Available online at www.sciencedirect.com
ScienceDirect
Information Sciences 178 (2008) 1479–1497
INFORMATION SCIENCES
AN INTERNATIONAL JOURNAL
www.elsevier.com/locate/ins

Random walk biclustering for microarray data

Fabrizio Angiulli, Eugenio Cesario, Clara Pizzuti *

ICAR-CNR, Via P. Bucci 41C, 87036 Rende, CS, Italy

Received 20 March 2007; received in revised form 16 November 2007; accepted 16 November 2007

Abstract

A biclustering algorithm, based on a greedy technique and enriched with a local search strategy to escape poor local minima, is proposed. The algorithm starts with an initial random solution and searches for a locally optimal solution by successive transformations that improve a gain function. The gain function combines the mean squared residue, the row variance, and the size of the bicluster. Different strategies to escape local minima are introduced and compared. Experimental results on several microarray data sets show that the method is able to find significant biclusters, also from a biological point of view.
© 2007 Elsevier Inc. All rights reserved.

Keywords: Biclustering; Microarray data; Local search

1. Introduction

In the past recent years, DNA *microarray* technology has captured the attention of scientific community because of its capability of simultaneously measuring the activity and interactions of thousands of genes. The relative abundance of the mRNA of a gene under a specific experimental condition (or sample) is called the *expression level* of a gene. The expression level of a large number of genes of an organism under various experimental conditions can be arranged in a data matrix, also known as *gene expression data matrix*, where rows correspond to genes and columns to conditions. Thus each entry of this matrix is a real number representing the expression level of a gene under a specific experiment. One of the objectives of gene expression data analysis is to group genes according to their expression under multiple conditions.

Clustering [15,33,7] is an important gene expression analysis method that has been extensively used to group either genes, to search for functional similarities, or conditions, to find samples characterized by homogeneous gene expression levels. However, generally, genes are not relevant for all the experimental conditions, but groups of genes are often co-regulated and co-expressed only under specific conditions. This important observation has lead the attention towards the design of clustering methods that try to simultaneously group

* Corresponding author.
E-mail addresses: angiulli@icar.cnr.it (F. Angiulli), cesario@icar.cnr.it (E. Cesario), pizzuti@icar.cnr.it (C. Pizzuti).

0020-0255/$ - see front matter © 2007 Elsevier Inc. All rights reserved.
doi:10.1016/j.ins.2007.11.007

genes and samples. The approach, named *biclustering* or *co-clustering* [25,32,9], detects subsets of genes that show similar patterns under a specific subset of experimental conditions. This simultaneous clustering thus searches for sub-matrices whose rows show some kind of coherence with respect to the conditions appearing in the same sub-matrix.

The concept of bicluster is analogous to that of subspace clustering in data mining [3,2,1,26,13,14], though there exist important differences regarding the criteria adopted to measure the coherence among the rows and the type of biclusters found, that generally do not overlap, i.e. a row or a column can participate to only one bicluster.

Biclustering was first defined by Hartigan [21] and called *direct clustering*. His aim was to find a set of sub-matrices having zero variance, that is with constant values. This concept was then adopted by Cheng and Church [11] by introducing a similarity score, called *mean squared residue*, to measure the coherence of rows and columns in the bicluster. A group of genes is considered coherent if their expression levels varies simultaneously across a set of conditions. Biclusters with a high similarity score and, thus, with low residue, indicate that genes show similar tendency on the subset of the conditions present in the bicluster.

In this paper a greedy search algorithm, named *Random Walk Biclustering* (*RWB*), to find k biclusters with a fixed degree of overlapping is proposed. The method is motivated by that of [11] because it uses the concept of mean squared residue to evaluate the similarity of gene activity under a specific subset of experimental conditions. However, it introduces the notion of *gain* that combines the mean squared residue, the row variance, and the size of the bicluster to guide the algorithm towards those parts of the search space that contain local optimal solutions. The method, in fact, is enriched with an heuristic that avoids to get trapped at poor local minima. The algorithm starts with an initial random bicluster and searches for a locally optimal solution by successive transformations that improve a *gain* function. The *gain* guarantees that the transformation of a solution to a local neighborhood is done only if there is either a reduction of the residue, or an increase of the row variance, or an enlargement of the volume. In order to escape poor local minima, that is low quality biclusters having negative gain in their neighborhood, random moves with given probability are executed. These moves delete or add a row/column on the base of different strategies introduced in the method.

To obtain k biclusters the algorithm is executed k times by controlling the degree of overlapping among the biclusters.

In order to assess the validity of the approach proposed, an extensive experimental study on several real life data sets has been performed. A performance analysis showed that the algorithm is able to find significant and coherent biclusters. Furthermore, a quantitative and qualitative comparison of our approach with that of Cheng and Church has pointed out that the *RWB* algorithm is able to obtain biclusters with lower residue and higher volume than those found by their algorithm. Indeed the presented algorithm is able to find highly coherent genes and conditions which lead to low residue, while that of [11] tends to find smaller biclusters with less coherence, i.e. larger residue.

The paper is organized as follows. The next section defines the problem of biclustering and the notation used. Section 3 describes the algorithm proposed. Section 4 discusses time and space complexity of the method. In Section 5 an overview of the existing approaches to biclustering is given. Finally, Section 6 reports the experiments on some real life data sets and the comparison with the Cheng and Church algorithm.

2. Notation and problem definition

In this section the notation used in the paper is introduced and a formal definition of bicluster is provided [11]. Let $X = \{I_1, \ldots, I_N\}$ be the set of genes and $Y = \{J_1, \ldots, J_M\}$ be the set of conditions. The data can be viewed as an $N \times M$ matrix A of real numbers. Each entry a_{ij} in A represents the relative abundance (generally its logarithm) of the mRNA of a gene I_i under a specific condition J_j. A bicluster is a subset of rows that shows a coherent behavior across a subset of columns and vice versa.

Definition 2.1. A *bicluster* is a sub-matrix $B = (I, J)$ of A, where $I \subseteq X$ is a subset of the rows X of A, and J is a subset of the columns Y of A.

The problem of biclustering can then be formulated as follows: given a data matrix A, find a set k of biclusters $B_i = (I_i, J_i)$, $i = 1, \ldots, k$ that satisfy some homogeneity characteristics. The kind of homogeneity a

bicluster must fulfil depends on the approach adopted. In the next the notion of *mean squared residue*, introduced by Cheng and Church, is recalled.

Definition 2.2. Given a bicluster $B = (I,J)$, let $a_{iJ} = \frac{1}{|J|}\sum_{j\in J} a_{ij}$ denote the mean of the ith row of B, $a_{Ij} = \frac{1}{|I|}\sum_{i\in I} a_{ij}$ the mean of the jth column of B, and $a_{IJ} = \frac{1}{|I||J|}\sum_{i\in I, j\in J} a_{ij}$ the mean of all the elements in the bicluster, where $|I|$ and $|J|$ denote the number of rows and columns of B respectively.

Definition 2.3. The *residue* r_{ij} of an element a_{ij} of the matrix A is defined as $r_{ij} = a_{ij} - a_{iJ} - a_{Ij} + a_{IJ}$.

The residue of an element provides the difference between the actual value of a_{ij} and its expected value predicted from its row, column, and bicluster mean. The residue of an element reveals its degree of coherence with the other entries of the bicluster it belongs to. The lower the residue, the higher the coherence. The quality of a bicluster can thus be evaluated by computing *the mean squared residue* r_{IJ}, i.e. the sum of all the squared residues of its elements.

Definition 2.4. The *volume* v_{IJ} of a bicluster $B = (I,J)$ is the number of entries a_{ij} such that $i \in I$ and $j \in J$, that is $v_{IJ} = |I| \times |J|$.

Definition 2.5. The residue of a bicluster $B = (I,J)$ is $r_{IJ} = \frac{\sum_{i\in I, j\in J}(r_{ij})^2}{v_{IJ}}$, where r_{ij} is the residue of an element a_{ij} of B and v_{IJ} is its volume.

The *mean squared residue* of a bicluster, as outlined by Cheng and Church in [11], provides the similarity score of a bicluster. The lower the residue, the larger the coherence of the bicluster, and thus better its quality.

Definition 2.6. Given a threshold $\delta \geqslant 0$, a sub-matrix $B = (I,J)$ is said a *δ-bicluster*, if $r_{IJ} < \delta$.

The aim is then to find large biclusters with scores below a fixed threshold δ. However, low residue biclusters should be accompanied with a sufficient variation of the gene values with respect to the row mean value, otherwise trivial biclusters having almost all constant values could be determined. To this end a relatively high variance could be preferred to discard biclusters with almost constant values.

Definition 2.7. The row variance var_{IJ} of a bicluster $B = (I,J)$ is defined as

$$var_{IJ} = \frac{\sum_{i\in I, j\in J}(a_{ij} - a_{iJ})^2}{v_{IJ}}.$$

The problem of biclustering can then be re-formulated as: given a data matrix A, find a set k of biclusters $B_i = (I_i,J_i)$, $i = 1,\ldots,k$ having large volume, a relatively high variance, and a mean squared residue lower than a given threshold δ.

A quality measure of a bicluster based on volume, variance, and residue allows to detect maximal submatrices having rows, thus genes, coherent though different. In the next section the *Random Walk Biclustering* (*RWB*) algorithm is presented. The method adopts the concept of *gain*, that combines mean squared residue, row variance, and volume to improve the quality of a bicluster.

3. Algorithm description

In this section we present *RWB*, a biclustering algorithm based on a greedy technique enriched with a local search strategy to escape poor local minima (see Fig. 1). The basic schema of our method derives from the WSAT algorithm of Selman et al. for the *Satisfiability problem* [28], opportunely modified to deal with the biclustering problem.

Local search is a well known optimization technique devised to solve NP-hard combinatorial optimization problems. Given an initial point, a local minimum is found by searching for a local neighbor which improves the value of the object function. The main problem in applying local search methods to combinatorial problems is that the search space presents many local optima and, consequently, the algorithm can get trapped at local minima. Some heuristics have been implemented to overcome this problem. Some of them are based on

Algorithm RWB
Input:
- $matrix$: a gene-expression matrix
- δ: stop when this value of residue is reached (0=stop at local minimum)
- max_flips: maximum number of iterations allowed
- $method$: type of random move
- p: probability of a random move (0 = no random move)
- $w_{res}, w_{var}, w_{vol}$: weight associated to the residue, row variance, and volume resp.
- row_{min}, row_{max}: minimum and maximum number of rows allowed in the bicluster
- col_{min}, col_{max}: minimum and maximum number of columns allowed in the bicluster

Method:

```
generate at random a bicluster that does not violate the constraints on the number of rows and columns
set flips = 0, res = +∞, local_minimum = false
while flips < max_flips and δ < res and not local_minimum
    flips = flips + 1
    if a random generated number is less than p then
        execute a random move according to the method chosen, that does not violate the constraints on the number of rows and columns, and update the residue value res
    else
        let m be the move, that does not violate the constraints on the number of rows and columns, with the maximum gain gain
        if gain > 0 then
            execute the move m and update the residue value res
        else
            set local_minimum = true
return the bicluster computed
```

Fig. 1. The *Random Walk Biclustering* algorithm.

allowing random moves to a new neighboring point of the local search space even if the value of the evaluation function increases or to randomly moving to a point furthest in the search space.

A bicluster $B = (I, J)$ can be encoded as a binary string b of length $N + M$, where N and M are the number of rows (genes) and columns (conditions) of the expression matrix, respectively. The first N bits of b are related to the rows, and the remaining M bits are related to the columns. If the value of the ith bit is set to 1 it means that the corresponding ith gene, if $1 \leqslant i \leqslant N$, or column, if $N < i \leqslant N + M$, belongs to the bicluster.

The algorithm starts with an initial random bicluster $B = (I, J)$ and searches for a δ-bicluster by successive transformations of B, until a gain function is improved. The transformations consist in the change of membership (called *flip* or *move*) of the row/column that leads to the largest increase of the gain function. If a bit is set from 0 to 1 it means that the corresponding gene or condition, which was not included in the bicluster B, is added to B. Vice versa, if a bit is set from 1 to 0 it means that the corresponding gene or condition is removed from the bicluster.

The *gain* function combines mean squared residue, row variance, and size of the bicluster by means of user-provided weights w_{res}, w_{var}, and w_{vol}. More formally, let

$$\Delta res = \frac{res_{old} - res_{new}}{res_{old}}, \quad \Delta var = \frac{var_{old} - var_{new}}{var_{old}}, \quad \Delta vol = \frac{vol_{old} - vol_{new}}{vol_{old}}$$

be the relative changes of residue, row variance, and volume when a row/column is added/removed, where res_{old}, var_{old}, and vol_{old} (resp. res_{new}, var_{new}, and vol_{new}) are respectively the values of the residue, row variance and volume of B before (after) the move. Then the function *gain* is defined as

$$gain = w_{res}(2^{\Delta res} - 1) - w_{var}(2^{\Delta var} - 1) - w_{vol}(2^{\Delta vol} - 1),$$

with $w_{res} + w_{var} + w_{vol} = 1$. This function assumes values in the interval $[-1, +1]$. In fact, relative changes Δres, Δvar, and Δvol range in the interval $[-\infty, +1]$, consequently the terms $2^{\Delta res}$, $2^{\Delta var}$, and $2^{\Delta vol}$ range in the interval $[0, 2]$, and the whole function assumes values between -1 and $+1$. The weights w_{res}, w_{var}, and w_{vol} provide a trade-off among the relative changes of residue, row variance, and volume. When $w_{res} = 1$ (and thus $w_{var} = w_{vol} = 0$), the algorithm searches for a *minimum residue bicluster*, since the gain monotonically increases with the residue of the bicluster. Decreasing w_{res} and increasing w_{var} and w_{vol}, biclusters with higher row variance and larger volume can be obtained.

Notice that when the residue after a flip diminishes, and the row variance and volume increase, Δres is positive, while Δvar and Δvol are negative. Thus, when the gain function is positive, *RWB* is biased towards large biclusters with a relatively high variance, and low residue. A negative gain, on the contrary, means a deterioration of the bicluster because there could have been either an augmentation of the residue or a decrease of the row variance or volume.

During its execution, in order to avoid get trapped into poor local minima (i.e. low quality biclusters with negative gain in their neighborhood), instead of performing the flip maximizing the gain, with a user-provided probability p the algorithm is allowed to execute a random move. We introduced three types of random moves:

- NOISE: with probability p, choose at random a row/column of the matrix and add/remove it to/from B;
- REMOVE: with probability p, choose at random a row/column of B and remove it from B;
- REMOVE-MAX: with probability p, select the row/column of B scoring the maximum value of residue, and remove it from B.

Thus, the NOISE is a purely random strategy that picks a row/column from the overall matrix, and not only from the bicluster, and adds or removes the row/column to the bicluster if it belongs or it does not belong to it. The REMOVE strategy removes at random a row/column already present in the bicluster, thus it could accidentally delete a worthless gene/condition from the current solution, and the REMOVE-MAX removes that row/column already present in the bicluster having the highest value of the residue, i.e. mostly contributing to worsen the gain.

Fig. 3 shows the algorithm *RWB*. The algorithm receives in input a gene expression matrix, a threshold value (δ) for the residue of the bicluster, the maximum number of times (*max_flips*) that a flip can be done, the kind of random move the algorithm can choose (*method*), the probability (p) of executing a random move, the weight to assign to residue (w_{res}), variance (w_{var}), and volume (w_{vol}), and some optional constraints (row_{min}, row_{max}, col_{min}, col_{max}) on the size of the bicluster to find.

The flips are repeated until either a preset of maximum number of flips (*max_flips*) is reached, or a δ-bicluster is found, or the solution cannot ulteriorly be improved (get trapped into a local minimum). Until the stop condition is not reached, it executes a random move with probability p, and a greedy move with probability $(1 - p)$.

In the experimental section the three strategies will be compared on two data sets and the advantages of each discussed. When a greedy move is executed, the current value *res* of the mean squared residue is updated. It is used to check if a bicluster having residue below δ has been found. On the contrary, if the value of res_{min} is zero, then the algorithm continues the execution until the gain can be improved.

In order to compute k biclusters, we execute k times the algorithm *RWB* by fixing two frequency thresholds, f_{row} and f_{col}, that allow to control the degree of overlapping among the biclusters. The former binds a generic row to participate to at most $k \cdot f_{row}$ biclusters among the k to be found. Analogously, f_{col} limits the presence of a column in at most $k \cdot f_{col}$ biclusters. During the k executions of the algorithm, whenever a row/column exceeds the corresponding frequency threshold, it is removed from the matrix and not taken into account any more in the subsequent executions.

4. Computational complexity

In this section the time and space complexity of the method is discussed.

As far as the temporal cost of the algorithm is concerned, it is upper bounded by

$$max\_flips \times C_u \times [(1 - p) \times (N + M) + p]$$

where C_u is the cost of computing the new residue and the new row variance of the bicluster after performing a move. In order to reduce the complexity of C_u, we maintain, together with the current bicluster $B = (I, J)$, the mean values a_{iJ} and a_{Ij}, for each $i \in I$, the summation $\sum_{j \in J} a_{ij}^2$, and the total sum of the row variances. The computation of the new residue of each element involves recomputing the $|I| + |J|$ mean values a_{iJ} $(1 \leqslant i \leqslant |I|)$ and a_{Ij} $(1 \leqslant j \leqslant |J|)$ after performing the move. This can be done efficiently, in time $\max\{|I|, |J|\}$, by exploiting the values maintained together with the current bicluster.

Computing the residue res_{new} of the new bicluster, requires the computation of the squares of its element residues, a time proportional to the volume of the new bicluster. We note that, usually, $|I||J| \ll NM$, and in general it is dominated by the number N of genes of the overall matrix.

Computing the new row variances can be done in a fast way by exploiting the summations $\sum_{j \in J} a_{ij}^2$ already stored. Indeed, if a column is added or removed, the new row variances can be obtained quickly by evaluating the $|I|$ expressions $\frac{1}{|J|}\sum_{ij}(a_{ij}^2) - a_{iJ}^2$ $(1 \leqslant i \leqslant |I|)$. For example, if the qth column is added, in order to compute the new variance of the row i, the following expression must be evaluated:

$$\frac{1}{|J|+1}\left(\sum_{j \in J}(a_{ij}^2) + a_{iq}^2\right) - \left(\frac{|J|a_{iJ} + a_{iq}}{|J|+1}\right)^2.$$

Analogously if a column is removed. Otherwise, if a row is added (removed resp.) the corresponding row variance must be computed and added (subtracted resp.) to the overall sum of row variances.

Before concluding, we note that the cost of a random move is negligible, as it consists in generating a random number, when the NOISE or REMOVE strategies are selected, while the row/column with the maximum residue, selected by the REMOVE-MAX strategy, is computed, with no additional time requirements, during the update of the residue of the bicluster at the end of each iteration, and, hence, it is always immediately available.

5. Related work

Biclustering approaches [21,24,17,11,8,34,31,35,23,12,10,30,29,27], also called co-clustering, are receiving a lot of attention in the research community and are tightly related to subspace clustering in data mining and text mining [3,2,1,26,13,14]. In the following we review the main proposals for biological data analysis. Surveys on biclustering algorithms for gene expression data can be found in [25,32,9]. In particular, in [25] biclustering approaches are classified with respect to the type of bicluster found and the search method adopted.

Hartigan [21] first suggested a partition-based algorithm, called *direct clustering*, that splits the data matrix to find sub-matrices having zero variance, that is with constant values. The variance of a bicluster is used to evaluate its quality and the algorithm stops when K sub-matrices are found. Though the aim of the approach was to obtain constant biclusters, Hartigan proposed to modify his algorithm to find biclusters with constant rows, constant columns or coherent values in rows and columns.

Cheng and Church [11] were the first who introduced the new paradigm of biclustering to gene expression data analysis. They proposed some greedy search heuristics that generate suboptimal biclusters satisfying the condition of having the mean squared residue below a threshold δ. Cheng and Church proved that the problem of finding biclusters with low mean squared residue, in particular maximal biclusters with scores under a fixed threshold, is NP-hard because it includes the problem of finding a maximum biclique in a bipartite graph as a special case [16]. Therefore, they proposed some heuristics that are able to generate good quality biclusters. The algorithm, in the following referred as *CC*, starts with the original data matrix and removes highest scoring rows and columns as long as the mean squared residue is above the threshold δ. After that, rows and columns are added provided that they do not increase the residue above the threshold. The approach is deterministic, thus to discover more than one cluster, in order to avoid to reobtain the same biclusters, the algorithm is repeated on a modified data matrix where the values of those elements a_{ij} already inserted in a bicluster are substituted with random numbers. This choice, however, has the negative effect of precluding the discovery of new biclusters, in particular those having significant overlaps with those already found. The *CC* algorithm assumes that the data matrix does not contain missing values.

Yang et al. [35] extended the definition of δ-bicluster to cope with missing values and to avoid the problems caused by random numbers. They defined a probabilistic *move-based* algorithm flexible overlapped bicluster-

ing (FLOC) that generalizes the concept of mean squared residue and it is based on the concept of *action* and *gain*. Given a row (or a column) x and a bicluster c, the *action* $Action(x, c)$ is defined as the change of membership of x to c. Thus if x already belongs to the bicluster c, $Action(x, c)$ denotes the removal of x from c, otherwise it denotes the addition. The algorithm consists of two phases. In the first phase, k biclusters are generated. In the second phase the quality of the biclusters is improved by performing the *best action* for each bicluster, as the one that produces the highest *gain* for all the k biclusters. The *gain* of an action $Action(x, c)$ is defined as a function of the relative reduction of the c's residue and the relative enlargement of the c's volume after performing it. After the best action is identified for each row and column, the $N + M$ actions are performed sequentially. However, in order to avoid local optimal solutions, a random weighted ordering is introduced. This ordering assigns higher probability to actions with positive gains with respect to those with negative gain.

Though our concept of *flip* in *RWB* is similar to that of *Action* in *FLOC*, the two algorithms are substantially different. In fact, *RWB* finds one cluster at a time and the change of membership of a row/column is done locally with respect to the current bicluster, *FLOC* obtains k biclusters at the same time and executes the action that globally improves the gain. Furthermore, the approaches to escape local minima are completely different. *RWB* performs random moves, while *FLOC* relies on random action ordering.

Cho et al. [12] propose two iterative co-clustering algorithms that use two different squared residue measures, one is that adopted by Hartigan [21], the second is that defined by Cheng and Church [11]. The two algorithms are based on the k-means clustering method. The authors formulate the problem of minimizing the residue as a trace optimization problem and use the spectral relaxation of such problem to initialize their methods.

Getz et al. [17] presented the *Coupled Two-Way Clustering* (*CTWC*) algorithm that uses a standard hierarchical clustering method separately on each dimension to discover significant clusters, called stable. The approach selects a subset of rows and columns and applies the one-dimensional algorithm iteratively on them until the columns detected are only those used to cluster the corresponding rows, or vice versa. This allows to detect stable sub-matrices, i.e. biclusters, that partition the data matrix. The approach avoids the regeneration of the same row and column subsets by dynamically maintaining a list of stable clusters and a list of pairs of row and column subsets.

Lazzeroni and Owen [24] introduced the Plaid model. The basic idea of this approach is that each element in the data matrix is viewed as a sum of terms called layers, where a layer corresponds to a bicluster. Thus the data matrix is described as a linear function of layers and the objective is to minimize a merit function that take into account the interactions between the biclusters.

Tanay et al. [31,30,29] presented statistical-algorithmic method for bicluster analysis (SAMBA), a biclustering algorithm that combines graph theory and statistics. The data matrix is represented as a bipartite graph where the nodes are conditions and genes, and edges denote significant expression changes. Vertex pairs are associated with a weight according to a probabilistic model, and heavy subgraphs correspond to significant biclusters. The problem of finding biclusters reduces thus to that of finding the heaviest subgraphs in the bipartite graph. Since this problem is NP-hard, to reduce the execution time, SAMBA applies some restrictions on the size of the biclusters and employs a heuristic to search for heavy subgraphs.

6. Experimental results

In order to evaluate the method proposed, we performed an experimental study on several benchmarks. We employed two well known gene expression data sets, the *Yeast Saccharomyces cerevisiae* cell cycle and the human B-cell *Lymphoma*, to study the behavior of *RWB* when the different random move strategies are adopted. Other seven data sets, *Colon Rectal Tumor*, *Lung Cancer*, *Prostate Cancer*, *Ovarian Cancer*, *ALL-AML Leukemia*, *Colon Tumor*, and the *Central Nervous System*, were used to compare our approach with that of Cheng and Church from a point of view of both quantitative and qualitative parameters. In fact we compared the biclusters found by *RWB* and *CC* with respect to residue, variance, and volume, and then we evaluated the ability of both the methods to produce biclusters conforming to prior biological knowledge.

The *RWB* algorithm has been implemented by using the C programming language. As regards the *CC* algorithm, we employed the Java implementation publicly available on the web site of the authors.[1] All the experiments have been performed on a Pentium Mobile 1700 MHz based machine.

6.1. Evaluation of random move strategies

This first set of experiments aims at comparing the three random move strategies introduced when different probabilities and input parameters are given and to discuss the advantages of each of them. For this experiment we used the *Yeast Saccharomyces cerevisiae* cell cycle expression data set, and the human B-cell *Lymphoma* data set. The preprocessed gene expression matrices can be obtained from [11] at http://arep.med.harvard.edu/biclustering. The yeast cell cycle data set contains 2884 genes and 17 conditions. The human lymphoma data set has 4026 genes and 96 conditions.

We computed $k = 100$ biclusters varying the probability p of a random move in the interval $[0.1, 0.6]$, for two different configurations of the weights, i.e. $\mathbf{w_1} = (w_{res}, w_{var}, w_{vol}) = (1, 0, 0)$ (dashed lines in Fig. 2) and $\mathbf{w_2} = (w_{res}, w_{var}, w_{vol}) = (0.5, 0.3, 0.2)$ (solid lines in Fig. 2). Notice that $w_{res} = 1$ and $w_{var} = w_{vol} = 0$ means that the gain function is completely determined by the residue value. We set *max_flips* to 100, δ to 0, and the frequency thresholds to $f_{row} = 10\%$ and $f_{col} = 100\%$, i.e. a row can participate in at most 10 biclusters, while a column can appear in all the 100 biclusters. The initial random generated biclusters are of size 14×14 for the Yeast data set and of size 20×20 for the Lymphoma data set, while we constrained biclusters to have at least $row_{min} = 10$ rows and $col_{min} = 10$ columns.

Fig. 2 shows the behavior of the algorithm on the two above mentioned data sets. From the top to the bottom, the figures show the average residue, row variance and volume of the 100 biclusters computed, the average number of flips performed by the method, and the average execution time. Figures on the left concern the Yeast data set, and figures on the right the Lymphoma data set.

We can observe that, as regards the residue, the REMOVE-MAX method performs better than the two others, as expected. In fact, its random move consists in removing the gene/condition having the highest residue. Furthermore, increasing the random move probability p improves the value of the residue. The residue of the NOISE method, instead, deteriorates when the probability increases. The REMOVE strategy, on the Yeast data set is better than the NOISE one, but worse than the REMOVE-MAX. On the Lymphoma data set, the value of the residue increases until $p = 0.3$ but then it decreases. The residue scored for parameters $\mathbf{w_1}$ (dashed lines) is lower with respect to that obtained for $\mathbf{w_2}$ (solid lines), for the two strategies NOISE and REMOVE, while, for REMOVE-MAX the difference is negligible.

As regards the variance, we can note that the variance of REMOVE is greater than that of NOISE, and that of NOISE is greater than that of REMOVE-MAX for both $\mathbf{w_1}$ and $\mathbf{w_2}$. This is of course expected, since in the former case we do not consider the variance in the gain function in order to obtain the biclusters, while in the latter the weight of the variance is almost as important as that of the residue (0.3 w.r.t. 0.5).

Analogous considerations hold for the volume, whose value is higher for $\mathbf{w_2}$. Furthermore, the volume is almost constant for the NOISE strategy, because the probability of adding or removing an element in the bicluster is more or less the same, but it decreases for the REMOVE and REMOVE-MAX strategies. These two strategies tend to discovery biclusters having the same size when the probability p increases.

As for the average number of flips, we can note that 100 flips are never sufficient for the NOISE method to reach a local minimum, while the other two methods do not execute all the 100 flips. In particular, the RANDOM-MAX strategy is the fastest since it is that which needs less flips before stopping.

As regards the execution time, the algorithm is faster for $\mathbf{w_1}$ w.r.t. $\mathbf{w_2}$, but, in general, the execution time decreases when the probability p increases and they are almost the same for higher values of p because the number of random moves augments for both.

Finally some consideration on the quality of the biclusters obtained. We noticed that the NOISE strategy, which works in a purely random way, gives biclusters with higher residue and it requires more execution time. On the contrary, REMOVE-MAX is positively biased by the removal of those elements in the bicluster having

[1] Downloadable on http://cheng.ececs.uc.edu/biclustering/Biclustering.java.

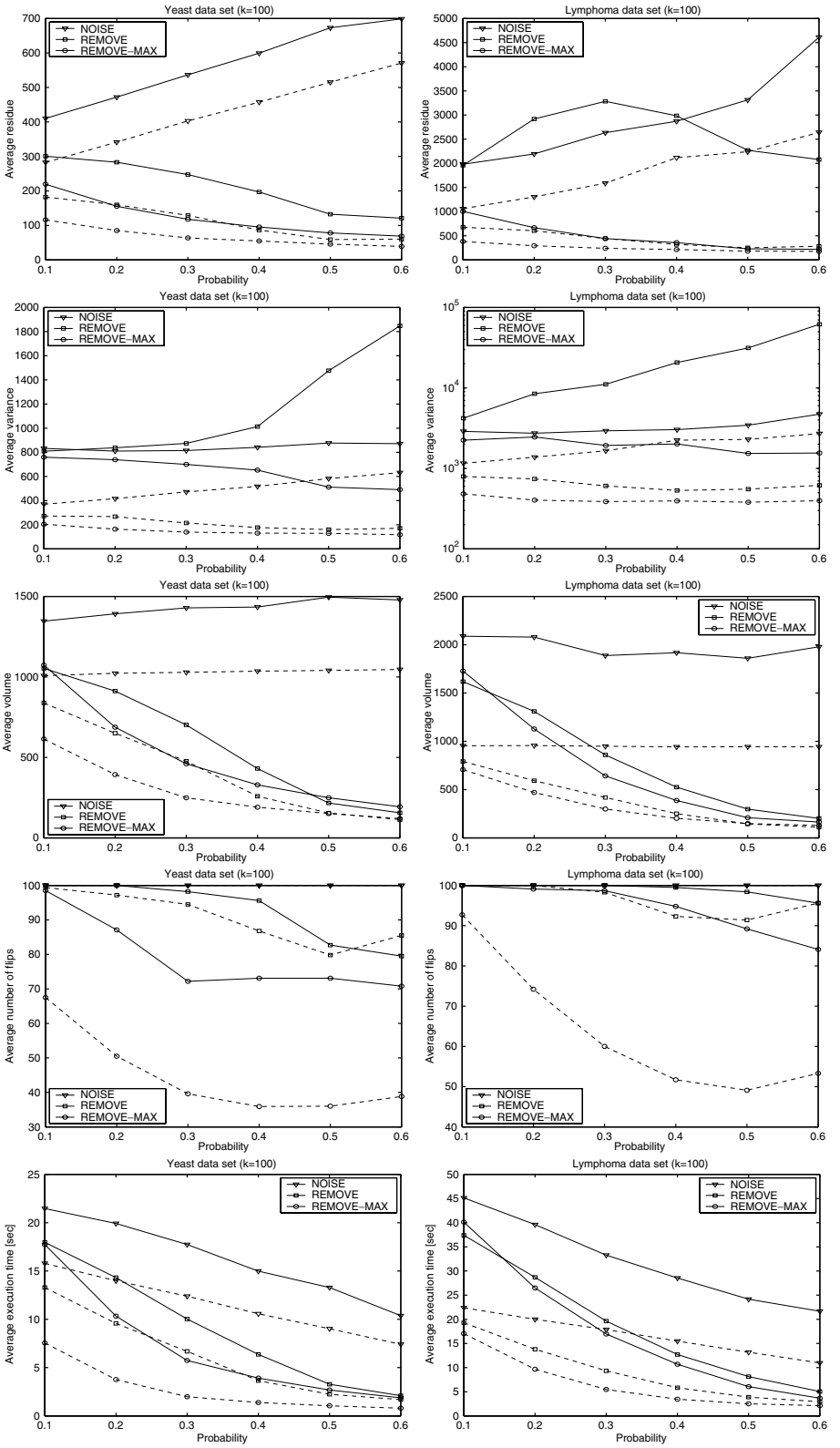

Fig. 2. Average residue, variance, volume, number of flips and execution time for Yeast (on the left) and Lymphoma (on the right) data sets.

the worst residue, thus it is able to obtain biclusters with lower values of residue, while the REMOVE strategy extracts biclusters with higher variance.

To show the quality of the biclusters found by *RWB*, Fig. 3 depicts some of the biclusters discovered in the experiments of Fig. 2 by using the REMOVE-MAX strategy for $(w_{res}, w_{var}, w_{vol}) = (0.5, 0.3, 0.2)$. The x axis corresponds to conditions, while the y axis gives the gene expression level. The figures point out the good quality of the biclusters obtained. In fact, their expression levels vary homogeneously under a subset of conditions, thus they present a high degree of coherence. Fig. 4a shows the heatmap of the overall Lymphoma data set. To create the map we used the Heatmap Builder software [22]. Fig. 4b displays the heatmap of a subset of the genes and samples, obtained by choosing ten biclusters among the 100 returned by the experiment. The figure has been generated by the BIVOC software [18], that searches for an optimal reordering of rows and columns before creating the picture. It is worth to note that, since *RWB* tries to minimize the residue and maximize row variance, the heatmap of a bicluster found cannot show a uniform color, instead the bicluster should present coherent values on both rows and columns, i.e., as reported in [25], each row or column can be obtained by adding a constant to each of the others or by multiplying each of the others by a constant value. In fact the biclusters showed in Fig. 4b have columns of the same color, confirming the presence of coherent values.

6.2. Comparative analysis of RWB and CC

There exist several studies to compare and validate clustering approaches [36,5,6]. Generally they make use of *validity indices* [19,20] to assess their quality. Validity indices are classified in internal, external, and relative. Internal indices rely on the input data and are mainly based on the concepts of homogeneity and separation between the groups, external indices use additional data to validate the clustering outcomes, and relative indices measure the influence of the input parameters on the results. In the context of gene expression data there is no general agreement on which validity indices to adopt because of the different types of biclusterings the algorithms detect: constant values, coherent values, coherent evolutions, checkerboard structure, overlapping, etc. [25]. However, internal indices meaningful for comparing our approach and that of Cheng and Church clearly are the residue, volume, and variance values. Furthermore, external criteria correspond to prior biological knowledge on the data set being studied.

In the following we compare *RWB* and *CC* first with respect to these internal indices, and then we evaluate them with respect to some external indices by studying the ability of both the algorithms to obtain biologically significant biclusters. To this end we used six real life data sets coming from the *Kent Ridge Bio-medical Data Set Repository*,[2] and one (*Colon Rectal Cancer*) used in [4].[3] A brief description of each data set follows.

The *Colon Rectal Cancer* contains the expression of the 2000 genes with highest minimal intensity across 62 parts of the colon. Each entry is a gene intensity derived from 20 feature pairs that correspond to the gene on the chip of the tissue.

The *Lung Cancer* data set collects 181 tissue samples, described by 12,533 genes, representing classified data between malignant pleural mesothelioma and adenocarcinoma of the lung.

The *Prostate Cancer* data set contains the RNA values of 12,600 genes collected on 136 human samples classified as normal and having prostate tumor. The prostate tumor samples came from patients undergoing radical prostatectomy between 1995 and 1997.

The *Ovarian Cancer Tumor* data set contains values of 253 women (91 no-ovarian cancer and 162 ovarian cancer), described by 15,154 proteomic features. This dataset is of a great interest for women who have a high risk of ovarian cancer due to family or personal history of cancer.

The *Leukemia* data set contains the RNA value of 7129 probes of human genes collected on 72 acute leukemia patients. From a biological point of view, such subjects are classified as Acute Lymphoblastic Leukemia (ALL) and Acute Myeloid Leukemia (AML). In this data set there are 47 ALL and 25 AML.

The *Colon Tumor* data set contains the RNA values of 2000 (selected from around 6500) human genes collected from 62 cancer-patients. Among them, 40 tumor biopsies are from tumor-unhealthy parts (labelled as

[2] http://sdmc.lit.org.sg/GEDatasets/Datasets.html.
[3] http://microarray.princeton.edu/oncology/affydata/index.html.

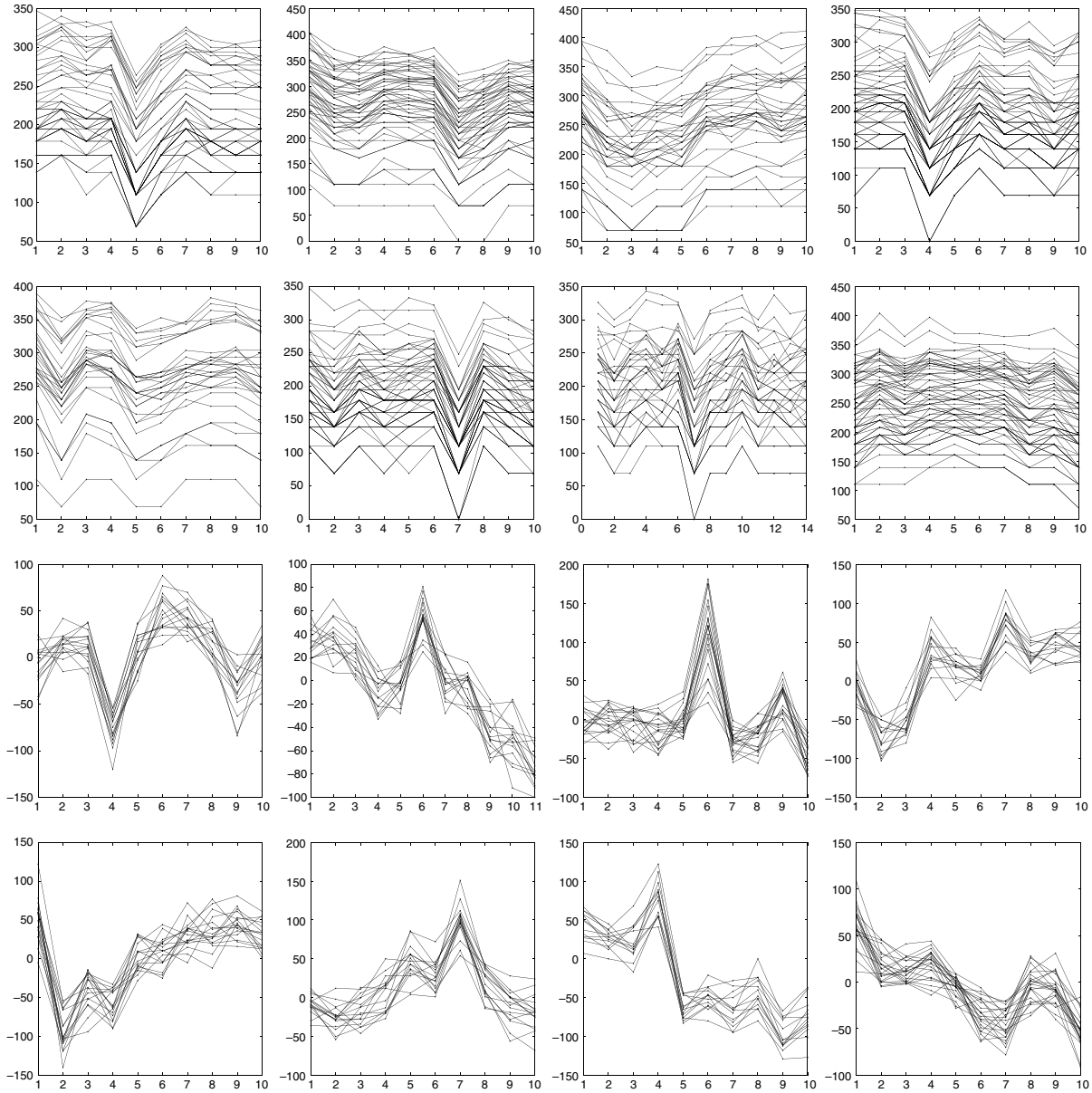

Fig. 3. Biclusters obtained by using the REMOVE-MAX strategy with $(w_{res}, w_{var}, w_{vol}) = (0.5, 0.3, 0.2)$ in the experiments of Fig. 2. The first two rows show eight biclusters of the Yeast data set ($p = 0.3$), while the subsequent two rows show eight biclusters of the Lymphoma data set ($p = 0.5$). From left to right and from top to bottom the values of (residue, variance, volume) are the following: (70.14, 590.65, 460), (99.51, 705.58, 530), (160.89, 834.79, 360), (113.47, 674.25, 630), (83.04, 439.81, 310), (136.31, 788.27, 580), (180.03, 545.23, 518), and (111.01, 356.24, 640) for the Yeast data set, and (214.63, 1414.45, 150), (169.96, 1626.74, 165), (366.18, 2012.5, 190), (181.34, 2135.5, 140), (323.17, 2472.65, 170), (182.45, 1499.95, 160), (200.24, 3412.19, 130), and (172.94, 1197.03, 220) for the Lymphoma data set.

"Negative") and 22 normal biopsies are from healthy parts (labelled as "Positive") of the colons of the same patients.

Embryonal tumors of the *Central Nervous System* represent an heterogeneous group of tumors about which little is known biologically, and whose diagnosis, based on morphologic appearance alone, is controversial. The Central Nervous System data set contains the RNA value of 7129 genes of 60 diseased patients, after a medical treatment. Patients are labelled as "Survivors" (21) or "Failures" (39), depending on the outcome of the treatment.

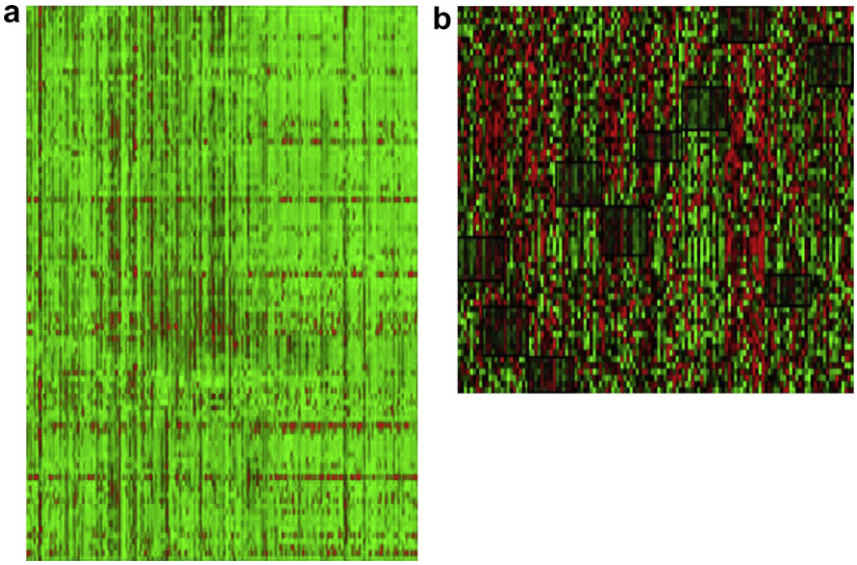

Fig. 4. Heatmap of the original Lymphoma data set (on the left) and biclusters found on a subset of the genes and samples.

6.2.1. Quantitative analysis

This set of experiments aims at evaluating and comparing the behavior of the *RWB* algorithm and that of Cheng and Church (*CC*, for short, in the following) when different values of some input parameters are used.

The experiments analyze the biclusters obtained by the two algorithms when the number of biclusters k, and the maximum residue value δ are varied. In particular, we asked the two algorithms to find a number of biclusters from 5 to 50, that is for $k = 5, 10, 20, 30, 50$, when the residue threshold δ is fixed to 0, 500, 800 and 1000. For all of experiments the *RWB* algorithm used the weight configuration $w = (0.6, 0.1, 0.3)$, a maximum number of flips *max_flips* of 3000, the frequency threshold $f_{row} = 100\%$ and $f_{col} = 100\%$, i.e. each row and column can potentially participate in all the k biclusters. Even though such a constrain could lead to heavy overlapping between biclusters, we verified that the rate of overlapping is marginal, below the 2%. Finally, we fixed the probability of random move to $p = 0.5$ and, for all the experiments, we adopted the *REMOVE-MAX* strategy. Both *RWB* and *CC* algorithms have been executed 10 times and the values obtained by averaging these 10 executions are reported.

Figs. 5–7 show, for the *Leukemia*, *Colon Tumor*, and *Central Nervous System* data sets respectively, the average residue, average row variance, and average volume of the biclusters extracted by the algorithms when the residue threshold assumes the values $\delta = 0, 500, 800, 1000$. From the Fig. 5 we can observe that the *RWB* algorithm, on the *Leukemia* data set, always obtains biclusters with lower residue (Fig. 5a,d,g and j), higher volume (Fig. 5c,f,i and l), and lower variance (Fig. 5b,e,h, and k) than the *CC* algorithm. This means that the former algorithm is able to find highly coherent genes and conditions which lead to low residue while the latter tends to find smaller biclusters with less coherence, i.e. larger residue. Lower variance results for *RWB* are due to the fixed weight value of 0.1. Another important observation to point out is that *RWB* maintains a residue value very near to the threshold requested as long as the number of biclusters to find increases. Analogously, the volume of biclusters do not sensibly varies along the executions. On the contrary, *CC* is not able to satisfy the threshold requirement because, though when we ask for few biclusters the average residue does not significantly move away from the fixed value, when the number of biclusters augments, the residue sensibly increases while the volume decreases. This behavior probably is due to the introduction of random numbers adopted by *CC* to bias the algorithm to find non overlapping biclusters. As regards *RWB*, the random strategy employed allows to explore different parts of the search space thus assuring low levels of overlapping, although the frequency thresholds of 100% enable each row and column to take part to all the biclusters.

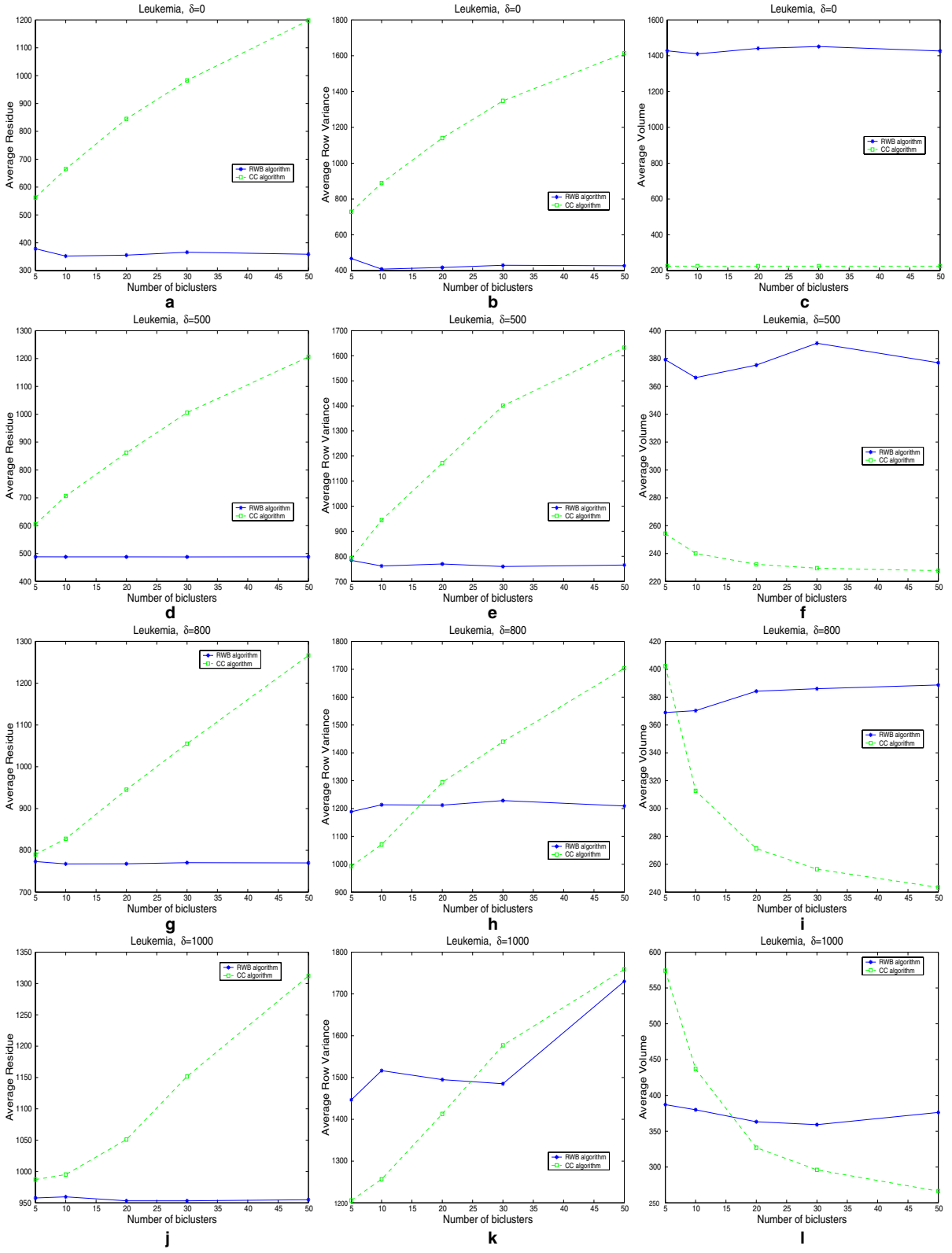

Fig. 5. Average residue, row variance, and volume, for the Leukemia data set when $\delta = 0, 500, 800, 1000$.

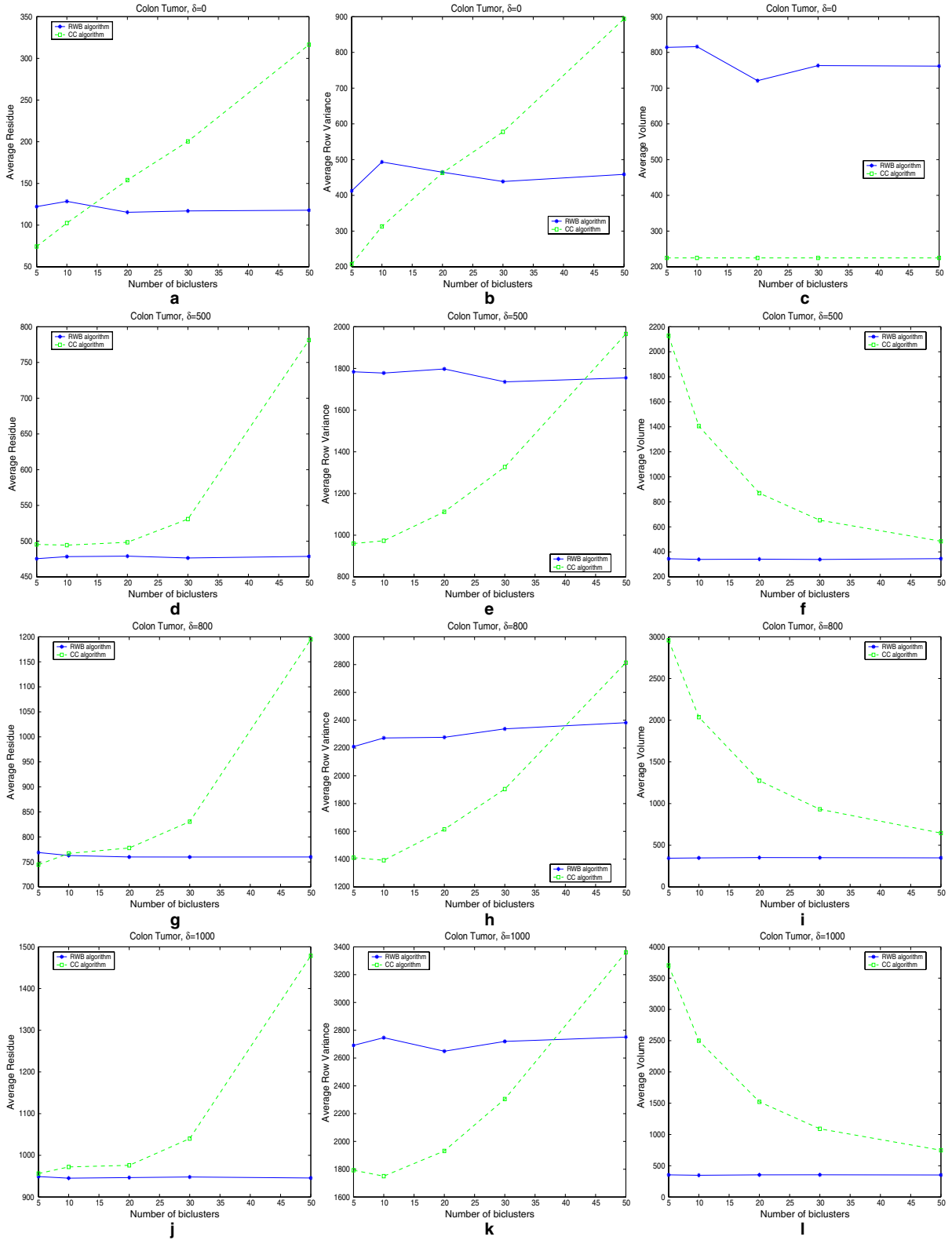

Fig. 6. Average residue, row variance, and volume, for the Colon Tumor data set when $\delta = 0, 500, 800, 1000$.

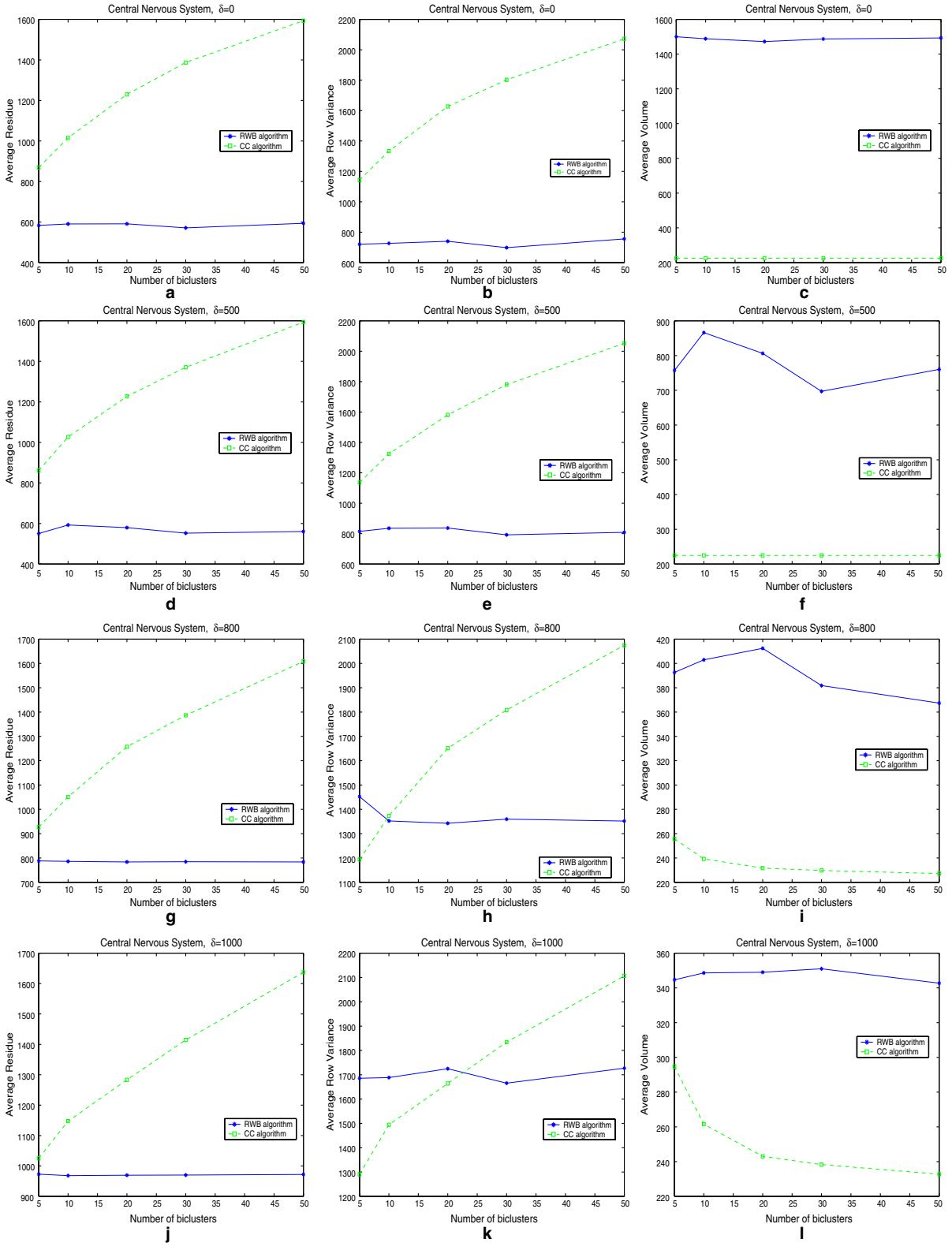

Fig. 7. Average residue, row variance, and volume, for the Central Nervous System data sets when $\delta = 0, 500, 800, 1000$.

Figs. 6 and 7 show a similar behavior of the two algorithms on the *Central Nervous System*, and *Colon Tumor* data sets. In particular, Fig. 7a,d,g and j confirm that the biclusters found by *RWB* have lower residue, lower variance (Fig. 7b,e,h and k) than those found by the *CC* algorithm, and higher volume (Fig. 7c,f,i and l). As regards the *Colon Tumor* data set, Fig. 6a,d,g and j show that the biclusters obtained by *RWB* have almost always lower residue than *CC*, though, for a threshold greater than zero, the volume of the biclusters found by *CC* is higher. It is worth to note that when *RWB* is executed with a threshold value different than zero it means that the algorithm is forced to an early stopping. This could preclude *RWB* to continue to explore the search space and reach a better local optimal solution.

Finally, in Table 1 we summarize the results obtained by running *RWB* and *CC*, besides the three already discussed data sets, on other four data sets well known in the literature. For this experiment the residue threshold is fixed to 0, and the number of biclusters varies from 5 to 50. The table confirms the good behavior of *RWB* that obtains biclusters of lower residue and higher or comparable volume with respect to *CC*. Notably is the result for *Prostate Cancer* where the residue is between 1 and 2, while the volume is more than sixteen thousand elements.

Table 1
Comparison between RandomWalk and Cheng–Church algorithms on all the datasets

		RandomWalk			Cheng–Church		
	N. biclusters	Residue	Variance	Volume	Residue	Variance	Volume
C. Rectal cancer	5	120.03	520.79	786	257.80	486.91	891
	10	152.79	399.50	855	385.81	780.79	912
	20	323.50	3411.19	707	641.48	1302.16	897
	30	229.43	918.03	502	1008.79	2023.27	945
	50	137.04	1186.59	319	3083.84	5472.27	966
Lung cancer	5	5.11	6.07	1411	13.85	18.25	225
	10	5.18	6.34	1436	19.97	26.20	225
	20	5.09	6.06	1418	29.10	38.13	225
	30	5.13	6.20	1421	35.94	49.07	225
	50	5.07	6.18	1392	47.13	63.27	225
Prostate cancer	5	1.64	1.69	16873	1009.59	3354.20	7204
	10	1.70	1.75	16926	1659.35	5032.73	7288
	20	1.82	1.87	16278	2955.64	7820.32	7132
	30	1.78	1.99	16125	4548.19	10831.76	7031
	50	2.34	2.00	16258	8596.91	16421.69	7287
Ovarian cancer	5	4.89	3.05	11451	5.09	6.28	12996
	10	5.38	4.02	10728	4.76	6.12	12192
	20	5.55	9.88	12054	4.99	6.07	11894
	30	6.48	7.11	11008	7.13	5.10	12077
	50	4.99	8.73	11451	5.01	6.34	13500
Leukemia	5	378.76	468.07	1428	556.99	688.99	225
	10	352.17	408.56	1410	671.53	853.97	225
	20	355.66	417.72	1441	858.69	1156.26	225
	30	366.14	429.86	1451	1006.11	1361.89	225
	50	358.97	427.28	1427	1201.15	1602.16	225
Colon tumor	5	122.19	412.51	814	74.29	207.91	225
	10	128.55	493.34	816	105.47	338.06	225
	20	115.40	464.76	721	166.14	464.87	225
	30	117.03	438.75	763	196.53	573.37	225
	50	117.86	458.71	762	313.78	923.59	225
C. nervous system	5	583.44	720.66	1500	798.61	1006.19	225
	10	590.64	726.76	1488	1081.11	1394.86	225
	20	591.10	740.22	1472	1194.65	1568.29	225
	30	571.34	698.27	1487	1357.35	1781.19	225
	50	593.84	755.79	1493	1609.93	2122.25	225

6.2.2. Biological evaluation

In this section we want to qualitatively evaluate the capability of our algorithm with respect to that of Cheng and Church to extract biclusters meaningful from a biological point of view. Assessing the biological relevance of the results, however, is not a trivial task because there do not exist general guidelines in the literature. An approach that can be pursued when the gene are annotated with their functional category consists in verifying whether the biclusters discovered represents biological functional modules, that is whether each bicluster contains a large portion of similar genes, i.e. belonging to the same functional class. Annotated data sets, however, are not easily available. In general data sets with experimental conditions classified as either belonging to a positive or a negative class, where the meaning of positive and negative depends on the data set, are more common.

The data sets used in the previous section are such that each experimental condition corresponds to a patient presenting a kind of pathology. For example, the Leukemia data set discriminates patients affected by either Lymphoblastic or Myeloid leukemia, the Colon Tumor data set if a patient is positive or negative to colon tumor, and the C. nervous system data set if a patient survives or not after a clinical treatment. Thus we do not know the biological correlation between rows (genes), while we know the medical classification of columns. In this case, we can evaluate the ability of an algorithm to separate the samples according to their known classification. To this end, we can compute the number of columns labelled with the same class and belonging to the same bicluster. Obviously, higher the number of columns in a bicluster labelled with the same class label, higher its biological quality. In fact, this means that many patients with the same diagnosis are grouped together with respect to a subset of genes, thus we could induce that those genes probably have similar functional category and characterize the majority class of patients.

In our experiments we used the biological information on the experimental conditions to assign a qualitative significance to the biclustering results by using a well known validity index, that is the *Purity* [17].

Suppose a set of experimental conditions be classified in two classes C_1, C_2, and let $B_1, \ldots, B_k$ be the k biclusters found by an algorithm. A $k \times 2$ *confusion matrix* M can be built where a row corresponds to a bicluster and a column to a class label. The term m_{ij} of the confusion matrix represents the number of conditions of the expression matrix A belonging to the bicluster B_i and labelled as class C_j. Each bicluster B_i is considered as representing the biological class C_j of the most frequent label associated with its columns, that is the class label j such that m_{ij} is maximal in the bicluster B_i.

Consider, for example, Table 2 where the confusion matrices obtained by running the RWB and CC algorithms on the Leukemia data sets, for $k = 5$ and $\delta = 0$, are reported.

The table shows that the bicluster 1 obtained by *RWB* contains 14 columns labelled ALL and only one column labelled AML. From a biological point of view this means that this bicluster is able to identify a biologically relevant partition of genes and conditions since it is characterized by a high coherence on the columns. Thus the genes participating to this bicluster better characterize ALL type Leukemia rather than AML type. These genes deserve deeper investigation because could be determinant for this kind of pathology. On the

Table 2
Confusion matrices obtained by running *RWB* and *CC* on the Leukemia data set for $k = 5$

Bicluster No.	No. of ALL	No. of AML
(a) RWB		
1	14	1
2	12	3
3	12	3
4	13	2
5	14	3
(b) CC		
1	13	2
2	11	4
3	13	2
4	13	2
5	9	6

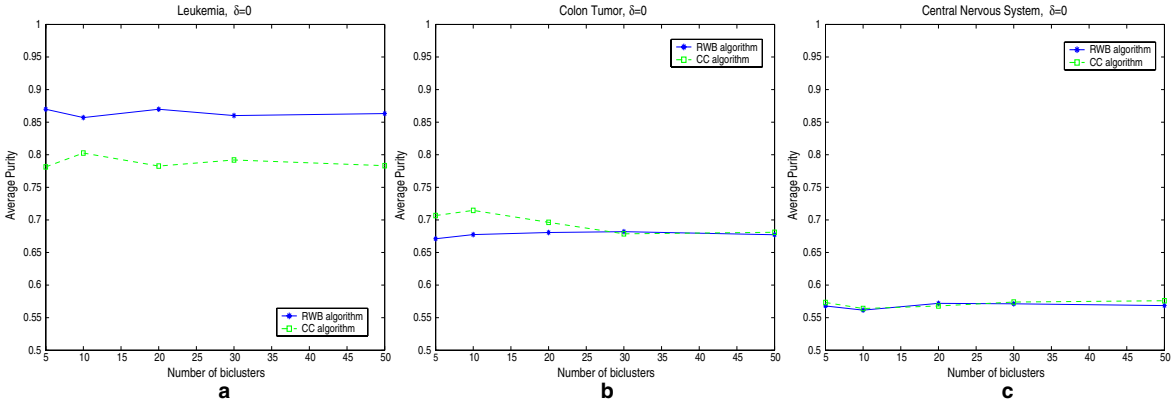

Fig. 8. Average purity for the Leukemia, Colon Tumor, and Central Nervous System data sets when $\delta = 0$.

other hand, the bicluster 5 found by *CC* contains nine ALL and six AML columns. In such a case it has a lower biological meaning than the bicluster 1 found by *CC* because there is not a good separation between its columns, thus genes do not show similar activity patterns.

The *purity* of a bicluster B_i reveals how much the assignment of a sample to a bicluster corresponds to its effective classification. Thus

$$purity_i = \frac{|B_i \bigcap C|}{|B_i|}$$

where C is the dominant class in that bicluster. Then

$$average\ purity = \left(\sum_{i=1}^{k} purity_i\right)/k$$

Obviously, high values of *purity* correspond to higher homogeneity of the biclusters, and thus better partitioning.

Fig. 8a–c plots the average purity value on the three data sets for an increasing number of biclusters when the residue threshold δ is fixed to 0. We can observe that for the *Leukemia* data set, Fig. 8a, the purity obtained by *RWB* is always higher than that obtained by *CC*, while for the other two data sets they are comparable. In particular, the results obtained on the *Leukemia* data set are quite interesting because show the capability of the *gain* function in grouping experimental conditions having similar patterns with respect to a subset of all the genes.

7. Conclusions

The paper presented a greedy search algorithm to find overlapped biclusters enriched with a local search strategy to escape poor local minima. The proposed algorithm is guided by a gain function that combines the mean squared residue, the row variance, and the size of the bicluster through user-provided weights. Different strategies to escape local minima have been introduced and compared. Experimental results showed that the algorithm is able to obtain groups of genes co-regulated and co-expressed under specific conditions.

References

[1] C.C. Aggarwal, C.M. Procopiucand, J.L. Wolf, P.S. Yu, J.S. ParkG, Fast algorithms for projected clustering, in: Proceedings of the ACM International Conference on Management of Data (SIGMOD'99), 1999, pp. 61–72.

[2] C.C. Aggarwal, P.S. Yu, Finding generalized projected clusters in high dimensional spaces, in: Proceedings of the ACM International Conference on Management of Data (SIGMOD'00), 2000, pp. 70–81.

[3] R. Agrawal, J.C. Gehrke, D. Gunopulos, P. Raghavan, Automatic subspace clustering of high dimensional data for data mining applications, in: Proceedings of the ACM International Conference on Management of Data (SIGMOD'98), 1998, pp. 94–105.
[4] U. Alon, N. Barkai, D.A. Notterman, K. Gish, S. Ybarra, D. Mack, A.J. Levine, Broad patterns of gene expression revealed by clustering of tumor and normal colon tissues probed by oligonucleotide arrays, Proceedings of the National Academy of Sciences, USA 96 (12) (1999) 6745–6750.
[5] F. Azuaje, A cluster validity framework for genome expression data, Bioinformatics 18 (2) (2002) 319–320.
[6] F. Azuaje, N. Bolshakova, Clustering genomic expression data: design and evaluation principles, in: D.P. Berrar, W. Dubitzky, M. Granzow (Eds.), A Practical Approach to Microarray Data Analysis, Kluwer, 2003, pp. 230–245.
[7] A. Ben Dor, R. Shamir, Z. Yakhini, Clustering gene expression patterns, Journal of Computational Biology 6 (3–4) (1999) 281–297.
[8] S. Busygin, G. Jacobsen, E. Kramer, Double conjugated clustering applied to leukemia microarray data, in: SIAM Data Mining Workshop on Clustering High Dimensional Data and its Applications, 2002.
[9] S. Busygin, O. Prokopyev, P.M. Pardalos, Biclustering in data mining, Computers and Operations Research, in press.
[10] S. Busygin, O. Prokopyev, P.M. Pardalos, Feature selection for consistent biclustering via fractional 0–1 programming, Journal of Combinatorial Optimization 10 (1) (2005) 7–21.
[11] Y. Cheng, G.M. Church, Biclustering of expression data, in: Proceedings of the 8th International Conference on Intelligent Systems for Molecular Biology (ISMB'00), 2000, pp. 93–103.
[12] H. Cho, I.S. Dhillon, Y. Guan, S. Sra, Minimum sum-squared residue co-clustering of gene expression data, in: Proceedings of the Fourth SIAM International Conference on Data Mining (SDM'04), 2004.
[13] I.S. Dhillon, Co-clustering documents and words using bipartite spectral graph partitioning, in: Proceedings of the Seventh International ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD'01), 2003, pp. 269–274.
[14] I.S. Dhillon, S. Mallela, D. Modha, Information-theoretic co-clustering, in: Proceedings of the Ninth International ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD'03), 2003, pp. 89–98.
[15] M.B. Eisen, P. Spellman, P.O. Brown, P. Botstein, Cluster analysis and display of genome-wide expression pattern, Proceedings of the National Academy of Sciences, USA 8 (1998) 14863–14868.
[16] M.R. Garey, D.S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, Freeman, San Francisco, 1979.
[17] G. Getz, E. Levine, E. Domany, Coupled two-way cluster analysis of gene microarray data, in: Proceedings of the National Academy of Sciences, USA, 2000, pp. 12079–12084.
[18] Gregory Grothaus, Adeel Mufti, T.M. Murali, Automatic layout and visualisation of biclusters, Algorithms for Molecular Biology 21 (15) (2006).
[19] M. Halkidi, Y. Batistakis, M. Vazirgiannis, Cluster validity methods: Part I, SIGMOD Record 31 (2) (2002) 40–45.
[20] M. Halkidi, Y. Batistakis, M. Vazirgiannis, Cluster validity methods: Part II, SIGMOD Record 31 (3) (2002) 19–27.
[21] J.A. Hartigan, Direct clustering of a data matrix, Journal of the American Statistical Association 67 (337) (1972) 123–129.
[22] J.Y. King, R. Ferrara, Pathway analysis of coronary atherosclerosis, Physiological Genomics 23 (1) (2005) 103–118.
[23] Y. Kluger, R. Basri, J.T. Chang, M. Gerstein, Spectral biclustering of microarray data: coclustering genes and conditions, Genome Research 13 (4) (2003) 703–716.
[24] L. Lazzeroni, A. Owen, Plaid models for gene expression data, Statistica Sinica 12 (1) (2002) 61–86.
[25] S.C. Madeira, A.L. Oliveira, Biclustering algorithms for biological data analysis: a survey, IEEE Transactions on Computational Biology and Bioinformatics 1 (1) (2004) 24–45.
[26] H. Nagesh, S. Goil, A. Choundhary, Mafia: efficient and scalable subspace clustering for very large datasets, Technical report 9906-010, Northwestern University, 1999.
[27] D.J. Reiss, N.S. Baliga, R. Bonneau, Integrated biclustering of heterogeneous genomewide datasets for the inference of global regulatory networks, BMC Bioinformatics 7 (280) (2006).
[28] B. Selman, H.A. Kautz, B. Cohen, Noise strategies for improving local search, in: Proceedings of the 12th Nation Conference on Artificial Intelligence (AAAI'94), 1994, pp. 337–343.
[29] A. Tanay, Computational Analysis of Transcriptional Programs: Function and Evolution, Ph.D. Thesis, August 2005.
[30] A. Tanay, R. Sharan, M. Kupiec, R. Shamir, Revealing modularity and organization in the yeast molecular network by integrated analysis of highly heterogeneous genomewide data, Proceedings of the National Academy of Science 101 (2004) 2981–2986.
[31] A. Tanay, R. Sharan, R. Shamir, Discovering statistically significant biclusters in gene expression data, Bioinformatics 18 (Suppl. 1) (2002) S136–S144.
[32] A. Tanay, R. Sharan, R. Shamir, Biclustering algorithms: a survey, in: Srinivas Aluru (Ed.), Handbook of Computation Molecular Biology, 2006.
[33] S. Tavazoie, J.D. Hughes, M. Campbell, R.J. Cho, G.M. Church, Systematic determination of genetic network architecture, Natural Genetics 22 (1999) 281–285.
[34] J. Yang, W. Wang, H. Wang, P. Yu, δ-clusters: capturing subspace correlation in a large data set, in: Proceedings of the 18th IEEE International Conference on Data Engineering, 2002, pp. 517–528.
[35] J. Yang, W. Wang, H. Wang, P. Yu, Enhanced biclustering on expression data, in: Proceedings of the 3rd IEEE Conference on Bioinformatics and Bioengineering (BIBE'03), 2003, pp. 321–327.
[36] K.Y. Yeung, D.R. Haynor, W.L. Ruzzo, Validating clustering for gene expression data, Bioinformatics 17 (4) (2001) 309–318.