac{\sum_{i=1,2} n_i (x_{g,i} - x_g)^2}{\sum_{i=1,2} \sum_{k \in i} (x_g^k - x_{g,i})^2}$$

Here i denotes the class number, n_i its size, k denotes the sample number, $x_{g,i}$ is the average expression level of gene g at class i, x_g is the average expression level of gene g, and x_g^k is the expression level of gene g at sample k. For each LOOCV iteration we chose the 50 genes with the highest value and performed the classification procedure on the reduced dataset which contained the expression levels of these 50 genes only. The results of this analysis are shown in Table 6. For both the colon and leukemia datasets the performance was improved on the reduced dataset.

5 Concluding Remarks

We presented in this paper a novel clustering algorithm which utilizes graph-theoretic and statistical techniques. At the heart of the algorithm is a process of recursively partitioning a weighted graph into components using minimum cut computations. The edge weights and the stopping criterion of the recursion are assigned probabilistic meaning, which gives the algorithm high accuracy. Our method has, however, several limitations: First, the probabilistic model is not always suitable, as is the case for example for protein similarity data (Sharan, 2002). Second, our algorithm cannot identify (with confidence) very small clusters, as small sets of elements may satisfy the kernel criterion merely by chance. Last, our method is designed to produce a hard partition of the elements into clusters, although such a partition is not always adequate, e.g., when element classification is hierarchical by nature.

CLICK was tested on several biological datasets, originating from a variety of applications, and was shown to outperform extant clustering algorithms according to several common figures of merit. It is also fast, allowing high-accuracy clustering of large datasets of size over 100,000 in a couple of hours. CLICK is available as part of the EXPANDER software package for clustering and visualizing gene expression data.

Acknowledgments

We thank Naama Arbili for her great help in programming CLICK. We thank Rani Elkon and Amos Tanay for many helpful discussions. R. Sharan was supported by a Fulbright Grant. This study was supported by a research grant from the Ministry of Science and Technology, Israel.

References

Alizadeh, A., Eisen, M., Davis, R. *et al.* (2000) Distinct types of diffuse large B-cell lymphoma identified by gene expression profiling. *Nature,* **403** (6769), 503–511.

Alon, U., Barkai, N., Notterman, D. *et al.* (1999) Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. *Proc. Natl. Acad. Sci. USA,* **96**, 6745–6750.

Ben-Dor, A., Bruhn, L., Friedman, N. *et al.* (2000) Tissue classification with gene expression profiles. *Journal of Computational Biology,* **7(3/4)**, 559–583.

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

Ben-Hur, A., Elisseeff, A. & Guyon, I. (2002) A stability based method for discovering structure in clustered data. In *Pac. Symp. Biocomput.* pp. 6–17.

Brazma, A. & Vilo, J. (2000) Gene expression data analysis. *FEBS Letters,* **480**, 17–24.

Chekuri, C., Goldberg, A., Karger, D., Levine, M. & Stein, C. (1997) Experimental study of minimum cut algorithms. In *Proceedings of the Eighth Annual ACM-SIAM Symposium on Discrete Algorithms* pp. 324–333.

Cho, R., Campbell, M., Winzeler, E. *et al.* (1998) A genome-wide transcriptional analysis of the mitotic cell cycle. *Mol. Cell,* **2**, 65–73.

Consortium, T. G. O. (2000) Gene Ontology: tool for the unification of biology. *Nature Genetics,* **25**, 25–29.

Dudoit, S., Fridlyand, J. & Speed, T. (2002) Comparison of discrimination methods for the classification of tumors using gene expression data. *Journal of the American Statistical Association,* **97** (457), 77–87.

Eisen, M. & Brown, P. (1999) DNA arrays for analysis of gene expression. In *Methods in Enzymology* vol. 303,. pp. 179–205.

Eisen, M., Spellman, P., Brown, P. & Botstein, D. (1998) Cluster analysis and display of genome-wide expression patterns. *Proc. Natl. Acad. Sci. USA,* **95**, 14863–14868.

Everitt, B. (1993) *Cluster analysis*. Third edition,, Edward Arnold, London.

