Knowl Inf Syst (2012) 30:341–358
DOI 10.1007/s10115-011-0383-7

REGULAR PAPER

BicFinder: a biclustering algorithm for microarray data analysis

Wassim Ayadi · Mourad Elloumi · Jin-Kao Hao

Received: 30 January 2010 / Revised: 25 October 2010 / Accepted: 22 January 2011 /
Published online: 18 February 2011

Abstract In the context of microarray data analysis, biclustering allows the simultaneous identification of a maximum group of genes that show highly correlated expression patterns through a maximum group of experimental conditions (samples). This paper introduces a heuristic algorithm called BICFINDER (The BICFINDER software is available at: http://www.info.univ-angers.fr/pub/hao/BicFinder.html) for extracting biclusters from microarray data. BICFINDER relies on a new evaluation function called *Average Correspondence Similarity Index* (ACSI) to assess the coherence of a given bicluster and utilizes a directed acyclic graph to construct its biclusters. The performance of BICFINDER is evaluated on synthetic and three DNA microarray datasets. We test the biological significance using a gene annotation web-tool to show that our proposed algorithm is able to produce biologically relevant biclusters. Experimental results show that BICFINDER is able to identify coherent and overlapping biclusters.

Keywords Biclustering · Heuristics · Evaluation function · Data mining · Analysis of DNA microarray data

1 Introduction

Microarray data analysis can be carried out according to at least two different and complementary perspectives. On the one hand, researchers on cancer studies are interested in categorical phenotypes like cancer occurrences, specific tumor subtypes, or cancer survivals,

W. Ayadi (✉) · M. Elloumi
UTIC, Higher School of Sciences and Technologies of Tunis, University of Tunis, 1008 Tunis, Tunisia
e-mail: ayadi@info.univ-angers.fr

M. Elloumi
e-mail: mourad.elloumi@fsegt.rnu.tn

W. Ayadi · J.-K. Hao
LERIA, University of Angers, 2 Boulevard Lavoisier, 49045 Angers, France
e-mail: hao@info.univ-angers.fr

which naturally leads to supervised classification of data [45]. *Supervised classification*, also called *class prediction* or *class discrimination*, aims to assign samples to predefined categories [2]. On the other hand, *data clustering*, i.e., *unsupervised classification*, aims to identify groups of genes, or groups of experimental conditions, that exhibit similar expression patterns. In such a context, it is particularly interesting to be able to identify simultaneously a group of genes that show similar expression trends across specific groups of experimental conditions, also called samples, [20,31]. This task is typically achieved by a particular type of clustering technique known as *biclustering* and constitutes the focus of this work. A *bicluster* is a subset of genes associated with a subset of conditions in which these genes are co-expressed. The *biclustering* problem concerns the identification of the *best* biclusters of a given dataset. Given the practical importance of the problem, many algorithms have been devised to extract good and close-to-optimal biclusters [9,16,29,32,48].

A microarray dataset is represented by a data matrix where each cell represents the gene expression level of a gene under a particular experimental condition. Formally, a *bicluster* can be defined as follows: Let $I = \{1, 2, \ldots, n\}$ be a set of indices of n genes, $J = \{1, 2, \ldots, m\}$ be a set of indices of m conditions, and $M(I, J)$ be a data matrix associated with I and J. A *bicluster* associated with the data matrix $M(I, J)$ is a couple (I', J') such that $I' \subseteq I$ and $J' \subseteq J$. Biclustering is known to be NP-hard [16]. Indeed, it is a highly combinatorial problem with a search space of order of $O(2^{|I|+|J|})$.

Existing biclustering algorithms can be grouped into two large classes [5]: those that adopt a *systematic search* approach and those that adopt a *stochastic search* one, also called *metaheuristic* approach. Algorithms that adopt a systematic search approach include greedy algorithms [9,15,16,29,44], divide-and-conquer algorithms [23,39], and enumeration algorithms [6,24,27,36,42]. Those that adopt a metaheuristic approach include neighborhood-based algorithms [13], GRASP [19], and evolutionary algorithms [12,21,22,34].

In this paper, we present a greedy algorithm, called BicFinder, for biclustering of DNA microarray data. The main features of this algorithm include the introduction of a new evaluation function called *Average Correspondence Similarity Index* (ACSI) and the utilization of a directed acyclic graph for biclusters extraction. When applied on both synthetic and real datasets, our algorithm shows better performances compared to other biclustering algorithms.

2 Our evaluation function: ACSI

To assess a given bicluster, an evaluation function is required. One of the most popular evaluation functions is called *Mean Squared Residue* (MSR) [16], which has been largely used by several biclustering algorithms [4,12,15,19,34,46,47]. However, MSR is known to be deficient to correctly assess the quality of certain types of biclusters [1,38,44]. Recently, another evaluation function called *Average Correlation Value* (ACV) has been proposed in [44]. However, the performance of ACV is known to be sensitive to noises [15]. In the following, we briefly present *Average Spearman's Rho* (ASR) proposed in [6] and our new evaluation function *Average Correspondence Similarity Index* (ACSI).

2.1 Average Spearman's Rho (ASR)

In [6], we have proposed an evaluation function called *Average Spearman's Rho* (ASR) based on Spearman's rank correlation [25].

Let (I', J') be a bicluster in data matrix $M(I, J)$, the ASR evaluation function is then defined by:

$$ASR(I', J') = 2 \max \left\{ \frac{\sum_{i \in I'} \sum_{j \in I'; j \geq i+1} \rho_{ij}}{|I'|(|I'|-1)}, \quad \frac{\sum_{k \in J'} \sum_{l \in J'; l \geq k+1} \rho_{kl}}{|J'|(|J'|-1)} \right\} \tag{1}$$

where $\rho_{ij} (i \neq j)$ is the Spearman's rank correlation associated with the row indices i and j in the bicluster (I', J') [25], $\rho_{kl} (k \neq l)$ is the Spearman's rank correlation associated with the column indices k and l in the bicluster (I', J'), and ASR $(I', J') \in [-1..1]$. A high (resp. low) ASR value, *close* to 1 (resp. *close* to -1), indicates that the genes/conditions of the bicluster are strongly (resp. weakly) correlated. The ASR function can be computed in $O(n^2 m)$.

In [7], it has been shown that Spearman's rank correlation is robust to the presence of noise in the data and does not require any normalization of the expression data matrix. Since the evaluation function ASR is based entirely on this correlation function, ASR is robust to the presence of noise.

2.2 Average Correspondence Similarity Index (ACSI)

Recently, in the area of clustering, Son and Baek [41] showed that the correlation coefficients like Spearman's rank correlation [25] or Pearson's correlation [35] are not reliable measures when conditions are few. Furthermore, a high correlation coefficient does not necessarily imply a homogeneous cluster, nor does a low correlation coefficient necessarily imply a heterogeneous cluster [37]. To avoid this difficulty, we propose a new evaluation function called *Average Correspondence Similarity Index* (ACSI) for bicluster evaluation. The proposed ACSI is based on *Concordance Index* (CI) [41], which is designed to measure clusters.

In fact, in microarray data analysis, genes are considered to belong to the same cluster if their trajectory patterns of expression levels are similar (see [30,37,40]), i.e., if they have the same profile shape (which may be either monotone increasing, or monotone decreasing, or up-down, or down-up, etc.). Our proposed ACSI is designed to keep track of the profile shape over conditions and preserves the similarity information of trajectory patterns of the expression levels.

In order to calculate ACSI, we first discretize the initial data matrix M (I, J), $I = \{1, 2, \ldots, n\}$, and $J = \{1, 2, \ldots, m\}$ into a matrix M' defined as follows:

$$M'[i, l] = \begin{cases} 1 & \text{if } M[i, l] < M[i, l+1] \\ -1 & \text{if } M[i, l] > M[i, l+1] \\ 0 & \text{if } M[i, l] = M[i, l+1] \end{cases} \tag{2}$$

with $i \in [1..n]$ and $l \in [1..m-1]$.

Let the *correspondence similarity list* between genes g_i and $g_j (i < j)$, denoted by $CSL_{i,j}$, be the list whose each element is represented by $T(M'[i, l] = M'[j, l])$ where $T(Func)$ is true, if and only if, $Func$ is true, and $T(Func)$ is false otherwise. Let $NumCSL_{i,j}$ be the number of times where we have a true value in $CSL_{i,j}$ and $MaxCSL_i = \max\{NumCSL_{i,i+1}, NumCSL_{i,i+2}, \ldots, \text{NumCSL}_{i,n}\}$. We define the *correspondence similarity index* as follows:

$$CSI(i, j, k) = \frac{\sum_{l=1}^{m-1} T(M'[i, l] = M'[j, l] = M'[k, l])}{MaxCSL_i} \tag{3}$$

with $i \in [1..n-2]$, $j \in [2..n-1]$, $k \in [3..n]$, $l \in [1..m-1]$, and $i < j < k$.

Table 1 Data matrix M

	c'_1	c'_2	c'_3	c'_4	c'_5	c'_6
g_1	10	20	5	15	40	18
g_2	20	40	10	30	24	20
g_3	23	12	8	15	29	50
g_4	4	8	2	6	5	5
g_5	12	13	8	15	31	50
g_6	23	12	8	15	29	50

Table 2 Discretized data matrix M'

	c_1	c_2	c_3	c_4	c_5
g_1	1	−1	1	1	−1
g_2	1	−1	1	−1	−1
g_3	−1	−1	1	1	1
g_4	1	−1	1	−1	0
g_5	1	−1	1	1	1
g_6	−1	−1	1	1	1

Proposition 1 *Let i, j, and $k, i < j < k$, be indices of rows in a data matrix $M(I, J)$, we have (see Proof 1 in appendix):*

$$0 \leq CSI(i, j, k) \leq 1.$$

$CSI(i, j, k)$ indicates the proportion of true's, i.e., the change in the same direction, that exists between rows i, j, and k in the same set of columns. This enables to see how genes g_i, g_j, and g_k behave over a subset of conditions.

Finally, for the whole bicluster, we define the Average Correspondence Similarity Index (ACSI) for the row i ($i \in I'$ and $i < j < k$):

$$ACSI_i(I', J') = 2 * \frac{\sum_{j \in I'; j \geq i+1} \sum_{k \in I'; k \geq j+1} CSI(i, j, k)}{|I''|(|I''| - 1)} \tag{4}$$

where $|I''|$ is the number of rows (genes) in the bicluster without the row index i.

Proposition 2 *Let (I', J') be a bicluster in a data matrix $M(I, J)$. We have (see Proof 2 in appendix):*

$$0 \leq ACSI_i(I', J') \leq 1.$$

Proposition 3 *The ACSI function can be computed in a time $O(n^2 m)$, where n is the number of the rows, and m is the number of the columns of the data matrix (see Proof 3 in appendix).*

A high (resp. low) ACSI value, *close* to 1 (resp. *close* to 0), indicates that the genes of the bicluster are strongly (resp. weakly) correlated.

To illustrate the computation of ACSI, we use the following example: Let M be a data matrix (Table 1). When we discretize M, using Eq. 2, we obtain the data matrix M' (Table 2).

Suppose that $Bic = (\{g_1, g_2, g_4, g_5\}; \{c'_1, c'_2, c'_3, c'_4\})$ is a bicluster. The ACSI of Bic is equal to $ACSI_1 = \frac{CSI(1,2,4)+CSI(1,2,5)+CSI(1,4,5)}{3(3-1)/2} = \frac{3/4+3/4+3/4}{3} = 0.75$.

The difference between CI [41] and ACSI is that CI can only measure the coherence between two genes with all conditions while ACSI measures the coherence of the whole bicluster, i.e., subset of rows under a subset of columns.

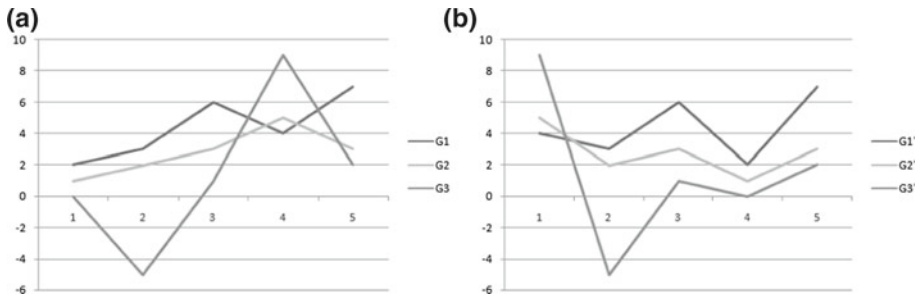

Fig. 1 The expression profiles of two biclusters B1 **a** and B2 **b** with 3 genes and 5 conditions. The genes of B1 show different profiles while those of B2 show similar patterns. B2 is a better bicluster than B1

With the discretization of the data matrix M, both ACSI and CI can only count the number of conditions that indicate the same sign of change between rows. Likewise, both ACSI and CI may lose the information concerning the change of the bicluster size [41]. That is why, we also compute the ASR evaluation function to take into account the information concerning the change of the bicluster size. Indeed, the existing evaluation functions adopted for biclustering can roughly be classified into two families: *numerical* measures and *qualitative* measures. *Numerical* measures, like Pearson's correlation or Euclidean distance, are easy to compute but they are quite sensitive toward outliers and noise. *Qualitative* measures, like measures that consider only *ups*, *downs*, and *no change* of conditions, are very sensitive to precise the values of changes. So, ASR, which is based on Spearman's rank correlation, can be considered as a good compromise between numerical measures and qualitative ones.

Finally, the example of Fig. 1 shows how ACSI can better discriminate biclusters having the same profile shape than other evaluation functions. We consider two biclusters B1 and B2 composed of three genes and five conditions. B1 = {G1(2, 3, 6, 4, 7), G2(1, 2, 3, 5, 3), G3(0, −5, 1, 9, 2)} and B2 = {G1(4, 3, 6, 2, 7), G2(5, 2, 3, 1, 3), G3(9, −5, 1, 0, 2)}. B2 is thus the same as B1, with the first and the fourth conditions of each gene permuted. According to these profile shapes shown in Fig. 1, B2 is intrinsically better than B1. Yet, the ACV score [44] (based on Pearson's correlation coefficient) of B1 and B2 is equally 0.498 just like ASR [6] (based on Spearman's correlation coefficient) that leads to the same value of 0.667 for both biclusters. When we apply our ACSI evaluation function,[1] we get an ACSI score of 0.334 (i.e., weakly correlated) for B1 and an ACSI score of 1.0 (i.e., strongly correlated) for B2, showing clearly B2 is better than B1.

The aim of the next section is to assess the quality of the proposed ACSI evaluation function in comparison with ASR and two popular functions that are the *Mean Squared Residue* (MSR) and the *Average Correlation Value* (ACV).

2.3 Study of the ACSI evaluation function

As in [44], we compare the ACSI evaluation function with the most used evaluation ones: MSR [16] and ACV [44]. We also compare ACSI with ASR [6].

To compare evaluation functions mentioned above, we use seven data matrices, named $M_1, M_2, \ldots, M_7$, representing all typical biclusters [31,44]: a constant bicluster (M_1), a constant rows bicluster (M_2), a constant column bicluster (M_3), a bicluster with coherent

[1] The values of ACSI and ACV (between 0 and 1) are normalized to drop in [−1 .. 1] to be compared with the scores of ASR which are between −1 and 1.

Table 3 ACSI versus ASR, MSR and ACV

	$M1$	$M2$	$M3$	$M4$	$M5$	$M6$	$M7$
MSR	0.00	0.00	0.00	0.00	0.62	2.425	131.87
ACV	1.00	1.00	1.00	1.00	1.00	1.00	0.84
ASR	1.00	1.00	1.00	1.00	1.00	1.00	0.99
ACSI	1.00	1.00	1.00	1.00	1.00	1.00	1.00

values using an additive model (M_4), a bicluster with coherent values using a multiplicative model (M_5), a bicluster with coherent values using a multiplicative model where the first row of M_5 is multiplied by 10 (M_6), and a coherent evolution bicluster (M_7).

For each bicluster $M_1, M_2, \ldots, M_7$, we calculate the coherence score with our evaluation functions ACSI and ASR. The values of MSR and ACV were taken from Table 2 in [44]. The results are summarized in Table 3. Concerning MSR, a low (resp. high) value, *close* to 0 (resp. higher than a fixed threshold), indicates that the genes/conditions of the bicluster are strongly (resp. weakly) correlated. Concerning ACV, a high (resp. low) value, *close* to 1 (resp. *close* to 0), indicates that the genes/conditions of the bicluster are strongly (resp. weakly) correlated. According to Table 3, the ASR, ACSI, ACV, and MSR functions are perfect to assess the quality of biclusters M_1, M_2, M_3, and M_4. However, MSR is deficient on M_6 and M_7, confirming the claim that MSR is not appropriate on certain types of biclusters [1,38,44]. On the other hand, ASR and ACV are perfect to assess the quality of biclusters M_5 and M_6. ASR is slightly better than ACV when applied on M_7. Let us notice that ACSI is perfect when applied in all biclusters $M_1, M_2, \ldots, M_7$.

3 BicFinder: our proposed biclustering algorithm

The BicFinder biclustering algorithm is based on the construction of a *Directed Acyclic Graph* (DAG) to represent the different correspondence similarity lists CSL_{ij}, $1 \leq i < j \leq n$, between rows. ASR and ACSI are used as evaluation functions. BicFinder operates in four main steps: dicretization of the data matrix, construction of the associated DAG, and extraction and selection of biclusters. The discretization of the data matrix M is obtained thanks to Eq. 2.

3.1 Construction of DAG and, extraction and selection of biclusters

A DAG associated with a data matrix M' is represented as follows: a node n_i represents a gene g_i and an arc connecting a node n_i to a node n_j if and only if $i < j$. To each arc (n_i, n_j), we assign $CSL_{i,j}$.

The next step extracts coherent biclusters. For each node n_i, first we initialize the associated bicluster $B_i = (I'_i, J'_i)$ to $(\emptyset, \emptyset)$. Then, we sort the arcs leaving n_i in a decreasing way according to the number of true's associated with each arc. We consider then the sorted arcs successively. Let (n_i, n_k) the current arc, if the evaluation function ACSI associated with the bicluster $(I'_i \cup \{g_i, g_k\}, J'_i \cup \{c_l, c_{l+1}$ such that $T(M'[i,l] = M'[k,l]) = true\})$ is greater than or equal to a fixed threshold α, then we set $B_i = (I'_i \cup \{g_i, g_k\}, J'_i \cup \{c_l, c_{l+1}$ such that $T(M'[i,l] = M'[k,l]) = true\})$. This process is repeated until all the arcs leaving n_i are processed.

Finally, we consider only the obtained biclusters for which the ASR evaluation function is greater than or equal to another fixed threshold β. The set of such biclusters represents a solution to our problem.

To describe formally the BICFINDER algorithm, let us define some variables:

M(resp. M'): data matrix (resp. discretized data matrix),
$\mathcal{B}$: set of biclusters,
n_i(resp. n_k): node that represents a gene g_i (resp. g_k),
$B_i = (I'_i, J'_i)$: current bicluster,
Ic, Jc: current subset of genes and current subset of conditions,
α, β: quality thresholds, respectively, according to ACSI and ASR.

Algorithm 1 BICFINDER

Input: M, α, β ; **Output**: $\mathcal{B}$
Discretize M using Equation 2 to obtain M' // Discretization step
Construct the DAG associated with M' // Construction step
$\mathcal{B} = \emptyset$ // Extraction step
for each n_i in the DAG **do**
  $I'_i = \emptyset$; $J'_i = \emptyset$; // $B_i = (I'_i, J'_i)$
  Sort the arcs leaving n_i in a decreasing way according to the number of true's
  for each arc (n_i, n_k) **do**
    $Ic = I'_i \cup \{g_i, g_k\}$; $Jc = J'_i \cup \{c_l, c_{l+1}$ such that $T(M'[i,l] = M'[k,l]) = true\}$;
    if $ACSI_i(Ic, Jc) \geq \alpha$ **then** $B_i = (Ic, Jc)$
  endfor
  $\mathcal{B} = \mathcal{B} \cup B_i$
endfor
for each bicluster $B_i = (I'_i, J'_i)$ in $\mathcal{B}$ **do** // Selection step
  if $ASR(I'_i, J'_i) < \beta$ **then** $\mathcal{B} = \mathcal{B} \backslash B_i$
endfor
Return $\mathcal{B}$

Proposition 4 *Time complexity of* BICFINDER *is* $O(n^5 m)$*, where n is the number of the rows, and m is the number of the columns of the data matrix (see Proof 4 in appendix).*

3.2 An illustrative example

By using the previous example in Sect. 2.2, we represent the data matrix M'(Table 2) by the DAG that represents all the genes with their CSL (see Fig. 3). Let us set $\alpha = 0.75$ and $\beta = 0.9$.

Figure 3 shows the first node g_1 with its CSL, i.e., arcs {(a),(b),(c),(d),(e)}. We have two $MaxCSL_1$:[2] arc (a) with $NumCSL_{1,2} = 4$ and arc (d) with $NumCSL_{1,5}$ =4. When we compute $ACSI_1$ of (a) and (d), we find that $ACSI_1$ is equal to 0.75, i.e., $ACSI_1 = \frac{CSI(1,2,5)}{2(2-1)/2} = \frac{3/4}{1} = 0.75$, then we can merge them in B_1, giving $B_1 = (\{g_1, g_2, g_5\}; \{c'_1, c'_2, c'_3, c'_4\})$.

After that, we test ACSI of (a) and (d) with (b) and get $ACSI_1 = \frac{CSI(1,2,3)+CSI(1,2,5)+CSI(1,3,5)}{3(3-1)/2} = 0.58$. Hence, this case is discarded because ACSI is lower than 0.75. Then, we test ACSI of (a) and (d) with (c) and we obtain $ACSI_1 = \frac{CSI(1,2,4)+CSI(1,2,5)+CSI(1,4,5)}{3(3-1)/2} = 0.75$, then we can merge them. So, B_1 becomes $B_1 = (\{g_1, g_2, g_4, g_5\}; \{c'_1, c'_2, c'_3, c'_4\})$. Finally, with the node g_1, we test ACSI of (a), (d) and

[2] Arcs with stars.

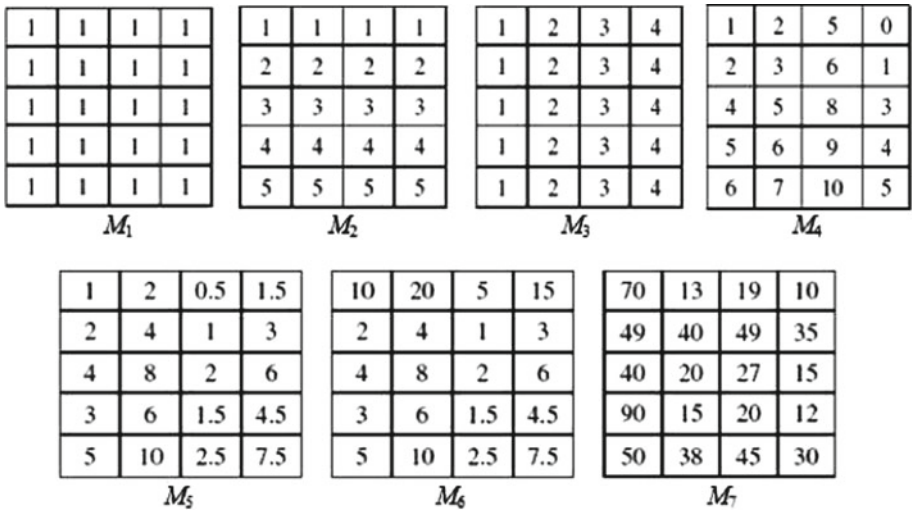

Fig. 2 Different typical biclusters

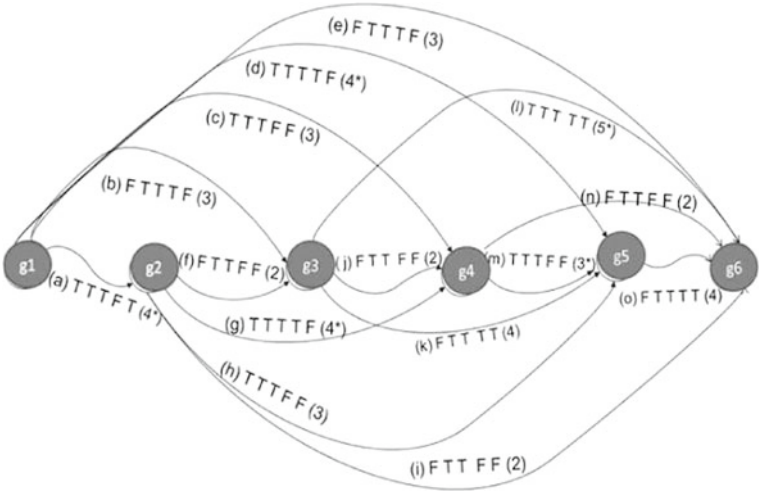

Fig. 3 DAG associated with M'

(c) with (e), then we obtain $ACSI_1 = \frac{CSI(1,2,4)+CSI(1,2,5)+CSI(1,2,6)+CSI(1,4,5)+CSI(1,4,6)+CSI(1,5,6)}{4(4-1)/2} = 0.625$. This case is discarded because ACSI is lower than 0.75. This completes the first node g_1 of DAG, and we repeat the same process for all the other nodes.

After that, we select only the biclusters that are higher than a β threshold, i.e., $\beta = 0.9$. At the end, we obtain two biclusters $B_1 = (\{g_1, g_2, g_4, g_5\}; \{c'_1, c'_2, c'_3, c'_4\})$ with ASR = 0.9 and $B_3 = (\{g_3, g_5, g_6\}; \{c'_2, c'_3, c'_4, c'_5, c'_6\})$ with ASR = 1. So, $\mathcal{B} = \{(\{g_1, g_2, g_4, g_5\}; \{c'_1, c'_2, c'_3, c'_4\})$; $(\{g_3, g_5, g_6\}; \{c'_2, c'_3, c'_4, c'_5, c'_6\})\}$ is the output of our algorithm.

4 Results

We assess the BicFinder algorithm on both synthetic and real DNA microarray datasets. For synthetic data, we compare our results with the results of some prominent biclustering algorithms used by the community, namely, CC [16], OPSM [9], ISA [10], and Bimax [39]. For these references, we have used *Biclustering Analysis Toolbox* (BicAT) which is a recent software platform for clustering-based data analysis that integrates all these biclustering algorithms [8]. For real datasets, in addition to the algorithms mentioned before, we compare our algorithm with the results of Samba [42], RMSBE [29], MOEA [34], MOPSOB [28], and CMOPSOB [26]. For both synthetic and real datasets, the parameters of BicFinder are fixed after a number of simulations. The parameter settings used for CC, OPSM, ISA, and Bimax for the synthetic datasets are the default values as used in [29]. For all the other experiments, we report the results of the compared algorithms from their original papers. The BicFinder algorithm was implemented in Java and was run on a PC Intel Core 2 Duo T6400 with 2.0 GHz CPU and 3.5Gb RAM.

4.1 Computational results of BicFinder on synthetic data

According to [13,14,44], we have randomly generated two types of synthetic datasets [3] of size $(|I|, |J|)= (200, 20)$. Different types of biclusters are embedded like constant, additive, multiplicative, and coherent evolution biclusters. The first (resp. second) dataset contains biclusters without (resp. with) overlapping. To obtain statistically stable results, for each type of datasets, we have generated 10 problem instances by randomly inserting the biclusters at different places in the data matrix.

4.1.1 Comparison criteria

Following [14], we use the following two ratios to evaluate our biclustering algorithm:

$$\theta_{\text{Shared}} = \frac{S_{\text{cb}}}{Tot_{\text{size}}} * 100 \tag{5}$$

where S_{cb} is the portion size of biclusters correctly extracted, and Tot_{size} is the total size of correct biclusters.

$$\theta_{\text{NotShared}} = \frac{S_{\text{ncb}}}{Tot_{\text{size}}} * 100 \tag{6}$$

where S_{ncb} is the portion size of biclusters not correctly extracted, and Tot_{size} is the total size of corrected biclusters.

The ratio θ_{Shared} (resp. $\theta_{\text{NotShared}}$) expresses the percent of shared (resp. not shared) biclusters volume that corresponds (resp. does not corresponds) with the real biclusters. In fact, when θ_{Shared} (resp. $\theta_{\text{NotShared}}$) is equal to 100% the algorithm extracts the correct (resp. not correct) biclusters. A perfect solution has θ_{Shared} =100% and $\theta_{\text{NotShared}}$=0%, respectively, thus, the exact number of genes and conditions of implanted biclusters.

4.1.2 Results and comparisons

For our algorithm, we fix the threshold of ACSI $\alpha = 0.85$ and threshold of ASR $\beta = 0.3$. We run all the algorithms and we select the 4 biclusters obtained by each algorithm which

[3] Datasets available at http://www.info.univ-angers.fr/pub/hao/BicFinder.html.

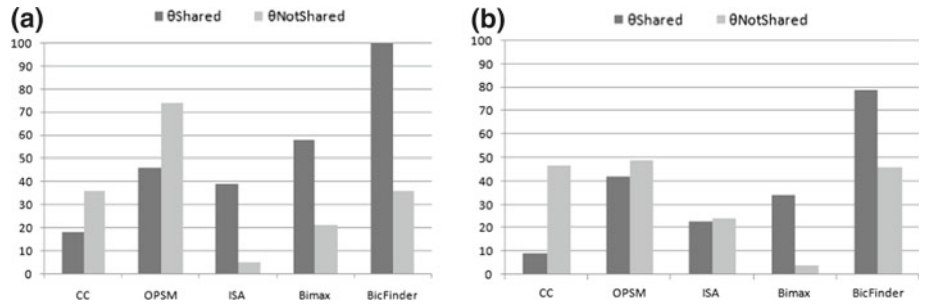

Fig. 4 **a** Results of BICFINDER and comparison with other algorithms on synthetic data without overlapped biclusters. **b** Results of BICFINDER and comparison with other algorithms on synthetic data with overlapped biclusters

best fit the 4 real biclusters. We compute the θ_{Shared} and the $\theta_{\text{NotShared}}$ for each algorithm to show the averaged percentage of volume of the resulting biclusters which is shared and not shared with the real biclusters. The objective of this experiment is to determine which algorithm is able to extract all types of biclusters. Figure 4 shows the statistics of the best biclusters provided by each algorithm for the two datasets.

As we can see in Fig. 4a, BICFINDER obtains 100% of the volume of the correctly extracted biclusters, with an extra volume that represents 36.18 %. In fact, to enlarge a bicluster, BICFINDER tries to add genes that have the maximum similarity with each other over the same subset of conditions. Hence, it can provide an extra volume only on conditions but gives exactly the correct number of genes. This is due to the higher number of extracted conditions. In fact, BICFINDER tries to keep the maximum number of conditions which have the same gene expression profile shape.

On the other hand, the best of the studied algorithms, i.e., Bimax, can extract only 58.18% of implanted biclusters with 21.39% of extra volume. The discretization preprocessing used by Bimax may render it impossible to identify the elements in the coherent biclusters and miss some implanted biclusters. ISA has, despite the low value of $\theta_{\text{NotShared}}$ (5.31%), a poor performance since it extracts only 39.38 % of the implanted biclusters ($\theta_{\text{Shared}} = 39.38\%$). In fact, ISA uses only upregulated and downregulated constant expression values. When coherent biclusters exist, ISA may miss some rows and columns of the implanted biclusters. As to CC, when the signal of the implanted biclusters is weak, the greedy nature of CC may delete some rows and columns of the implanted biclusters at the beginning of the algorithm and miss definitively the deleted rows and columns in the output biclusters. OPSM seeks only up- and downregulation expression values with coherent evolution. Its performance decreases when there exist constant biclusters.

Figure 4b illustrates the statistics of the best biclusters provided by each algorithm for the second dataset. As we can see in Fig. 4b, the results with BICFINDER present the highest coverage of the correctly extracted biclusters (79.94%) with an extra volume that represent 46.11% of implanted biclusters. On the contrary, the best of the studied algorithms, i.e., OPSM, can extract only 42.87% of implanted biclusters with 49.31% of extra volume. To find overlapped biclusters in a given matrix, some algorithms, e.g., CC, need to mask the discovered biclusters with random values which is not necessary for BICFINDER. ISA and OPSM are sensitive to overlapping biclusters. They use a normalization step in the first preprocessing step. With overlapping biclusters, the expression value range after normalization becomes narrower. Figure 4b shows that BICFINDER is marginally affected by the implanted

Table 4 Five biclusters found on human B-cell Lymphoma dataset

Genes	Conditions	Size	ASR
649	86	55,814	0.3064
472	91	42,952	0.3246
588	86	50,568	0.3107
596	83	49,468	0.3009
1,018	41	41,738	0.2746

overlap biclusters compared to other algorithms. This study tends to show that, contrary to existing algorithms, BICFINDER is able to extract all types of biclusters.

4.2 Computational results of BICFINDER on real data

In this section, we show computational results of BICFINDER on three well-known real datasets. For the experiments, the two thresholds of BicFinder, i.e., α (ACSI threshold) and β (ASR threshold), are fixed after a number of simulations. To proceed, we fix one threshold and tune the other, and vice-versa. For each experiment, five values are tested between 0.5 and 1 with a stepwise of 0.1 for α and ten values are tested between 0.1 and 1 with a stepwise of 0.1 for β. For each combination, we compute the p-values of the obtained biclusters. Note that a smaller p-value, *close* to 0, is indicative of a better match [43]. When a good combination of α and β is identified, we test ten more values above and below the fixed thresholds with a stepwise of 0.01 (for both α and β) to obtain more precise values that lead to biclusters with the lowest p-values.

4.2.1 Human B-cell lymphoma dataset

The Human B-cell Lymphoma dataset [3] contains 4026 genes and 96 conditions.[4] As in [12,16,26,28,34], we use the criterion of the coverage which is defined as the total number of cells in microarray data matrix covered by the obtained biclusters. For this experiment, the two parameters of BICFINDER α and β are experimentally set to 0.85 and 0.27. The running time of BICFINDER on this test was 78 min.

We select one hundred biggest biclusters out of 727 with high ASR like [12,16,26,28]. These biclusters cover 55.89% of the genes, 100% of the conditions, and in total 44.24% cells of the expression data matrix.

On the contrary, in [26], the authors report an average coverage of 38.3% cells of the dataset matrix, while a coverage of 20.96%, 36.9%, and 36.81% cells is reported in [28,34], and [16], respectively.

Table 4 shows the information of five biclusters out of the one hundred biclusters found by BICFINDER on the Human lymphoma dataset with their ASR. The largest bicluster, in this table, is of size 55,814.

This implies that BICFINDER can generate maximal biclusters with high coverage of a data matrix.

4.2.2 Saccharomyces cerevisiae dataset

The *Saccharomyces cerevisiae* dataset[5] contains the expression levels of 2993 genes under 173 experimental conditions. In order to evaluate the biological relevance of our proposed

[4] Available at http://arep.med.harvard.edu/biclustering/.

[5] Available at http://www.tik.ethz.ch/sop/bimax/.

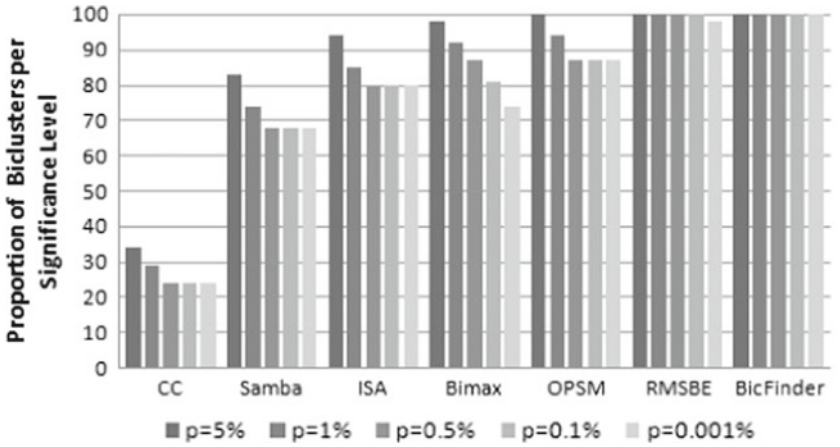

Fig. 5 Proportions of Biclusters significantly enriched by GO annotations

biclustering algorithm, we compute the p-values to indicate the quality of the extracted biclusters. For this experiment, the two parameters of BicFinder α (ACSI threshold) and β (ASR threshold) are experimentally set as follows: $\alpha = 0.85$ and $\beta = 0.29$. The running time of BicFinder on this test was 484 min.

Following the same process as in [18,29,39], we extract the largest 100 biclusters out of 1,163. The results of BicFinder are compared against the reported scores of RMSBE, Bimax, OPSM, ISA, Samba, and CC from [29,39]. The idea is to determine whether the set of genes discovered by biclustering algorithms shows significant enrichment with respect to a specific *Gene Ontology* (GO) annotation. We use the web-tool *FuncAssociate* [11] for this purpose. *FuncAssociate* computes the adjusted significance scores for each bicluster, i.e., adjusted p-values (p). Indeed, the adjusted significance scores assess genes in each bicluster by computing , which indicates how well they match with the different GO categories.

Figure 5 presents different significant scores p for each algorithm over the percentage of total extracted biclusters. On the one hand, BicFinder and RMSBE seem to outperform other algorithms. The BicFinder (resp. RMSBE) result shows that 100% (resp. 98%) of discovered biclusters are statistically significant with $p < 0.001\%$. On the other hand, apart from CC, other algorithms have reasonably good performance, i.e., the best of the other compared algorithms, OPSM, has 87% of biclusters with $p < 0.001\%$. CC underperforms because it is unable to find coherent biclusters and its lack of robustness against noise.

4.2.3 Yeast cell-cycle dataset

The *Yeast Cell-Cycle* dataset is described in [43]. This dataset is processed in [16] and publicly available from [17]. It contains the expression profiles of more than 6000 yeast genes measured at 17 conditions over two complete cell cycles. In our experiments, we use 2884 genes selected by [16]. To assess the quality of the extracted biclusters, we use a well-known web-tool on yeast cell-cycle to search for the significant shared Gene Ontology terms of the groups of selected genes. For this experiment, the two parameters of BicFinder α (ACSI threshold) and β (ASR threshold) are experimentally set as follows: $\alpha = 0.80$ and $\beta = 0.45$. The running time of BicFinder on this test was 5 minutes.

Table 5 Most significant shared GO terms (process, function, component) for two biclusters on yeast data extracted by BicFinder

Biclusters	Biological process	Molecular function	Cellular component
92 genes × 16 conditions	Translation (59.7%, 4.33e-30)	Structural constituent of ribosome (52.1%, 3.67e-48)	Cytosolic ribosome (53.2%, 6.22e-56)
	Cellular protein metabolic process (65.2%, 1.48e-18)	Structural molecule activity (53.2%, 3.76e-40)	Cytosolic part (54.3%, 8.15e-52)
	Protein metabolic process (65.2%, 1.43e-17)		
50 genes × 17 conditions	Cellular response to DNA damage stimulus (37.3%, 3.24e-14)	Double-stranded DNA binding (9.8%, 0.00026)	Replication fork (21.6%, 3.36e-12)
	Response to DNA damage stimulus (37.3%, 5.27e-13)	Cyclin-dependent protein kinase regulator activity (7.8%, 0.00124)	Chromosome (35.3%, 1.94e-10)
			Chromosomal part (31.4%, 5.28e-09)

In order to identify the biological annotations for the biclusters, we use *GOTermFinder*[6] which is a tool available in the *Saccharomyces Genome Database* (SGD). *GOTermFinder* is designed to search for the significant shared GO terms of the groups of genes and provides users with the means to identify the characteristics that the genes may have in common. We present the significant shared GO terms (or parent of GO terms) used to describe the two selected set of genes with 92 genes × 16 conditions and 50 genes × 17 conditions in each bicluster with ASR equal to 0.4543 and 0.7844, respectively, for biological process, molecular function and cellular component. As [33], we report the most significant GO terms shared by these biclusters. For example, with the first bicluster (Table 5), the genes (YEL034W, YER074W, YER117W, YER131W, YGR214W, YHL001W, YIL069C, YJL111W, YJL136C, YJL177W, YJL189W, YJL190C, YJR123W, YKL056C, YKL156W, YKL180W, YKR057W, YKR094C, YLR029C, YLR048W, YLR075W, YLR150W, YLR167W, YLR185W, YLR248W, YLR249W, YLR325C, YLR344W, YLR367W, YLR380W, YLR406C, YLR441C, YLR448W, YML026C, YML063W, YML073C, YMR143W, YMR225C, YNL030W, YNL067W, YNL096C, YNL162W, YNL301C, YNL302C, YOL039W, YOL040C, YOL127W, YOL139C, YOR167C, YOR234C, YOR293W, YOR312C, YOR369C, YPL037C, YPL081W, YPL090C, YPL143W, YPR043W, YPR102C, YPR163C) are particularly involved in the cellular protein metabolic process and protein metabolic process. The values within parentheses after each GO term in Table 5, such as (59.7%, 4.33e-30) for *translation* in the first bicluster, indicate the cluster frequency and the statistical significance. The cluster frequency (59.7%) shows that out of 92 genes in the first bicluster 55 belong to this process, and the statistical significance is provided by a p-value of 4.33e-30 (highly significant).

Figure 6 shows the two biclusters of Table 5 found by BicFinder algorithm on the yeast dataset. From a visual inspection of the biclusters presented, we can notice that the genes do present a similar behavior under the selected conditions. This fact confirms the claim that in microarray data analysis, genes are considered to be in the same cluster if their trajectory patterns of expression levels are similar across a set of conditions [30,37,40].

The experiments on these real datasets show that our proposed algorithms can identify biclusters with a high biological relevance.

[6] Available at http://db.yeastgenome.org/cgi-bin/GO/goTermFinder.

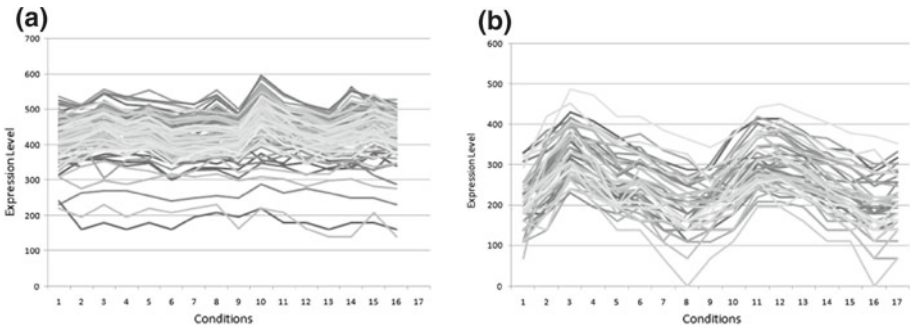

Fig. 6 Two Biclusters found by BICFINDER on yeast data: **a** 92 genes × 16 conditions with ASR = 0.4543 **b** 50 genes × 17 conditions with ASR = 0.7844

5 Conclusion

We have proposed a novel greedy biclustering algorithm called BICFINDER. BICFINDER, do not require fixing a minimum or a maximum number of genes or conditions, enabling a generation of diversified biclusters.

BICFINDER is based on several new features including a more informative evaluation function, *Average Correspondence Similarity Index* (ACSI), and an effective bicluster extraction mechanism using a *Directed Acyclic Graph* (DAG).

To assess the performance of BICFINDER algorithm, experiments were done on both synthetic and real DNA microarray datasets. This experimental study that we have achieved showed highly competitive results of BICFINDER in comparison with other popular biclustering algorithms. Biological significance of biclustering results has been verified using *Gene Ontology* (GO) annotations. This study also shows that the BICFINDER algorithm competes very favorably with several existing algorithms in terms of the percentage and the number of functionally enriched biclusters for p-values.

Acknowledgments We are grateful to the reviewers for their careful reviews and highly helpful comments. The work is partially supported by the French *Biogenouest* Network and the Region "Pays de La Loire" *via* the "Bioinformatique Ligérienn" Project.

Appendix

Proof 1 In the same set of columns, $\sum_{l=1}^{m-1} T(M'[i,l] = M'[j,l] = M'[k,l])$ represents the number of true's between indices i, j, and k, and $MaxCSL_i$ represents the maximum number of true's between $\{(i, i+1), (i, i+2), \ldots, (i, n)\}$, i.e., $MaxCSL_i = \max\{NumCSL_{i,i+1}, NumCSL_{i,i+2}, \ldots, \text{NumCSL}_{i,n}\}$. So, the number of true's in $\sum_{l=1}^{m-1} T(M'[i,l] = M'[j,l] = M'[k,l])$ is usually smaller than or equal to $MaxCSL_i$. Since we have:

$\sum_{l=1}^{m-1} T(M'[i,l] = M'[j,l] = M'[k,l]) \geq 0$, and $MaxCSL_i \geq 0$, then we have:

$0 \leq \frac{\sum_{l=1}^{m-1} T(M'[i,l]=M'[j,l]=M'[k,l])}{MaxCSL_i} \leq 1$, i.e., $0 \leq CSI(i, j, k) \leq 1$. □

Proof 2 Indeed, we have $\frac{|I''|(|I''|-1)}{2}$ values of CSI to calculate. For $CSI(i, j, k)$, $i < j < k$, we have then:

$0 \leq CSI(i, j, k) \leq 1$, then, $0 \leq \sum_{j \in I'; j \geq i+1} \sum_{k \in I'; k \geq j+1} CSI(i, j, k) \leq \frac{|I''|(|I''|-1)}{2}$, hence, $0 \leq \frac{\sum_{j \in I'; j \geq i+1} \sum_{k \in I'; k \geq j+1} CSI(i,j,k)}{\frac{|I''|(|I''|-1)}{2}} \leq 1$, i.e., $0 \leq ACSI_i(I', J') \leq 1$. □

Proof 3 The computing of $T(M'[i, l] = M'[j, l] = M'[k, l])$ can be achieved in a time $O(1)$. We repeat this function $m - 1$ times. We have $\frac{|I''|(|I''|-1)}{2}$ values of *Correspondence Similarity Index* to calculate, i.e., $O(n^2)$. Hence, the ACSI function can be computed in a time $O(n^2 m)$. □

Proof 4 Time complexity of the discretization step is $O(nm)$. Indeed, this step is achieved *via* a scanning of the whole data matrix M that is of size nm.

Time complexity of the construction step is $O(n^2 m)$. In fact, the construction of an arc and the counting of the number of the associated true's in the DAG is made in $O(m)$. Since the total number of arcs is $\frac{n(n-1)}{2}$, then time complexity of the construction step is $O(n^2 m)$.

Time complexity of the extraction step is $O(n^5 m)$. Indeed, in the worst case, a node n_i has $(n - 1)$ outgoing arcs. So, the determination of the outgoing arcs of a node n_i that have the maximum number of true's is achieved in $O(n)$. Each one of these arcs will be compared to a bicluster by computing the corresponding ACSI evaluation function. According to Proposition 3, this is done in a time $O(n^2 m)$. In the worst case, the comparison between a bicluster and all the outgoing arcs, that have the maximum number of true's, is achieved in a time $O(n^3 m)$. In the worst case, we have $O(n)$ biclusters, so the comparison between all the existing biclusters and all the outgoing arcs, that have the maximum number of true's, is achieved in $O(n^4 m)$. We have n nodes, so time complexity of the extraction step is $O(n^5 m)$.

Time complexity of the selection step is $O(n^4 m)$. In fact, for a bicluster, the ASR evaluation function is computed in a time $O(n^2 m)$. In the worst case, we have $\frac{n(n-1)}{2}$ extracted biclusters. Thus, this step is achieved in a time $O(n^4 m)$.

Hence, time complexity of BicFinder is $O(n^5 m)$.

References

1. Aguilar-Ruiz JS (2005) Shifting and scaling patterns from gene expression data. Bioinformatics 21:3840–3845
2. Akadi A, Amine A, El Ouardighi A, Aboutajdine D (2010) A two-stage gene selection scheme utilizing MRMR filter and GA wrapper. Knowl Inf Syst, Published online: 10 March 2010
3. Alizadeh AA, Eisen MB, Davis RE et al (2000) Distinct types of diffuse large b-cell lymphoma identified by gene expression profiling. Nature 403:503–511
4. Angiulli F, Cesario E, Pizzuti C (2008) Random walk biclustering for microarray data. J Inf Sci 178:1479–1497
5. Ayadi W, Elloumi M (2011) Algorithms in computational molecular biology: techniques, approaches and applications, chapter biclustering of microarray data. In: Wiley book series on bioinformatics : computational techniques and engineering, Wiley-Blackwell, John Wiley & Sons Ltd., New Jersey (Publish.) (to appear)
6. Ayadi W, Elloumi M, Hao JK (2009) A biclustering algorithm based on a bicluster enumeration tree: application to dna microarray data. BioData Min 2(1):9
7. Balasubramaniyan R, llermeier H, Weskamp E, Kamper J (2005) Clustering of gene expression data using a local shape-based similarity measure. Bioinformatics 21:1069–1077
8. Barkow S, Bleuler S, Prelic A, Zimmermann P, Zitzler E (2006) Bicat: a biclustering analysis toolbox. Bioinformatics 22(10):1282–1283
9. Ben-Dor A, Chor B, Karp R, Yakhini Z (2002) Discovering local structure in gene expression data: the order-preserving submatrix problem. In: RECOMB '02: proceedings of the sixth annual international conference on computational biology. ACM, New York, pp 49–57
10. Bergmann S, Ihmels J, Barkai N (2004) Defining transcription modules using large-scale gene expression data. Bioinformatics 20(13):1993–2003

11. Berriz GF, King OD, Bryant B, Sander C, Roth FP (2003) Characterizing gene sets with funcassociate. Bioinformatics 19(18):2502–2504
12. Bleuler S, Prelic A, Zitzler E (2004) An ea framework for biclustering of gene expression data. In: Proceedings of congress on evolutionary computation. pp 166–173
13. