Brain Inspired Cognitive Systems
August 29 – September 1, 2004
University of Stirling, Scotland, UK

A NEW BICLUSTERING TECHNIQUE BASED ON CROSSING MINIMIZATION

Ahsan Abdullah
Center for Agro-Informatics Research
National University of Computers & Emerging Sciences,
Islamabad, Pakistan. E-maill: ahsan@nu.edu.pk

Amir Hussain
Department of Computing Science & Mathematics
University of Stirling, Stirling FK9 4LA
Scotland, UK. E-mail: ahu@cs.stir.ac.uk

ABSTRACT
Clustering only the records in a database (or data matrix) gives a global view of the data. For detailed or a local view, biclustering is required i.e. clustering the records and the attributes simultaneously. In this paper, a new graph theoretic, crossing minimization based biclustering technique is proposed. The performance of the proposed technique is compared with a recently reported classical biclustering technique of Cheng & Church [4] demonstrating notably better results. A white noise model is also used to demonstrate the robustness of the proposed technique.

INTRODUCTION & BACKGROUND
Different applications, sensors etc. are constantly recording data in a diverse set of application domains such as agro informatics, bio informatics, e-commerce etc. A call made by a phone, an email sent/received, money drawn from an ATM machine, web browsing, payment made by a credit card, using electrical appliances etc. all result in data recording or transaction logging. The data is usually logged in a database (or a data matrix) and has been historically analyzed in "two" dimensions [13] (i) record dimension i.e. rows and (ii) the attribute dimension i.e. columns. The objective is discovery of interesting patterns; this is achieved by either comparing rows in the data matrix or columns in the data matrix. Common objectives pursued when analyzing the data matrix include:

1. Grouping of records according to multiple attributes (unsupervised learning).
2. Classification of a new record based on its attributes and the attributes of other records, with known classification (supervised learning).
3. Grouping of attributes based on the attributes of a number of records.
4. Classification of a new group, given the attributes of records in that group.

Traditional (one-way) clustering techniques can be used to group either records or attributes; therefore, objectives 1 and 3 can be pursued directly, while objectives 2 and 4 can be pursued indirectly.

Application of traditional clustering algorithms for detailed data analysis and pattern discovery has significant problems. Consider the case of hundreds and thousands of items being sold at a super market. Typically the items procured by men will have a high level of mutual exclusivity as compared to items procured by women. Discovering such similar local grouping of attributes (or items) may be the key to uncovering many interesting and useful patterns that are not apparent otherwise. It is therefore highly desirable to move beyond the one-way clustering paradigm, and to develop algorithmic approaches capable of discovering local patterns in data matrices i.e. biclustering.

One-way clustering methods can be applied to either the rows or the columns of the data matrix, separately. Biclustering methods, on the other hand, perform simultaneous clustering of rows and columns. This means that clustering methods give a *global view* while biclustering algorithms give a *local view*. When clustering algorithms are used, each record in a given cluster is evaluated using all the attributes. This may actually distort the cluster, as all attributes may not be contributing towards that cluster. Similarly, each attribute in an attribute cluster is characterized by the activity of all the records, which may not always be true. However, each record in a bicluster is selected using only a subset of the attributes and each attribute in a bicluster is selected using only a subset of the records [4].

The goal of biclustering techniques is thus to identify subgroups of records and the subgroups of attributes, by performing simultaneous clustering of both rows and columns of the data matrix, instead of clustering rows and columns separately, and more or less independently. Therefore, biclustering approaches are the key technique to be used when one or more of the following situations apply:

1. Only a small set of the records are active in a pattern of interest.
2. An interesting group is active only in a subset of the attributes.

For these reasons, biclustering algorithms should identify groups of records and attributes, as per the following restrictions [13]:

- A cluster of records should be defined with respect to only a subset of the attributes.
- A cluster of attributes should be defined with respect to only a subset of the records.
- The clusters should not be exclusive and/or exhaustive: a record or attribute should be able to belong to more than one cluster or to no cluster at all and be grouped using a subset of attributes or records, respectively.

In this paper we will deviate from the third restriction, as only mutually exclusive clusters will be considered. Additionally, robustness in biclustering algorithms is especially relevant because of high level of noise in the data, which makes the use and requirement of intelligent tools crucial.

2.0 PREVIOUS WORK

Biclustering was introduced in the seventies by Hartigan [9]. Crossing minimization paradigm was first used for clustering by Abdullah and Brobst [2], but only for one-way clustering. Although the complexity of the biclustering problem depends on the exact problem formulation, and specifically, on the quantification measures for bicluster evaluation, almost all interesting variants of this problem are NP-complete. Therefore, instead of coming up with exact algorithms, the pragmatic approach would be to come up with heuristics or approximation algorithms. In this regard, the researchers have used different approaches to solve this problem, which can be divided into five classes as follows (in view of space constraint details are not given here, interested reader is referred to [13]):

1. Divide and Conquer
2. Greedy Iterative Search
3. Exhaustive Bicluster Enumeration
4. Distribution Parameter Identification
5. Iterative Row and Column Clustering Combination

3.0 DEFINITIONS, AND MODEL FORMULATION

The input data for a clustering problem is typically given in one of the two forms as described by [8]:

- Data matrix (or *object-by-variable structure*) $\mathbf{S}$ is an $n \times p$ matrix, where corresponding to each of the n objects there are p variables, also called measurements or attributes. Usually $n >> p$.

- Similarity (or dissimilarity) matrix (or *object-by-object structure*) is an $n \times n$ symmetric matrix, which contains the pair-wise similarity (or dissimilarity) that is usually computed from p for all pairs of n objects.

Because of the nature of the problem, in this paper, only data matrix will be considered.

Let $\mathbf{S}$ correspond to the matrix representation of a weighted bipartite graph, such that each row of the data matrix corresponds to a vertex of the bipartite graph in one partition, and each attribute of the data matrix corresponds to a vertex of the bipartite graph in the other partition. Note that $\mathbf{S}$ is unlikely to be symmetric. To get a simple graph from a weighted graph, edge weights are discretized. This consequently converts $\mathbf{S}$ into a binary data matrix denoted by $\mathbf{S}_B$. Note that the transformation of $\mathbf{S}$ to $\mathbf{S}_B$ is like a double edged sword, as this may result in removal of weak and un-interesting relationships, even noise. However, this can also result in loss of actual data too, as it is very difficult to differentiate between noise and data.

Let the bipartite graph (bi-graph) corresponding to $\mathbf{S}_B$ be denoted by G_B, most likely G_B will not consist of pure clusters and zero noise. However, to facilitate defining the model, it is first assumed that the data matrix consists of pure disjoint clusters and zero noise, with perfect grouping of rows and columns corresponding to the clusters. Such a data matrix is denoted by $\mathbf{S}^*_B$ and the corresponding bi-graph denoted by $G^*_B(V_0, V_1, E)$. G^*_B will be a union of bipartite graph cliques i.e. $K_{i,j}$ and V_0, V_1 is the bipartition of vertices of G^*_B such that $V_0 \cap V_1 = \varnothing$. E is the edge set such that $e = |E|$. As two-way clustering (or biclustering) is being considered in this paper i.e. clustering the rows and columns of $\mathbf{S}$, therefore, $i \geq j$ for $K_{i,j}$ and $n = |V_1| + |V_0|$. Density of G_B is denoted by δ i.e. $\delta(G_B) = e/(|V_1| \times |V_0|)$.

Let a bi-graph drawing (or layout) of G^*_B be obtained by placing the vertices of V_0 and V_1 on distinct locations on two horizontal lines y = 1 i.e. TOP (top partition) and y = 0 i.e. BOT (bottom partition) in the XY-plane, respectively. The vertices of every clique are located on consecutive x-coordinates for TOP and BOT partitions i.e. in the same neighborhood. Now draw each edge with one straight-line segment which connects the points on y = 0 and y = 1 where the end vertices of the edges were placed. This will result in a bi-graph drawing, in which only those edges intersect, that belong to the same clique. Let φ^* be the bi-graph drawing corresponding to G^*_B and the order of vertices in the bipartitions V_0 and V_1 is denoted by π^*_0 and π^*_1, respectively. Note that for $K_{i,j}$ (as a convention) it will be assumed that the vertices in bi-partition *i* will be placed in TOP and the vertices in *j* will be placed in BOT. For more on graph drawing reader is referred to [3].

Let the order of vertices in the bipartitions V_0 and V_1 of G_B be denoted by π_0 and π_1, respectively. Let $\Phi(G)$ denote the set of all possible bi-graph drawings of the bi-graph G_B. Thus the optimum biclustering problem is defined as: given a bi-graph G_B (V_0, V_1, E) find φ^* among Φ (G) with relative permutation of vertices π^*_0 and π^*_1.

Relative permutation of vertices means vertices of a cluster remain in their cluster neighborhood, and do not cross cluster boundaries. For example, if vertices labeled {2, 5, 9, 1} and {3, 7, 6, 4, 8} correspond to two clusters C1 and C2, then the relative ordering of vertices within the cluster is unimportant. However, cluster quality deteriorates when noise causes (say) vertex 5 to move from C1 to C2.

Figure-1(a) shows a G^*_B that consists of $K_{4,8} \cup K_{4,4} \cup K_{8,2}$ with n = 16 + 14 and e = 64. Figure-2(a) shows the corresponding $\mathbf{S}_B$.

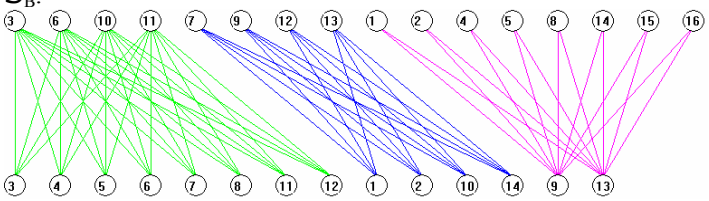


Figure-1(a): $G^*_B = K_{4,8} \cup K_{4,4} \cup K_{8,2}$ with 232 crossings

Real data sets will always have noise and disorder. Therefore, as a first step towards modeling non-ideal data sets, the vertices in each bipartition of G^*_B are now randomly permuted. Figure-1(b) shows the G^*_B of Fig-1(a) with vertices randomly permuted. Figure-2(b) shows the corresponding $\mathbf{S}_B$.

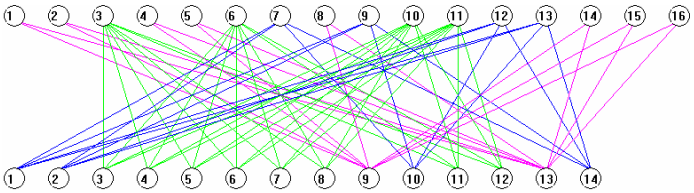
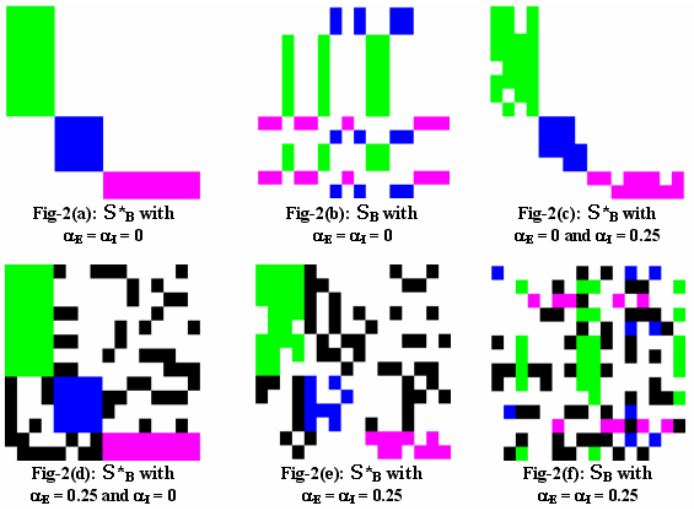


Figure-1(b): G_B obtained after randomly permuting G^*_B resulting in 894 crossings

Now the second step towards non-ideal data sets i.e. "contamination" of G_B by randomly adding edges (white noise) between $v \in V_0$ (i.e. TOP) and $u \in V_1$ (i.e. BOT) with probability $\alpha_E < ½$. Similarly, edges are also randomly removed (white noise) from each of $K_{i,j}$ with probability $\alpha_I < ½$ resulting in G_B such that $\alpha_E = \alpha_I$. The effect of these operations on S_B would be replacement of 1's by 0's for α_E and replacement of 0's by 1's for α_I (the original 0's not the converted 0's).

The effect of white noise is demonstrated in Figure-2(c) that shows the input S^*_B of Figure-2(a) with $\alpha_E = 0$ and $\alpha_I = 0.25$, note that the corresponding effect on G_B will be removal of edges. Figure-2(d) shows S^*_B of Figure-2(a) with $\alpha_E = 0.25$ and $\alpha_I = 0$, note that the corresponding effect on G_B will be addition of edges across cliques. The collective effect of $\alpha_E = \alpha_I = 0.25$ is shown in Figure-2(e) which is S^*_B with noise. Permuting the vertices of the bi-graph corresponding to Fig-2(e) generates an isomorphic graph drawing, and the corresponding data matrix is shown in Figure-2(f). Note that Figure-2(f) without color or grey coding (showing cluster classification) would correspond to real data. The objective of this paper is to demonstrate the working of a technique, that takes as input S_B similar to Fig-2(f), and using a CMH reorders the rows to generate an output similar to Fig-2(e) i.e. S^*_B.



Let G^+_B denote the isomorphic graph as a result of using Crossing Minimization Heuristic (CMH) on G_B and the corresponding bi-graph drawing be φ^+. S^+_B is a visualization of the clustering solution generated after running CMH on G^+_B. Note that for *n* and *e* in hundreds φ (similar to Figure-1(b)) is visually meaningless and only S^+_B makes sense. However, when *n* and *e* are in thousands, screen resolutions prohibit displaying S^+_B (similar to Figure-2(b)) completely, and then only results extracted using statistical means are meaningful (please see section 3).

For the sake of understanding, the proposed solution in rather oversimplified form is given as a five step procedure as follows:

1. Discretize S to obtain S_B.
2. Run a CMH on G_B corresponding to S_B to get G^+_B
3. Extract clusters from G^+_B.
4. Analyze clusters.

Example-1

In this example the working of the proposed solution is demonstrated using a variant of MaxSort CMH [2] i.e. MinSort (MS). Details of the heuristic are given in section 2.4. Figure 3(a) shows a simple input data matrix S i.e. a table consisting of 16 rows (R1 to R16) and ten columns or variables (V1 to V10). Note that the last row in Fig-3(a) shows the average value of each column.

	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10
R1	4	4	1	1	1	1	1	4	4	1
R2	1	1	3	3	3	3	3	1	1	3
R3	4	4	1	1	1	1	1	4	4	1
R4	4	4	1	1	1	1	1	4	4	1
R5	1	1	3	3	3	3	3	1	1	3
R6	4	4	1	1	1	1	1	4	4	1
R7	1	1	3	3	3	3	3	1	1	3
R8	4	4	1	1	1	1	1	4	4	1
R9	4	4	1	1	1	1	1	4	4	1
R10	4	4	1	1	1	1	1	4	4	1
R11	4	4	1	1	1	1	1	4	4	1
R12	1	1	3	3	3	3	3	1	1	3
R13	1	1	3	3	3	3	3	1	1	3
R14	1	1	3	3	3	3	3	1	1	3
R15	4	4	1	1	1	1	1	4	4	1
R16	1	1	3	3	3	3	3	1	1	3
	2.7	2.7	1.9	1.9	1.9	1.9	1.9	2.7	2.7	1.9

Fig-3(a): Input i.e. S

	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10
R1	1	1	0	0	0	0	0	1	1	0
R2	0	0	1	1	1	1	1	0	0	1
R3	1	1	0	0	0	0	0	1	1	0
R4	1	1	0	0	0	0	0	1	1	0
R5	0	0	1	1	1	1	1	0	0	1
R6	1	1	0	0	0	0	0	1	1	0
R7	0	0	1	1	1	1	1	0	0	1
R8	1	1	0	0	0	0	0	1	1	0
R9	1	1	0	0	0	0	0	1	1	0
R10	1	1	0	0	0	0	0	1	1	0
R11	1	1	0	0	0	0	0	1	1	0
R12	0	0	1	1	1	1	1	0	0	1
R13	0	0	1	1	1	1	1	0	0	1
R14	0	0	1	1	1	1	1	0	0	1
R15	1	1	0	0	0	0	0	1	1	0
R16	0	0	1	1	1	1	1	0	0	1

Fig-3(b): S_B based on average value

Figure-3(b) shows S_B obtained after using a discretization threshold of average value of each column (or row) of Fig-3(a) i.e. replacing values > average by 1 and others by 0. The corresponding bipartite graph drawing is shown in Figure-3(c).

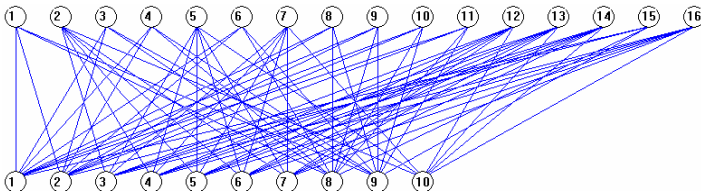


Fig-3(c): Bipartite graph drawing corresponding to Fig-3(b)

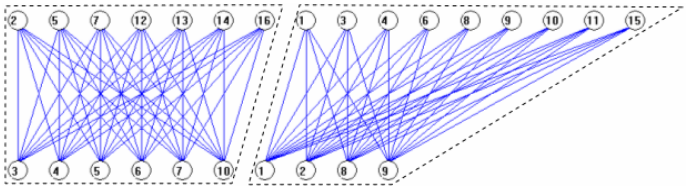


Fig-3(d): Bipartite graph drawing obtained after running MS on Fig-3(c)

Figure-3(d) shows an isomorphic graph drawing of G_B obtained after running MS (MinSort) Heuristic on Figure-3(c). Figure-3(d) also shows two clusters extracted i.e. C_1, and C_2 shown by dotted lines. Figure-3(e) is the S^+_B corresponding to Fig-3(d), while Fig-3(e) is the final solution with clusters identified by bold boxes. Observe that clustering in Fig-3(e) corresponds to perfect row matches for the table shown in Fig-3(a).

	A3	A4	A5	A6	A7	A10	A1	A2	A8	A9
R2	1	1	1	1	1	1	0	0	0	0
R5	1	1	1	1	1	1	0	0	0	0
R7	1	1	1	1	1	1	0	0	0	0
R12	1	1	1	1	1	1	0	0	0	0
R13	1	1	1	1	1	1	0	0	0	0
R14	1	1	1	1	1	1	0	0	0	0
R16	1	1	1	1	1	1	0	0	0	0
R1	0	0	0	0	0	0	1	1	1	1
R3	0	0	0	0	0	0	1	1	1	1
R4	0	0	0	0	0	0	1	1	1	1
R6	0	0	0	0	0	0	1	1	1	1
R8	0	0	0	0	0	0	1	1	1	1
R9	0	0	0	0	0	0	1	1	1	1
R10	0	0	0	0	0	0	1	1	1	1
R11	0	0	0	0	0	0	1	1	1	1
R15	0	0	0	0	0	0	1	1	1	1

Fig-3(e): Discretized output

	A3	A4	A5	A6	A7	A10	A1	A2	A8	A9
R2	3	3	3	3	3	3	1	1	1	1
R5	3	3	3	3	3	3	1	1	1	1
R7	3	3	3	3	3	3	1	1	1	1
R12	3	3	3	3	3	3	1	1	1	1
R13	3	3	3	3	3	3	1	1	1	1
R14	3	3	3	3	3	3	1	1	1	1
R16	3	3	3	3	3	3	1	1	1	1
R1	1	1	1	1	1	1	4	4	4	4
R3	1	1	1	1	1	1	4	4	4	4
R4	1	1	1	1	1	1	4	4	4	4
R6	1	1	1	1	1	1	4	4	4	4
R8	1	1	1	1	1	1	4	4	4	4
R9	1	1	1	1	1	1	4	4	4	4
R10	1	1	1	1	1	1	4	4	4	4
R11	1	1	1	1	1	1	4	4	4	4
R15	1	1	1	1	1	1	4	4	4	4

Fig-3(f): Final output

4.0 DATA DISCRETIZATION

Discretization is critical to the proposed solution, as it reduces the complexity of the problem. Discretization is achieved by partitioning continuous variables or attributes into discrete values or categories. Recognize that weighted graph corresponding to S will always be a clique with quadratic number of edges. Working with S will force considering each and every edge of the un-discretized graph, hence a highly undesirable $\Omega(n^2)$ time complexity. Thus the viable way out is discretization of S leading to S_B. Note that even algorithms with quadratic time complexities are unacceptable for most KDDM (Knowledge Discovery by Data Mining) applications according to Fayyad and Uthurusamy [6].

As this work deals with unsupervised clustering, therefore, supervised discretization methods such as Fuzzy discretization, Entropy Minimization discretization etc will not be considered. Instead unsupervised discretization methods will be covered i.e. making use of information about the distribution of values of attributes without class information. For a detailed comparison of different discretization methods please see Yang and Webb [15]. The discretization methods applicable to the problem are discussed in the subsequent sub-sections.

4.1 Equal Width Discretization (EWD)

EWD divides the number line between v_{min} and v_{max} into k intervals of equal width. Thus the intervals have width $w = (v_{max} - v_{min})/k$ and the cut points are at $v_{min} + w$; $v_{min} + 2w \ldots v_{min} + (k - 1)w$. k is a user predefined parameter and is set as 2 in this work. Here v_{min} and v_{max} are minimum and maximum values, respectively.

4.2 Equal Frequency Discretization (EFD)

EFD divides the sorted values into k intervals so that each interval contains approximately the same number of instances. Thus each interval contains $n=k$ (possibly duplicated) adjacent values. k is a user predefined parameter and is set as 2 in this work.

Note that for small values of k more information about the original data is ignored. On the other hand, large values of k will result in too few points per interval to get a reasonable estimate of the frequency of each interval. Both EWD and EFD are deemed simplistic, and potentially suffer from critical information loss due to the formation of inappropriate interval boundaries, since k is determined without reference to the properties of the given data. Furthermore both techniques are vulnerable to outliers that may drastically skew the range.

5.0 CROSSING MINIMIZATION HEURISTICS

Crossing minimization problem has been studied for over two decades, and its two variants i.e. one-layer and two layers are known to be NP-Complete problems, please see Garey and Johnson [7]. There are basically three types of crossing minimization heuristics, (i) the classical ones that do not count the crossings, hence are very fast and yet give good results such as BaryCenter Heuristic BC by Sugiyama et al [14] and Median Heuristic (MH) by Eades and Wormald [5]. Then there are (ii) crossing counting heuristics that use a diverse set of approaches, such as greedy, tabu-search, simulated annealing, genetic algorithms, etc. and work by counting the number of crossings, hence carry the counting overhead. Lastly (iii) meta-heuristics i.e. use the classical heuristics to generate the initial

arrangement of vertices, and then improve upon it using other heuristics. For a detailed study see Marti and Laguna [12].

5.1 MaxSort Heuristic

It reorders the vertices on the changeable layer according to the MaxSort weight [1] or representative value defined as follows:

$$MaxSort(v) = \begin{cases} Max(P(w)) & \text{if } |\text{adj(v)}| \neq 0 \\ 0 & \text{otherwise} \end{cases}$$

Here v is a vertex on the changeable layer, $adj(v)$ is the set of neighbors of v on the fixed layer, and P(w) is the position of w on the fixed layer. In this work, instead of using MaxSort i.e. sorting on maximum value, its variant MinSort (MS) will be used i.e. sorting on minimum value, as it provides better results as compared to MaxSort. In MS* a variant of MS, vertices with same MinSort weight are sorted in ascending order based on |adj(v)|. Note that for BC and MH, vertices on the changeable layer are reordered based on their barycenter (average) and median values, respectively.

Example-2

In this example working of the MS heuristic is demonstrated for a simple bi-graph.

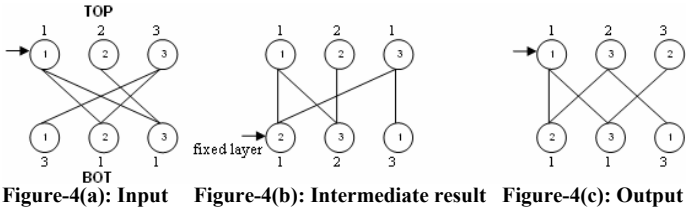


Figure-4(a): Input **Figure-4(b): Intermediate result** **Figure-4(c): Output**

Figure-4(a) shows a bi-graph with 5 crossings. First the vertices in the fixed layer (indicated by the arrow) are assigned ordering i.e. 1, 2, and 3. Subsequently the orderings for the changeable layer are generated based on the minimum value resulting in reordering of vertices to 2, 3 and 1 as shown in Figure-4(b) BOT bipartition. This process is repeated till equilibrium is reached i.e. no further change in orderings. The final non-optimal output is shown in Figure4(c); with two crossings.

For a G_B numbering the vertices in the TOP and subsequently sorting vertices in the BOT, and then repeating this process for the BOT has the effect of two forces that are alternated until an equilibrium state is reached. An equilibrium state corresponds to the final output that does not change with further application of that CMH.

5.2 How biclustering occurs

As a consequence of crossing minimization the edge lengths are decreased, and vertices come closer. If arbitrary vertices come close, then edge length will not decrease. Meaning, vertices with high interconnectivity come together in a neighborhood, thus enhancing clustering. The effect of noise is "moving" the vertices in the "wrong" neighborhood. Therefore, need to have robust estimators of position. Although Median is a robust estimator, and works well too, but the corresponding MH is slow. Thus the performance deteriorates for large problems.

6.0 BICLUSTER QUANTIFICATION

To quantify the biclustering solution for a bicluster Ci with r_i rows and a_i attributes the following notation is proposed:

Let the total number of 1's present in a cluster be T_{i1}
Let the maximum number of 1's in a cluster be M_{i1} (= r_i x a_i)
Let the maximum possible number of 1's be P_{i1} (= α M_{i1})
The accuracy of the bicluster extracted will be $A_i = T_{i1}/P_{i1}$ represented as a percentage.

Hence the accuracy of the solution will be the weighted sum of A_i for all i divided by sum of T_{i1} for all i. Note that the percentage can be greater than 100% too, since P_{i1} is a probable measure, as the actual values are generated randomly.

In the following analysis, only those bicluster extractions are considered for which the bicluster extracted only consist of elements from a single bicluster. Based on the above bicluster quantification criterion, the following table is generated for simulated data consisting of $K_{32,32} \cup K_{64,32} \cup K_{64,128}$

Noise α	0%	5%	10%	15%	20%	25%
Accuracy	100	100	100	99.65	87.72	73.21

Table-1: Quantification of biclustering under noise

7.0 COMPLEXITY ANALYSIS OF CRA_CMH

Cheng and Church [4] were the first to apply biclustering to gene expression data. Given a data matrix S a maximum acceptable mean squared residue score $\delta > 0$ was used, the goal was to find δ-biclusters i.e. subsets of rows and subsets of columns with a score no larger than δ.

This goal was achieved by using several greedy row/column removal/addition algorithms that were then combined into an overall approach that made it possible to find a given number of δ-biclusters. The obvious disadvantage of their technique being large number of iterations, which in turn translates to a slow solution. Figure-5(b) shows the randomly permuted input with α = 10% while Fig-5(b) shows the output using the Cheng and Church (C&C) technique. Interestingly, for α = 0% Cheng and Church technique required several iterations before termination, while our technique terminated in just a single iteration!

Figure-5(c) shows perfect biclustering using MH. The time taken was orders of magnitude less than the Cheng and Church technique. Note that biclustering is represented by grouping of 1's.

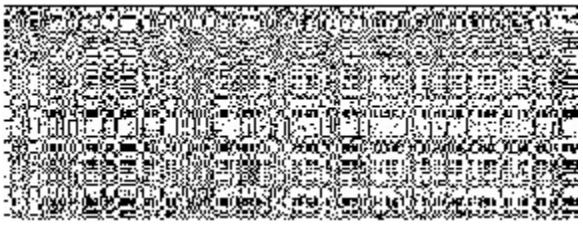


Fig-5(a): Randomly permuted input data matrix

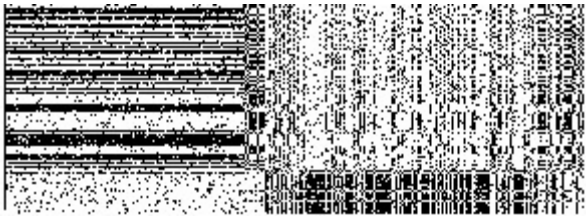


Fig-5(b): Bi-clustered output using C&C with 300 iterations

Fig-5(c): Bi-clustered output using crossing minimization

8.0 APPLICATIONS

Cheng and Church [4] applied biclustering to two gene expression data matrices, specifically to the Yeast Saccharomyces Cerevisiae cell cycle expression data with 2,884 genes and 17 conditions and the human B-cells expression data with 4,026 genes and 96 conditions.

Dhillon [10] used biclustering to perform simultaneous clustering of documents and words by considering a word-by-document matrix with the rows corresponding to words, the columns to documents, and a non-zero element indicating the presence of word in the document.

Movie recommendation data matrices have been biclustered to recommend movies to interested viewers [11].

9. CONCLUSIONS

Biclustering is a hard, yet interesting problem with diverse applications. There are a number of ways of performing biclustering, but this work is a first attempt at using the crossing minimization paradigm. Our proposed technique works well as compared to the traditional biclustering technique recently developed by Cheng & Church, requiring relatively fewer and faster iterations. In addition, we have also utilized a white noise based model to demonstrate the robustness of the proposed clustering technique.

References

[1] Ahsan Abdullah, 1993, "On placement of logic schematics", in proc. of IEEE TENCON'93, Beijing, China, pp. 885-888.

[2] Ahsan Abdullah and Stephen Brobst, 2003, "Clustering by recursive noise removal" with S. Brobst, in proceedings Atlantic Symposium on Computational Biology and Genome Informatics (CBGI'03) Cary, North Carolina, USA, pp. 973-977.

[3] G. D. Battista, P. Eades, R. Tamassia, I. G. Tollis, 1999, Graph Drawing: Algorithms for the Visualization of Graphs, Prentice Hall.

[4] Yizong Cheng and George M. Church, 2000, "Biclustering of expression data", in Proceedings of the 8th International Conference on Intelligent Systems for Molecular Biology (ISMB'00), pages 93–103.

[5] P. Eades, N. Wormald, 1986, The Median Heuristic for Drawing 2-layers Networks, Technical Report 69, Department of Computer Science, University of Queensland, Brisbane, Australia.

[6] U. Fayyad, and R. Uthurusamy, 1996, Data Mining and Knowledge Discovery in Databases, Comm. ACM, vol. 39, no. 11, pp. 24-27.

[7] M. R. Garey and D. S Johnson, 1983, Crossing number is NP-Complete, SIAM J. Algebraic Discrete Methods, 4(1983), 312-316.

[8] Han and Kamber, 2000, Data Mining: Concepts and Techniques, Pub Morgan Kaufmann Publishers.

[9] J. A. Hartigan. Direct clustering of a data matrix. Journal of the American Statistical Association (JASA), 67(337):123–129, 1972.

[10] Inderjit S. Dhillon, 2001, "Co-clustering documents and words using bipartite spectral graph partitioning", in Proceedings of the 7th ACMSIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'01), pages 269–274.

[11] Jiong Yang, Wei Wang, Haixun Wang, and Philip Yu., 2002, Æ-clusters: Capturing subspace correlation in a large data set. In Proceedings of the 18th IEEE International Conference on Data Engineering, pages 517–528.

[12] Marti R., and Laguna M., 2001, Heuristics and Meta Heuristics for 2-layer Straight Line Crossing Minimization, Discrete Applied Mathematics, Vol. 127 - Issue 3, pp. 665 – 678.

[13] Sara C. Madeira and Arlindo L. Oliveira, 2004, "Biclustering Algorithms for Biological Data Analysis: A Survey", IEEE/ACM Transactions on Computational Biology and Bioinformatics.

[14] K. Sugiyama, S. Tagawa and M. Toda, 1981, Methods for Visual Understanding of Hierarchical Systems. IEEE Trans. Syst. Man Cybern., SMC-11(2):109-125.

[15] Y. Yang and G. I. Webb, 2002, A Comparative Study of Discretization Methods for Naive-Bayesian Classifiers, in proc. of Pacific Rim Knowledge Acquisition Workshop, National Center of Sciences, Tokyo, Japan.