Furey, T., Cristianini, N., Duffy, N., Bendarski, D., Schummer, M. & Haussler, D. (2000) Support vector machine classification and validation of cancer tissue samples using microarray expression data. *Bioinformatics,* **16**, 906–914.

Ghosh, D. & Chinnaiyan, A. (2002) Mixture modelling of gene expression data from microarray experiments. *Bioinformatics,* **18**, 275–286.

Golub, T., Slonim, D., Tamayo, P. *et al.* (1999) Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring. *Science,* **286**, 531–537.

Hansen, P. & Jaumard, B. (1997) Cluster analysis and mathematical programming. *Mathematical Programming,* **79**, 191–215.

Hao, J. & Orlin, J. (1994) A faster algorithm for finding the minimum cut in a directed graph. *Journal of Algorithms,* **17** (3), 424–446.

Hartigan, J. (1975) *Clustering Algorithms*. John Wiley and Sons.

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

Hartuv, E. & Shamir, R. (2000) A clustering algorithm based on graph connectivity. *Information Processing Letters,* **76(200)**, 175–181.

Herwig, R., Poustka, A., Meuller, C., Lehrach, H. & O'Brien, J. (1999) Large-scale clustering of cDNA-fingerprinting data. *Genome Research,* **9** (11), 1093–1105.

Hochbaum, D. S., ed. (1997) *Approximation Alogrithms for NP-Hard Problems*. PWS Publishing, Boston.

Hughes, J., Estep, P., Tavazoie, S. & Church, G. (2000) Computational identification of cis-regulatory elements associated with groups of functionally related genes in *Saccharomyces Cerevisiae*. *J. Mol. Biol.,* **296**, 1205–1214.

Iyer, V., Eisen, M., Ross, D. *et al.* (1999) The transcriptional program in the response of human fibroblasts to serum. *Science,* **283** (1), 83–87.

Jelinsky, S., Estep, P., Church, Q. & Samson, L. (2000) Regulatory networks revealed by transcriptional profiling of damaged *Saccharomyces Cerevisiae* cells: Rpn4 links base excision repair with proteasomes. *Mol. Cell Biol.,* **20(21)**, 8157–8167.

Johnson, R. & Wichern, D. (1982) *Applied multivariate statistical analysis*. Prentice-Hall Inc.

Kerr, M., Martin, M. & Churchill, G. (2000) Analysis of variance for gene expression microarray data. *Journal of Computational Biology,* **7(6)**, 819–837.

Maglott, D., Katz, K., Sicotte, H. & Pruitt, K. (2000) NCBI's LocusLink and RefSeq. *Nucleic Acids Research,* **28(1)**, 126–128.

Mirkin, B. (1996) *Mathematical Classification and Clustering*. Kluwer.

Pollard, K. & van der Laan, M. (2002) A method to identify significant clusters in gene expression data. In *Sixth World Multiconference on Systemics, Cybernetics, and Informatics* pp. 318–325.

Roth, F., Hughes, J., Estep, P. & Church, G. (1998) Finding DNA regulatory motifs within unaligned noncoding sequences clustered by whole-genome mRNA quantitation. *Nature Biotech.,* **16**, 939–945.

Shamir, R. & Sharan, R. (2002) Algorithmic approaches to clustering gene expression data. In *Current Topics in Computational Biology,* (Jiang, T., Smith, T., Xu, Y. & Zhang, M., eds),. MIT Press pp. 269–299.

Shamir, R., Sharan, R. & Tsur, D. (2002) Cluster graph modification problems. In *Proceedings of the 27th International Workshop Graph-Theoretic Concepts in Computer Science (WG)* pp. 379–390. LNCS 2573.

Sharan, R. (2002). *Graph Modification Problems and their Applications to Genomic Research.* PhD thesis, School of Computer Science, Tel-Aviv University. http://www.icsi.berkeley.edu/∼roded.

Sharan, R., Elkon, R. & Shamir, R. (2002) Cluster analysis and its applications to gene expression data. In *Proceedings of the 38th Ernst Schering workshop on Bioinformatics and Genome Analysis,* (Mewes, H.-W., Seidel, H. & Weiss, B., eds),. Springer Verlag pp. 83–108.

Sharan, R. & Shamir, R. (2000) CLICK: a clustering algorithm with applications to gene expression analysis. In *Proceedings of the Eighth International Conference on Intelligent Systems for Molecular Biology (ISMB)* pp. 307–316 AAAI Press.

Spellman, P., Sherlock, G., Zhang, M. *et al.* (1998) Comprehensive identification of cell cycle regulated gene of the yeast *Saccharomyces Cerevisia* by microarray hybridization. *Mol. Biol. Cell,* **9**, 3273–3297.

Tamayo, P., Slonim, D., Mesirov, J. *et al.* (1999) Interpreting patterns of gene expression with self-organizing maps: methods and application to hematopoietic differentiation. *Proc. Natl. Acad. Sci. USA,* **96**, 2907–2912.

Tavazoie, S., Hughes, J., Campbell, M., Cho, R. & Church, G. (1999) Systematic determination of genetic network architecture. *Nature Genetics,* **22**, 281–285.

Tibshirani, R., Walther, G. & Hastie, T. (2000). Estimating the number of clusters in a dataset via the gap statistics. Technical report Stanford University Stanford.

van't Veer, L., Dai, H., van de Vijver, M. *et al.* (2002) Gene expression profiling predicts clinical outcome of breast cancer. *Nature,* **415**, 530–536.

Whitfield, M., Sherlock, G. *et al.* (2002) Identification of genes periodically expressed in the human cell cycle and their expression in tumors. *Molecular Biology of the Cell,* **13**, 1977–2000.

Xing, E. & Karp, R. (2001) CLIFF: clustering of high-dimensional microarray data via iterative feature filtering using normalized cuts. *Bioinformatics,* **17 Suppl. 1**, S306–S315.

Xiong, M., Jin, L., Li, W. & Boerwinkle, E. (2000) Computational methods for gene expression based tumor classification. *Biotechniques,* **29**, 1264–1270.

Yang, Y., Dudoit, S., Luu, P., Peng, V., Ngai, J. & Speed, T. (2002) Normalization for cDNA microarray data: a robust composite method addressing single and multiple slide systematic variation. *Nucleic Acid Research,* **30(4)**, e15.

Yeung, K., Haynor, D. & Ruzzo, W. (2001) Validating clustering for gene expression data. *Bioinformatics*, **17**, 309–318.

Zhang, M. (1999) Large scale expression data analysis: a new challenge to computational biologist. *Genome Research*, **9**, 681–688.

Zhu, J. & Zhang, M. (1999) SCPD: a promoter database of the yeast *Saccharomyces Cerevisiae*. *Bioinformatics*, **15**, 607–611.

Cluster structure	$\Delta = 0.75$	$\Delta = 1$	$\Delta = 1.5$	$\Delta = 2$	$\Delta = 2.5$
5 clusters of size 100	0.81	0.95	0.99	1	1
10 clusters of size 50	0.39	0.8	0.97	1	1
6 clusters of size 50,60,...,100	0.75	0.93	0.99	1	1

Table 1: CLICK's accuracy in simulations: Average Jaccard coefficients over 20 runs. Δ is specified in standard deviations.

Program	#Clusters	Homogeneity		Separation		Jaccard
		H_{Ave}	H_{Min}	S_{Ave}	S_{Max}	
CLICK	18	0.62	0.46	-0.05	0.33	0.54
GeneCluster	30	0.59	0.22	-0.01	0.81	0.28

Table 2: A comparison between CLICK and GeneCluster on a yeast cell-cycle dataset of Cho *et al.* (1998). The Jaccard score is computed with respect to the putative solution of Cho *et al.* (1998).

Program	#Clusters	Homogeneity		Separation	
		H_{Ave}	H_{Min}	S_{Ave}	S_{Max}
CLICK.1	6	0.72	0.42	-0.29	0.55
CLICK.2	6	0.78	0.68	-0.19	0.54
Hierarchical	10	0.76	0.65	-0.08	0.75

Table 3: A comparison between CLICK and the hierarchical clustering of Eisen *et al.* (1998) on the dataset of response of human fibroblasts to serum (Iyer *et al.*, 1999). CLICK.1 represents CLICK's solution with the default homogeneity parameter. CLICK.2 represents a solution of CLICK with the homogeneity parameter set to 0.76.

Program	#Clusters	Homogeneity		Separation		Significantly Enriched Categories (p)
		H_{Ave}	H_{Min}	S_{Ave}	S_{Max}	
CLICK	9	0.44	0.31	0.07	0.33	REP ($5 \cdot 10^{-5}$), CC (0.002)
(Whitfield *et al.*, 2002)	5	0.22	0.12	0.12	0.23	REP ($7 \cdot 10^{-4}$)

Table 4: A comparison between the solutions of CLICK and (Whitfield *et al.*, 2002) on a human cell-cycle dataset of (Whitfield *et al.*, 2002). Abbreviations for GO categories: REP – DNA replication and chromosome cycle; CC – mitotic cell cycle.

Dataset	Method	Correct	Incorrect	Unclassified
Colon	CLICK	87.1	12.9	0.0
	CAST	88.7	11.3	0.0
Leukemia	CLICK	94.4	2.8	2.8
	CAST	87.5	12.5	0.0

Table 5: A comparison of the classification quality of CLICK and CAST on the colon data of (Alon *et al.*, 1999) and the leukemia data of (Golub *et al.*, 1999). For each dataset and clustering algorithm the percents of correct classifications (in the LOOCV iterations), incorrect classifications and unclassified elements are specified.

Dataset	Size	Correct	Incorrect	Unclassified
Colon	2000	87.1	12.9	0.0
	50	90.3	9.7	0.0
Leukemia	3549	94.4	2.8	2.8
	50	97.2	2.8	0.0
ALL	3549	97.9	0.0	2.1
	50	97.9	2.1	0.0

Table 6: A summary of the classifications obtained by CLICK on the colon data of (Alon *et al.*, 1999), the whole leukemia dataset of (Golub *et al.*, 1999), and part of the leukemia dataset which contains ALL samples only. For each dataset classifications were performed with respect to the total number of genes, and with respect to the 50 most informative genes. The percents of correct classifications (in the LOOCV iterations), incorrect classifications and unclassified elements are specified.

Figure Legends

Figure 1: The full CLICK algorithm. $\mathcal{L}$ and R are the current lists of kernels and singletons, respectively. Initially, R contains all elements, and $\mathcal{L}$ is empty. The Split procedure partitions the graph induced on the elements in R into components, using approximated minimum weight cut computations.

Figure 2: Matrix images in EXPANDER. A: The raw yeast cell-cycle data matrix of Spellman *et al.* (1998). Rows correspond to genes and columns to conditions. Reds/greens represent over/under-expression levels. B: The data matrix after clustering the genes into six clusters and reordering the rows accordingly. C: The similarity matrix. Rows and columns correspond to genes. Reds/greens: higher/lower similarity values. D: The similarity matrix after clustering the genes and reordering the rows and columns accordingly. (For a clear image, an arbitrary subset of 400 genes and 25 conditions only is shown in A and B.)

Figure 3: Functional enrichment. The pie charts show the functional enrichment for CLICK's clusters 3 (left) and 4 (right) computed on the yeast cell-cycle of Spellman *et al.* (1998). Only functional categories containing at least 10% of the genes in a cluster are shown. The most enriched categories are transport (in cluster 3, $p = 1.7 \cdot 10^{-7}$) and developmental processes (in cluster 4, $p = 1.8 \cdot 10^{-6}$).

Figure 4: CLICK's clustering of the yeast cell cycle data of (Cho *et al.*, 1998). x-axis: Time points 0-80,100-160 at 10-minutes 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.

```
While some change occurs do:
    Split($G_R$).
    Let $\mathcal{S}$ be the set of resulting components.
    For each $C \in \mathcal{S}$ do:
        Remove edges with negative weight from $C$.
        Filter low-degree vertices from $C$.
        Basic-CLICK($C$).
    Let $\mathcal{L}'$ be the list of kernels produced.
    Let $R$ be the set of remaining singletons.
    Adoption($\mathcal{L}', R$).
    $\mathcal{L} \leftarrow \mathcal{L} \cup \mathcal{L}'$.
Merge($\mathcal{L}$).
Adoption($\mathcal{L}, R$).
```

Figure 1:

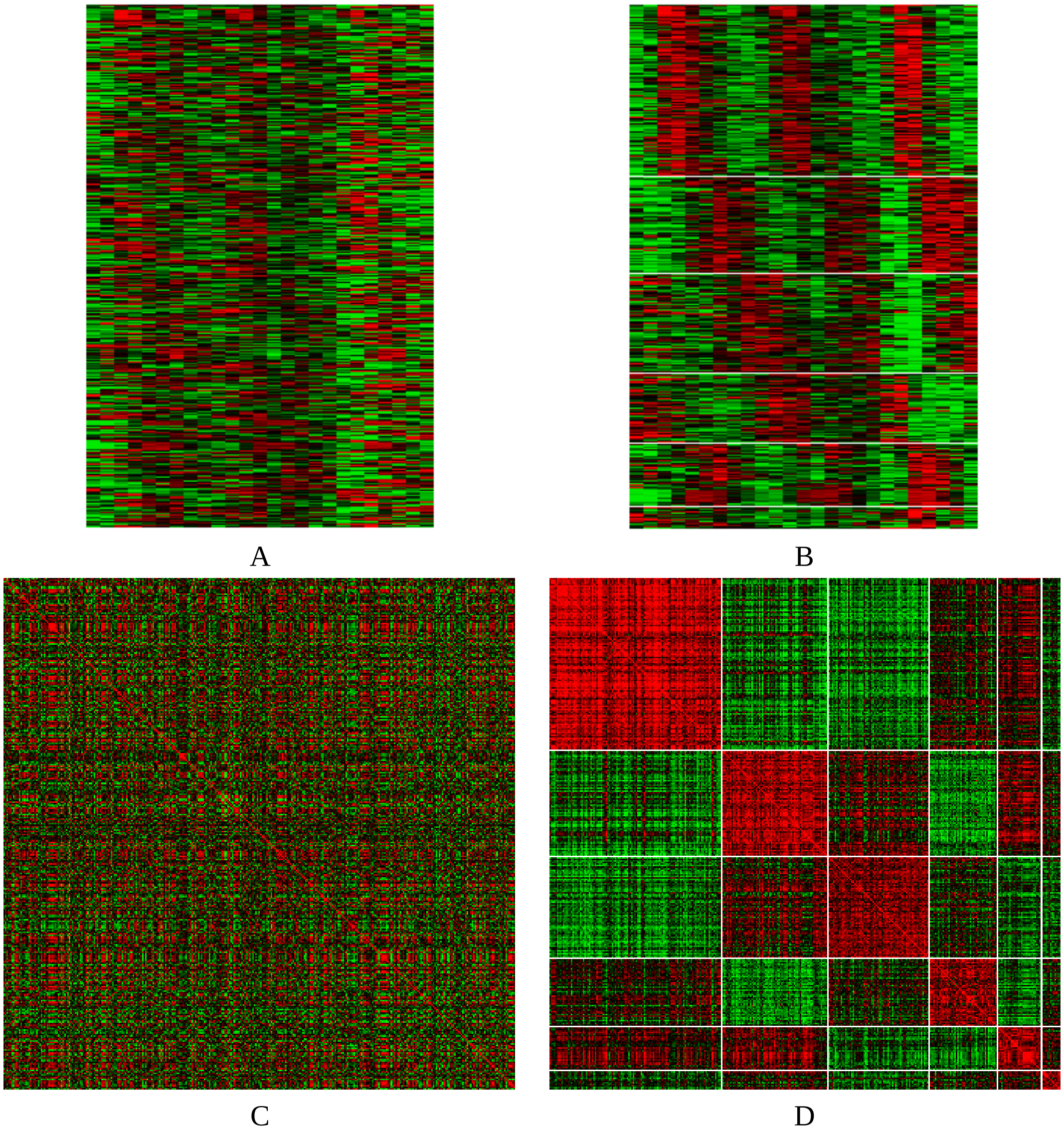

Figure 2:

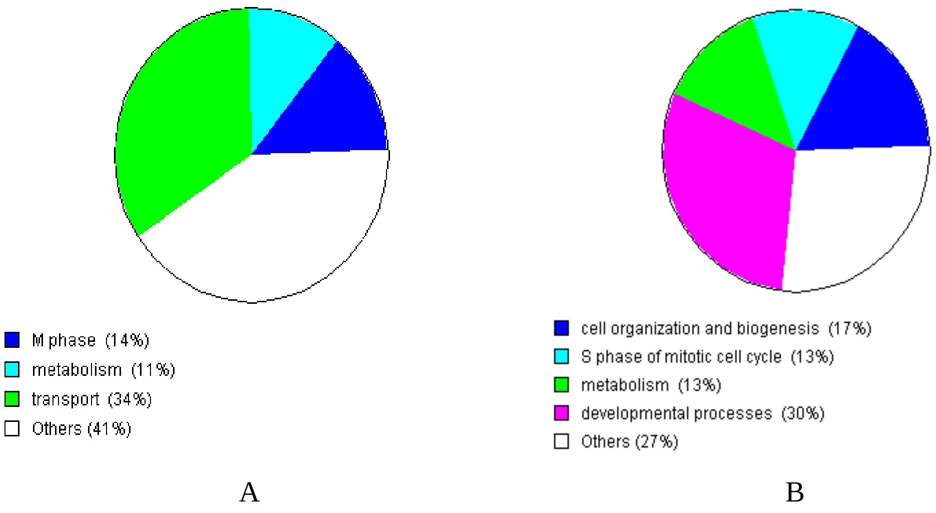

Figure 3:

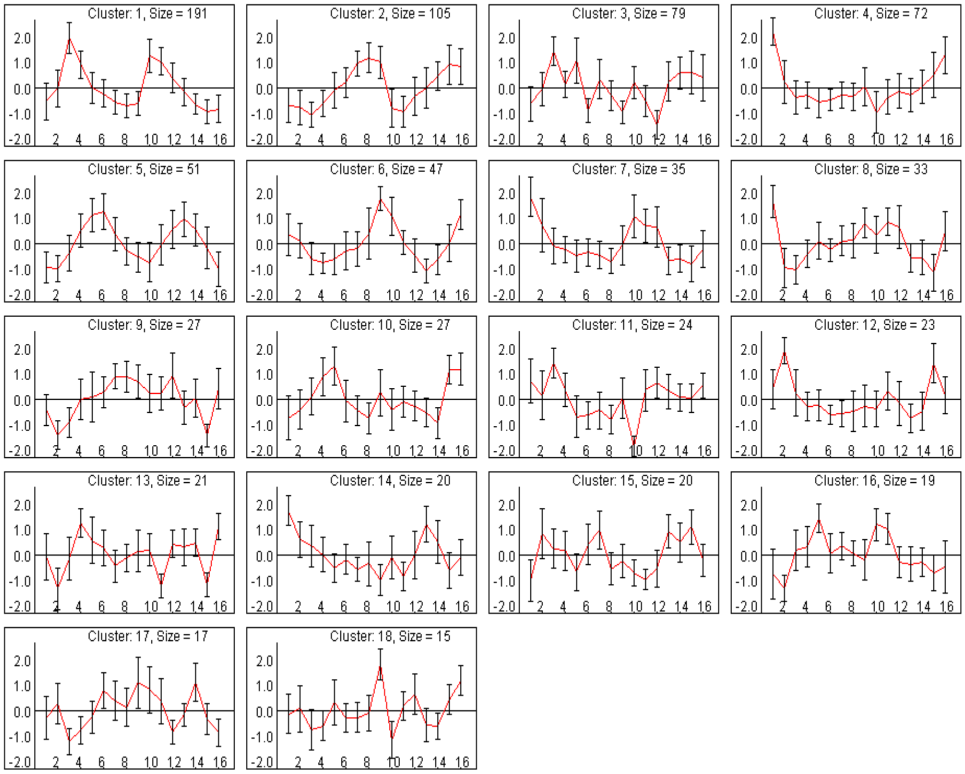

Figure 4: