Application of Simulated Annealing to the Biclustering of Gene Expression Data

Kenneth Bryan, Pádraig Cunningham, and Nadia Bolshakova

***Abstract*—In a gene expression data matrix, a bicluster is a submatrix of genes and conditions that exhibits a high correlation of expression activity across both rows and columns. The problem of locating the most significant bicluster has been shown to be NP-complete. Heuristic approaches such as Cheng and Church's greedy node deletion algorithm have been previously employed. It is to be expected that stochastic search techniques such as evolutionary algorithms or simulated annealing might improve upon such greedy techniques. In this paper we show that an approach based on simulated annealing is well suited to this problem, and we present a comparative evaluation of simulated annealing and node deletion on a variety of datasets. We show that simulated annealing discovers more significant biclusters in many cases. Furthermore, we also test the ability of our technique to locate biologically verifiable biclusters within an annotated set of genes.**

***Index Terms*—Biclustering, data mining, gene expression, simulated annealing.**

I. Introduction

In RECENT years, the advent of DNA microarray technologies has revolutionized gene expression analysis. It is now possible to monitor the expression of thousands of genes in parallel over many experimental conditions (e.g., different patients, tissue types, and growth environments), all within a single experiment (see Lander [1]). The results from these experiments are usually presented in the form of a data matrix in which rows represent genes and columns represent conditions. Each entry in the matrix is a measure of the expression level of a particular gene under a specific condition. Thorough analysis of these datasets aids in the annotation of genes of unknown function and the discovery of functional relationships between genes. This ultimately contributes to the elucidation of biological systems at a molecular level [2].

Gene expression datasets typically contain thousands of genes and hundreds of conditions, and mining functional and class information from such large volumes of data is a far from trivial. One of the main methods used thus far to investigate the underlying structure of gene expression datasets has been *cluster analysis* [3]–[5]. In this approach, genes showing similar expression activity over the set of conditions are grouped together into clusters. The premise behind this is that similarly behaving genes may be coregulated and share a related function; i.e., belong to a common pathway or a cellular structure. Conditions may also be clustered, enabling disease types such as cancers to be defined in terms of their unique expression profiles [6]. Gene expression datasets are continually growing in size as more experiments are carried out, and as experimental capacity improves. As datasets increase size, it becomes less likely that objects (genes) will retain similarity across all attributes (conditions), making clustering problematic. Furthermore, it is not uncommon for the expression of genes to be highly similar under one set of conditions, and yet independent under another set [7]. Clustering genes over a subset of similar conditions would be more beneficial in such cases. This approach has been termed *biclustering* and was first introduced to gene expression analysis by Cheng and Church [8]. Cheng and Church identified the problem of finding significant biclusters as being NP-Hard and employed a greedy node deletion algorithm in their search. The review of biclustering algorithms for biological data analysis presented by Madeira and Oliveira [9] also identifies greedy search algorithms as a promising approach. Greedy search algorithms start with an initial solution and find a locally optimal solution by successive transformations that improve some fitness function. Stochastic methods such as simulated annealing (SA) [10] improve on greedy search due to their potential to escape local optima (see Section III). In this paper, we present a biclustering technique based on SA that improves the results produced by Cheng and Church's node deletion algorithm (see Section II). We carry out a comparative evaluation using both synthetic and real gene expression datasets, and show that our SA based approach finds more significant biclusters in each dataset (see Section V). We then use our SA algorithm to analyze an annotated set of genes with a view to discovering biologically verifiable biclusters.

II. Biclustering

In general, biclustering refers to the "simultaneous clustering" of both rows and columns of a data matrix [11]. Hartigan pioneered this type of analysis in the seventies using two-way analysis of variance to locate constant valued submatrices within datasets. Biclustering may be viewed as a more specific type of subspace clustering that enforces correlation within a subset of features (conditions) as well as a subset of objects (genes). This approach suits the gene expression context, as related genes are thought to be regulated in a synchronized fashion and over certain conditions [7]. Therefore, discovering the dominant biclusters within a gene expression dataset may aid the discovery of these coregulated groups. More recently, inspired by Hartigan's so called "direct clustering" approaches [12], the concept was introduced to the area of gene expression analysis by [8]. Since then, several alternative biclustering approaches have been taken within gene expression analysis. One approach, taken by Tanay *et al.* [13], likens biclustering to the search

Manuscript received August 30, 2005; revised December 19, 2005. This work was supported by Science Foundation Ireland under Grant SFI-02/IN1/I111.

The authors are with The Machine Learning Group, Faculty of Computer Science, Trinity College Dublin, Dublin 2, Ireland (e-mail: Kenneth.Bryan@cs.tcd.ie; Padraig.Cunningham@cs.tcd.ie; Nadia.Bolshakava@cs.tcd.ie).

Digital Object Identifier 10.1109/TITB.2006.872073

for complete subgraphs within a bipartite graph. They develop a statistical model of the expression data matrix and propose an heuristic search based on discovering statistically significant subgraphs. Lazzeroni and Owen [14] developed what they termed a *plaid model* in which the dataset is represented by a linear function of variables or *layers*, which correspond to biclusters. Another approach, taken by Kluger *et al*. [15], involves decomposing the data matrix into its principle components by singular value decomposition. The resulting eigenvectors are then used to reorder the data matrix to reveal the set of biclusters as a *checkerboard* structure. These approaches are less intuitive and theoretically quite different from that of [8]. For a review of the preceeding approaches, the reader is directed to [9].

Cheng and Church defined a bicluster to be a subset of genes and a subset of conditions with a high similarity score, where similarity is a measure of the coherence of genes and conditions in the subset. A group of genes are said to be coherent if their level of expression reacts in parallel or correlates across a set of conditions. Similarly, a set of conditions may also have coherent levels of expression across a set of genes. Cheng and Church developed a measure, called the *mean squared residue score*, which takes into account both row and column correlations, and therefore makes it possible to simultaneously evaluate the coherence of rows and columns within a submatrix. They thus defined a bicluster to be a submatrix composed of subsets of genes and conditions with a low mean squared residue score (the lower the score, the better the correlation of the rows and columns). The residue score of an entry a_{ij} in a bicluster $B(IJ)$ (where I is the subset of rows, and J is the subset of columns making up the bicluster) is a measure of how well the entry fits into that bicluster. It is defined as

$$R(a_{ij}) = a_{ij} - a_{Ij} - a_{iJ} + a_{IJ} \tag{1}$$

where a_{iJ} is the mean of the ith row in the bicluster, a_{Ij} is the mean of the jth column, and a_{IJ} is the mean of the whole bicluster. The overall *mean squared residue score* is

$$H(I,J) = \frac{1}{|I||J|} \sum_{i \in I, j \in J} R(a_{ij})^2. \tag{2}$$

The next problem to be tackled is how to locate the low scoring biclusters within a parent data matrix. The deterministic approach is to sequentially run through all the possible combinations of rows and columns of the data matrix and find the sub-matrices which satisfy a predefined low score, δ (the set of δ-biclusters). The most significant biclusters, the largest δ-biclusters, would be of most interest as they capture the relationships between the largest number of objects. However, the number of possible submatrices increases exponentially with the size of the parent matrix, making this task practically impossible when the matrix exceeds the fairly modest size of a few hundred elements. Cheng and Church likened the maximum bicluster search to that of locating a maximum biclique (largest complete subgraph) within a parent bipartite graph, which has been proven to be NP-Hard [16]. Biclustering based upon this graph theoretic paradigm was more fully developed in other studies [13]. Cheng and Church designed a set of heuristic algorithms to locate δ-biclusters sequentially in a top-down manner by deleting the row and column nodes from the parent matrix, which most improved the mean squared residue score. Upon reaching the δ threshold, a node addition phase is then carried out to add rows/columns which may have been missed. Inversely correlated rows, which may represent negatively regulated genes, are also added at this stage. A subsequent study [17] noted that, as with other greedy searches, there is a possibility that the system may become trapped at a locally good solution. It is thus unlikely that the *global maximum*, or maximal δ-bicluster, will be found. Applying a stochastic search technique to locate this global maximum seems to be the next logical step in the bicluster search problem.

III. Simulated Annealing

Stochastic techniques which accept reversals in fitness have been shown to improve on greedy approaches by performing more in-depth searches of the solution space. Recently, evolutionary optimization schemes employing the mean squared residue function have been used to tackle the bicluster search problem [18], [19]. These attempts failed to find more significant solutions than the Cheng and Church technique in terms of bicluster size, and instead focused on returning sets of smaller biclusters with high row variability.

Simulated annealing is a well established stochastic technique originally developed to model the natural process of crystalization [20] and later adopted to solve optimization problems [10]. As with a greedy search, it accepts all changes that lead to improvements in the fitness of a solution. However, it differs in its ability to allow the probabilistic acceptance of changes which lead to worse solutions; i.e., reversals in fitness. The probability of accepting a reversal is inversely proportional to the size of the reversal with the acceptance of smaller reversals, being more probable. This probability also decreases as the search continues, or as the system cools, allowing eventual convergence on a solution. It is defined by Boltzman's equation

$$P(\Delta E) \propto e^{-\frac{\Delta E}{T}} \tag{3}$$

where ΔE is the difference in energy (fitness) between the old and new states, and T is the temperature of the system. In the virtual environment, the temperature of the system is lowered after certain predefined number of accepted changes, successes, or total changes, attempts, depending on which is reached first. The rate at which temperature decreases depends on the cooling schedule. In the natural process, the system cools logarithmically; however, this is so time consuming that many simplified cooling schedules have been introduced for practical problem solving. The following simple cooling model is popular:

$$T(k) = \frac{T(k-1)}{1+\sigma} \tag{4}$$

where $T(k)$ is the current temperature, $T(k-1)$ is the previous temperature, and σ dictates the cooling rate, simulated annealing has been applied to such problems as the well-known traveling salesman problem [21] and optimization of wiring on computer

chips [10]. Its application to biclustering gene expression data is a logical step, given the drawbacks of current approaches.

IV. Experimental Methods

A. Biclustering Using Simulated Annealing

Several parameters are common to every simulated annealing implementation. The most crucial parameter is the fitness function, or how to quantitatively discern whether or not the solution improves after a perturbation. The mean squared residue score was used as a measure of bicluster fitness in this study. In simulated annealing, it is also important to ensure that a search of sufficient depth is performed at each temperature. As mentioned in Section III, this is dictated by the predfined number of attempts or successes that must occur before each reduction in system temperature. This selection depends upon the depth and size of the search space as determined by the size and dimensionality of the dataset. In this study, the number of successes needed to be achieved before cooling occurs was set at ten times the number of genes. The number of attempts at each temperature is ten times this again (this ensures a good search even if there are not the required number of successes). So for a dataset of 1000 genes, the system would only lower the temperature after 10 000 successes or 100 000 attempts. The rate at which the temperature is lowered, the annealing schedule, was of the type shown in (4) with $\sigma = 0.1$. Consequently, each subsequent temperature is approximately 0.9 times that of the previous temperature. Another important parameter is the initial temperature of the system t_0. If this parameter is set too high, the system will take too long to converge, and if it is set too low, the proportion of the search space covered will be much reduced. It has been found by experiment that, in general, an optimal starting temperature is one which allows 80% of reversals to be accepted [22].

Our simulated annealing biclustering (SAB) algorithm is illustrated in Fig. 1. It begins the search in a top-down manner with the initial solution x_0, containing all rows and columns in the input data matrix M. The current solution x, is then iteratively perturbed by deletion or addition of rows or columns to give a new solution, x_{new}. The method for generating x_{new} is given, and takes into account the ratio of rows to columns in the current solution and adjusts the probability of a row or column flip accordingly. So, for example, if there are 100 columns and ten rows in a current solution, the probability of choosing a row to flip is $1/10$. In the random row flip method, it can be seen that if the row size of the solution row equals the minimum row size threshold $\text{row}_{\min}$, and if a perturbation calls for a row to be deleted, it is replaced by another randomly chosen row from the data matrix. The same also occurs in the equivalently coded random column flip method. This allows perturbance of the solution while maintaining the row and column sizes above their respective minimum size limits. In our implementation, a minimum solution size of 10×10 was chosen. This was deetend to represent the minimum significant size of a solution in this study. So, for example, if genes correlate over 10 conditions, it is likely that they may be related. This minimum solution size also prevents the search from ending on a trivial bicluster of one row or one column and score 0. After x_{new} is generated, it is scored using the mean squared residue fitness function f. As discussed in Section II, the lower the mean squared residue score, the better the row and column correlation of a solution. This accounts for the less than sign in the if statement on line six of the SAB code. Lines eight and nine in SAB then cover

Variable definitions: x_0 : initial solution, x : current solution, t_0 : initial temperature, t : current temperature, t_{Min} : finish temperature, *rate* : temperature fall rate, a : attempts, s : successes, a_{count} : current number of attempts, s_{count} : current number of successes, f : fitness function, M : datamatrix, row_x : rows in current solution, col_x : columns in current solution, row_{min} : minimum row size threshold, col_{min} : minimum column size threshold, δ : mean squared residue threshold.

SAB($f, x_0, t_0, rate, row_{min}, col_{min}, s, a, M, \delta$)

1. $x \leftarrow x_0$
2. $t \leftarrow t_0$
3. while($t > t_{Min}$)
4. While($a_{count} < a$ AND $s_{count} < a$)
5. $x_{new} \leftarrow$ GenerateNewSolution($M, x, row_{min}, col_{min}$)
6. if$f(x_{new}) < f(x)$
7. then $x \leftarrow x_{new}$
8. else if exp($-\frac{\Delta E}{T}$)> random(0,1)
9. then $x \leftarrow x_{new}$
10. if ($f(x_{new}) \leq \delta$ AND size of x_{new} > size of x)
11. then $col_{min} \leftarrow$ columns in x_{new}
12. then $row_{min} \leftarrow$ rows in x_{new}
13. $t \leftarrow$ Cool(t,*rate*)
14. $x \leftarrow$ NodeAddition(x)
15. return x

GenerateNewSolution($M, x, row_{min}, col_{min}$)

1. $row_x \leftarrow$ rows in x
2. $col_x \leftarrow$ columns in x
3. if($row_x \geq col_x$)
4. then generate random $r \in \mathbb{R}$, range$[0, \frac{row_x}{col_x}]$
5. if ($r = 0$)
6. then $x_{new} \leftarrow$ FlipRandomColumn(M, x, col_{min}, col_x)
7. else
8. then $x_{new} \leftarrow$ FlipRandomRow(M, x, row_{min}, row_x)
9. else if($row_x < col_x$)
10. then generate random $r \in \mathbb{R}$, range$[0, \frac{col_x}{row_x}]$
11. if ($r = 0$)
12. then $x_{new} \leftarrow$ FlipRandomRow(M, x, row_{min}, row_x)
13. else
14. then $x_{new} \leftarrow$ FlipRandomColumn(M, x, col_{min}, col_x)
15. return x_{new}

FlipRandomRow(M, x, row_{min}, row_x)

1. choose random row, row_{rand}, from M
2. if $row_{rand} \notin x$
3. then add row_{rand} to x to give x_{new}
4. else if $row_{rand} \in x$ AND $row_x > row_{min}$
5. then remove row_{rand} from x to give x_{new}
6. else if $row_{rand} \in x$ AND $row_x = row_{min}$
7. then remove row_{rand} from x
8. choose random row, $row_{rand'} \neq row_{rand}$ AND $\notin x$
9. add $row_{rand'}$ to x to generate x_{new}
10. return x_{new}

Fig. 1. SAB algorithm. The FlipRandomColumn method is not detailed, as it is equivalent to the FlipRandomRow method.

the standard simulated annealing probabilistic acceptance of the new solution x_{new}, as discussed in Section III.

To allow the comparison of SAB with the node deletion algorithm, some way is needed to return biclusters of a chosen δ value. This is detailed in lines 10–12 of the SAB. Upon reaching a solution with a score equal or lower than δ and larger in size, the minimum row and column limits are reset to that size. As the search continues, these opposing provisos of increased size and less than or equal to δ-score keeps the current solution around the δ-threshold while allowing room for growth in size of the δ-bicluster. To align SAB and the Cheng and Church node deletion approach, their node addition is performed after the SAB search. This adds any missed rows or columns and, importantly for SAB alignment, adds inversely correlated rows which may represent negatively regulated genes. Cheng and Church masked each δ-bicluster solution with randomly imputed numbers from the same range as the dataset. This prevents the bicluster from being rediscovered by the deterministic node deletion algorithm. We use the same method of masking discovered solutions. Typically, using the preceeding parameters and for a dataset of 3000 genes and 20 conditions, the search takes about 60 min to converge on a bicluster solution. As a result of the masking of solutions, this convergence time is reduced for subsequent bicluster searches.

B. Datasets Used

Cheng and Church chose a yeast cell cycle dataset[1] in their study. This dataset contains 2884 genes and 17 conditions. We used this dataset and two additional real datasets to compare our SAB algorithm with node deletion. The first additional dataset contains 27 conditions and 2774 genes.[2] It is derived from a study on scleroderma, a potentially serious skin disorder which affects epithelial cells [23]. This dataset contains gene expression data from both normal and affected patients. The second additional dataset of 3051 genes and 38 conditions, representing different classes of lymphoma, was distilled from a larger dataset [24] using techniques described in [25] to enrich the dataset with genes with the highest variance across conditions. A synthetic dataset was also used to compare the algorithms. The attraction of a synthetic dataset is that all the major biclusters can be defined and embedded in the data. The success of bicluster discovery can then be more quantitively measured. We have developed a synthetic dataset construction technique, and believe it to be a more faithful rendering of reality than previous approaches [26]. A dataset of size 100×100 was constructed. Biclusters were generated using real gene profiles from the yeast dataset as templates. Firstly, an expression level shift was added to the template gene profile vector. The amount of shift is chosen randomly for each additional artificial profile and maintained within a user defined range. This resulting spread mirrors the expression level variations which occur *in vivo*, and also makes individual expression profiles more discernable within the bicluster. As it stands, the bicluster has a perfect score of 0. Some error needs to be introduced to reflect the *in vivo* model. Each

[1]http://arep.med.harvard.edu/biclustering/yeast.matrix

[2]http://genome-www.stanford.edu/scleroderma/data.shtml

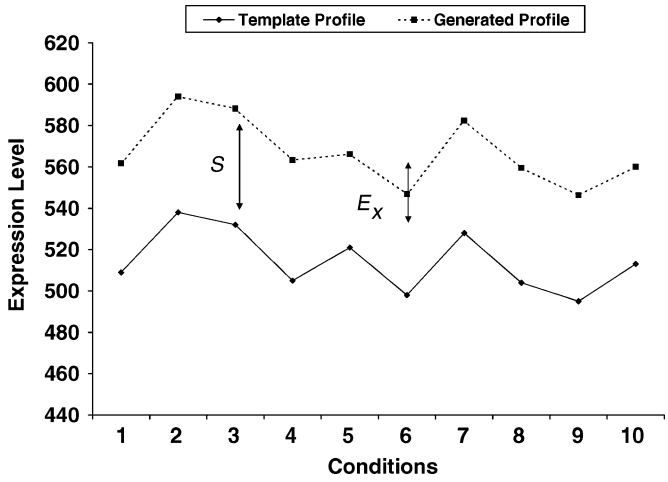

Fig. 2. Illustration of template profile and generated profile. S represents the shift applied to the profile and E_x represents the correlation error applied to each value.

expression value in each artificial profile is then augmented by a correlation error (E_x) such that

$$E_x = \sigma(\mathbf{x}) \cdot \epsilon \cdot r_x \tag{5}$$

where $\sigma(\mathbf{x})$ is the standard deviation of the template gene profile (this scales the error for the particular template), ϵ is a user defined constant in a range [0,1] (this variable dictates the level of error and the quality of the biclusters), and r_i is a random variable in a range $[-1, 1]$ (this enables the expression level of the generated profile to be greater or less than the template). Given that the original gene profile template is defined as

$$\mathbf{X} = \{x_1, \ldots, x_n\} \tag{6}$$

the newly constructed correlating profile will be given as

$$\mathbf{Y} = \{x_1 + (S + E_1), \ldots, x_n + (S + E_n)\} \tag{7}$$

where x represents a particular expression value, S is the shift applied to the vector, and E is the correlation error applied to each correlating expression value. An illustration of the template profile and a generated profile is shown in Fig. 2. All five biclusters, were constructed of sizes 10×10, 20×10, 10×20, and two 10×10 overlapping biclusters, and embedded in a background randomly generated within the same range as the biclusters (0–600).

V. Evaluation of Biclustering Using Simulated Annealing

Three questions are dealt with in this section. Firstly, we investigate whether SAB can retrieve solutions closer to the global maximum than Cheng and Church's node deletion (ND) approach, i.e., larger δ-biclusters. We then investigate, using a synthetic dataset, the ability of SAB to discover all the bicluster signals within a dataset. Lastly, we use an annotated dataset to investigate whether biclusters discovered by SAB reflect *in vivo* functional modules, i.e., whether SAB can discover biologically verifiable biclusters.

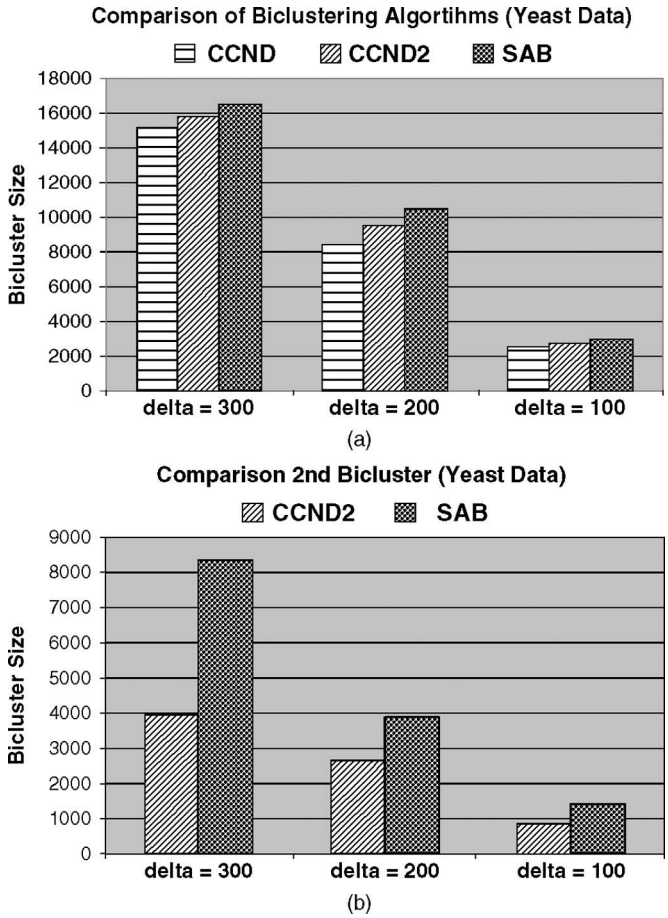

Fig. 3. (a) Comparisons of Cheng and Church's node deletion algorithm (ND), our adjusted node deletion algorithm (ND2) and simulated annealing biclustering (SAB) using the yeast dataset over δ-scores of 300, 200 and 100. (b) The second biclusters found by CCND2 and SAB.

A. Comparative Evaluation With Node Deletion

Cheng and Church carried out node deletion on the previously mentioned yeast dataset and used a mean squared residue threshold (δ) of 300 (as determined by (2)). The SAB algorithm was applied to the same yeast dataset. In this study, δ thresholds of 300, 200, and 100 were set, and the size of the discovered biclusters compared in each case. SAB produces biclusters of at least ten columns (conditions) in width. To ensure that the column size of the resultant biclusters does not bias the results, an adjusted node deletion algorithm (ND2) is also run where in the column size of resultant biclusters is set to ten. This is achieved in ND2 by prioritizing column deletion until the minimum threshold of 10 has been reached whereupon row deletion alone is carried out. Fig. 3(a) shows the size of the first bicluster found by ND, ND2, and SAB over the various δ thresholds for the yeast dataset. Fig. 3(b) shows the second bicluster discovered when the first was masked with random numbers, as described in Section IV. SAB performed better than ND and ND2 on the yeast data for all δ-scores, locating larger δ-biclusters in all cases. The results for all three data sets are shown in Table I. The numbers shown in bold mark the best biclusters. Biclusters in italics are taken from a significantly larger dataset (after masking a smaller bicluster 1 solution) and cannot be compared

TABLE I
COMPARISON OF BICLUSTERS DISCOVERED IN EACH REAL DATASET

		ND	ND2	SAB	ND	ND2	SAB
	δ	Bicluster 1			Bicluster 2		
Yeast	300	15165	15750	**16460**	*9012*	3930	**8320**
	200	8463	9540	**10360**	*4972*	2630	**3860**
	100	2520	2700	**2940**	*1260*	830	**1390**
Scleroderma	300	13590	**18260**	**18230**	*4320*	**6780**	6310
	200	7296	12920	**13210**	*7876*	3290	**4030**
	100	2730	**5170**	**5140**	*1570*	830	**850**
Lymphoma	300	1344	**3320**	3220	*518*	1740	**1810**
	200	1032	**2510**	2460	*300*	**1370**	1200
	100	851	**1780**	**1790**	*136*	**1050**	810

TABLE II
BICLUSTERS RECOVERED FROM SYNTHETIC DATASET

Biclusters Embedded	Solutions Recovered (rows×columns)	
	Node Deletion	SAB
A(10×10)	3×2(6%)	10×10(100%)
B(20×10)	21×9(90%)	21×10(100%)
C(10×20)	8×15(60%)	11×17(94%)
D(10×10)	4×6(24%)	10×10(100%)
E(10×10)	6×10(60%)	10×1(0100%)

with neighboring values in the second bicluster column. SAB performed better than ND, discovering larger δ-biclusters in all cases. The ND2 algorithm performed better than the original ND, but even so, SAB still performed better in most cases. It can be seen that SAB performs better than ND2, discovering a larger first bicluster in four out of nine cases, and draws in an additional three cases. SAB performed best in discovering the second biclusters in six out of nine cases over the three datasets.

B. Bicluster Retrieval in Synthetic Data

It has been shown in the previous section that SAB has the ability to retrieve more significant biclusters than node deletion. However, it is difficult to know how successful SAB, is in recovering all the significant signals in this real data. To test this aspect of SAB, we decided to use a synthetic dataset constructed in the manner described in Section IV-B. We then carried out a comparative evaluation of SAB and node deletion using this dataset, and measured their abilities to recover the five embedded biclusters. From Table II, it can be seen that both node deletion and SAB discovered five bicluster signals within the synthetic dataset, but SAB retrieves substantially more of the embedded biclusters.

C. Biological Interpretation

A further way to evaluate SAB is to use a fully annotated dataset. To our knowledge, this precise approach to bicluster evaluation has not been used before in the literature. Of the 2884 genes in the yeast dataset, 550 can be annotated from the online database called the Kyoto Encyclopaedia of Genes and Genomes, KEGG.[3] Ideally, the biclusters in such a dataset, would then reflect *in vivo* groups of genes known to be functionally related. Because of the smaller size of the

[3]http://www.genome.jp/kegg/genes.html

TABLE III
KNOWN FUNCTIONAL MODULES (FM) FOUND BY SAB IN THE ANNOTATED GENE DATASET

Bic.	# Genes	Dominant FM	Genes in FM	P-value
1	81	Ribosomal Proteins(96)	61	$< 1 \times 10^{-15}$
		Glycolysis/Glucogenisis(26)	5	0.17
2	59	Basal Transcription Factors(10)	6	1.08×10^{-7}
		Nucleotide Metabolism(81)	16	2.3×10^{-3}

annotated dataset, a δ-score of 100 was chosen as the mean squared residue threshold. In Table III, it can be seen that the first bicluster discovered from this annotated dataset is rich in genes from the ribosomal functional category. The second bicluster contains transcription factors and genes involved in nucleotide metabolism. These genes are the main regulators of protein production and gene expression in the cell. The statistical significance of discovering each functional category is given in terms of *p-values*, as formulated in [27]. Further biclusters contained correlating genes but no dominating known functional categories, so they are not listed. This may be partly due to the incomplete nature of the dataset, as only the annotated genes were used from the yeast data.

VI. CONCLUSION

Using SAB, we have shown that stochastic methods have the potential to give improved results for the bicluster search problem. SAB discovers more significant biclusters than Cheng and Church's original node deletion approach. The biclusters discovered by SAB are considered more significant as they have the same level of row/column correlation as those discovered by Cheng and Church (as measured by δ), but are larger in size. One could also search for biclusters of the same size and lower δ-score. However, this approach would yield information on fewer genes/conditions, and direct comparison with Cheng and Church's results would not be possible. SAB also performs better when compared to our improved version of the node deletion algorithm. Furthermore, we have shown that SAB discovers more complete biclusters than node deletion when using the synthetic dataset. When applied to the annotated yeast dataset, SAB discovers biclusters which represent recognizable classes of genes. In the annotated dataset, SAB discovered just two biclusters which had an overrepresentation of known functional groups. Apart from the incompleteness of this annotated dataset, as mentioned in Section V-C, this may be due to the manner in which SAB searches for biclusters. SAB works in top-down manner with the mean squared residue function promoting the deletion of rows/columns which do not fit in with the trends in the dataset. As a result, biclusters may be biased toward core regulatory genes which govern the general state of gene expression the cell. Outlying biclusters would tend to have their ill-fitting rows/columns deleted early on in the search. Evidence of this can be seen in the nature of the classes of genes in biclusters 1 and 2 from the annotated set. Perhaps this bias could be harnessed to discover regulatory genes within gene expression data. Although a bottom-up search approach using the mean squared residue as a fitness function would probably not find such large biclusters, it would, perhaps, promote more variability in the classes of genes it discovers. In future research, we intend to use simulated annealing in a bottom-up search in a bid to discover smaller, more natural biclusters which may better reflect the natural state or organization in an organism.

REFERENCES

[1] E. S. Lander, "Array of hope," *Nature Genet.*, vol. 21, pp. 3–4, 1999.

[2] D. Berrer, W. Dubitzky, and S. Draghici, *A Practical Approach to Microarray Data Analysis*. Norwell, MA: Kluwer, 2003, ch. 1, pp. 15–19.

[3] M. B. Eisen, P. T. Spellman, P. O. Brown, and D. Botstein, "Cluster analysis and display of genome-wide expression patterns," *Proc. Nat. Acad. Sci.*, vol. 8, no. 95, pp. 14863–14868, 1998.

[4] S. Tavazoie, J. D. Hughes, M. Campbell, R. J. Cho, and G. M. Church, "Systematic determination of genetic network architecture," *Proc. Nat. Acad. Sci.*, vol. 22, no. 3, pp. 281–285, 1999.

[5] I. S. Dhillon, S. Mallela, and D. S. Modha, "Information theoretic co-clustering," in *Proc. 9th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining*, Washington, DC, 2003, pp. 89–98.

[6] S. L. Pomeroy, P. Tamayo, M. Gaasenbeek, L. M. Sturla, M. Angelo, M. E. McLaughlin, J. Y. Kim, L. C. Goumnerova, P. Black, C. Lau, J. C. Allen, D. Zagzag, J. M. Olson, T. Curran, C. Wetmore, J. A. Biegel, T. Poggio, S. Mukherjee, R. Rifkin, A. Califano, G. Stolovitzky, D. N. Louis, J. P. Mesirov, E. S. Lander, and T. R. Golub, "Prediction of central nervous system embryonal tumour outcome based on gene expression," *Nature*, vol. 24, no. 415, pp. 436–442, 2002.

[7] A. Ben-Dor, B. Chor, R. Karp, and Z. Yakini, "Discovering local structure in gene expression data: The order-preserving submatrix problem," *J. Comput. Biol.*, vol. 10, no. 3–4, pp. 373–384, 2003.

[8] Y. Cheng and G. M. Church, "Biclustering of expression data," in *Proc. 8th Int. Conf. Intelligent Systems for Molecular Biology (ISMB)*, San Diego, CA, 2000, pp. 93–103.

[9] S. C. Madeira and A. L. Oliveira, "Biclustering algorithms for biological data analysis: a survey," *IEEE/ACM Trans. Comput. Biol. Bioinformatics*, vol. 1, no. 1, pp. 24–45, 2004.

[10] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, pp. 671–680, 1983.

[11] B. Mirkin, *Mathematical Classification and Clustering*. Norwell, MA: Kluwer, 1996.

[12] J. A. Hartigan, "Direct clustering of a data matrix," *J. Amer. Statistical Assoc.*, vol. 67, no. 337, pp. 123–129, 1972.

[13] A. Tanay, R. Sharan, and R. Shamir, "Discovering statistically significant biclusters in gene expression data," *Bioinformatics*, vol. 18, pp. 36–44, 2002.

[14] L. Lazzeroni and A. Owen, "Plaid models for gene expression data," *Statistica Sinica*, vol. 12, pp. 61–86, 2002.

[15] Y. Kluger, R. Basri, J. T. Chang, and M. Gerstein, "Spectral biclustering of microarraydata: coclustering genes and conditions," *Genome Res.*, vol. 13, pp. 703–716, 2003.

[16] D. S. Johnsen, "The np-completeness column: an ongoing guide," *J. Algorithms*, vol. 8, pp. 438–448, 1987.

[17] J. Yang, H. Wang, W. Wang, and P. Yu, "Enhanced biclustering on expression data," in *Proc. IEEE 3rd Symp. Bioinformatics and Bioengineering (BIBE)*, Bethesda, MD, 2003, pp. 321–327.

[18] S. Bleuler, A. Prelić, and E. Zitzler, "An EA framework for biclustering of gene expression data," in *Proc. Congr. Evolutionary Comput. (CEC-2004)*, Piscataway, NJ, 2004, pp. 166–173.

[19] J. S. Aguilar-Ruiz and F. Divina, "Evolutionary computation for biclustering of gene expression," in *Proc. 2005 ACM symp. Applied Computing*, Santa Fe, NM, 2005, pp. 959–960.

[20] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, "Equations of state calculations by fast computing machines," *J. Chem. Phys.*, vol. 21, pp. 1087–1092, 1958.

[21] K. Binder and D. Stauffer, *A Simple Introduction to Monte Carlo Simulations and Some Specialized Topics, in Applications of the Monte Carlo Method in Statistical Physics*. Berlin, Germany: Springer-Verlag, 1985, pp. 1–36.

[22] B. Preiss, *Data Structures and Algorithms with Object-Oriented Design Patterns in Java*. New York: Wiley, 1999.

[23] M. L. Whitfield, D. R. Finlay, J. I. Murray, O. Troyanskaya, J.-T. Chi, A. Pergamenschikov, T. McCalmont, P. O. Brown, D. Botstein, and M. K. Connolly, "Systemic and cell type-specific gene expression patterns in

scleroderma," *Proc. Nat. Acad. Sci.*, vol. 100, no. 21, pp. 12319–12324, 2003.
[24] T. Golub, D. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. Mesirov, H. Coller, M. Loh, J. Downing, M. Caligiuri, C. Bloomfield, and E. Lander, "Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring," *Science*, vol. 286, no. 5439, pp. 531–537, 1999.
[25] S. Dudoit and J. Fridlyand, "Bagging to improve the accuracy of a clustering procedure," *Bioinformatics*, vol. 19, no. 9, pp. 1090–1099, 2003.
[26] Q. Sheng, Y. Moreau, and B. D. Moor, "Biclustering microarray data by Gibbs sampling," *Bioinformatics*, vol. 19, no. 2, pp. 196–205, 2003.
[27] S. Draghici, P. Khatri, R. P. Martins, G. C. Ostermeier, and S. A. Krawetz, "Global functional profiling of gene expression," *Genomics*, vol. 81, no. 2, pp. 98–104, 2003.

Kenneth Bryan received the B.A.Mod. (Hons) degree in microbiology from. Trinity College Dublin, Dublin, Ireland, in 2001, and the Graduate Diploma in information technology from Dublin City University, Dublin, Ireland. He is currently working toward the Ph.D. degree of Trinity College Dublin.

His work involves the computational analysis of gene expression data, and developing heuristic models for efficient biclustering of gene expression data.

Pádraig Cunningham received the B.E. and M.Eng.Sci. degrees from the National University of Ireland, Galway, Ireland, in 1983 and 1985, respectively, and the Ph.D. degree in 1989 from Dublin University, Dublin, Ireland.

He is currently an Associate Professor in the Department of Computer Science, Trinity College, Dublin, Ireland (TCD). His research focus is on machine learning—specifically, case-based reasoning, clustering, and ensemble methods. After completing the Ph.D. degree, he worked with Digital Equipment Corporation as a Software Engineer, and with Hitachi Europe Ltd. as a Research Scientist.

Dr. Cunningham became a Fellow of TCD in 1998, and was elected to Fellowship on the European Coordinating Committee for Artificial Intelligence Research (ECCAI) in 2004.

Nadia Bolshakova received the Ph.D. degree from the State Technical University, Russia, in 2000.

She is currently a Research Fellow in the Department of Computer Science, Trinity College Dublin, Dublin, Ireland. Her research interests include the intersection of computer science and the life sciences such as bioinformatics and intelligent systems; and developing methods, tools, and systems to support research on microarrayi data and protein secondary structure prediction. She took part in research projects with the Institute of Cytology of the Russian Academy of Sciences. In addition, she has published articles in journals, conference proceedings, and books relating to bioinformatics, biomodeling, and cell biology, and has been a referee for several scientific journals.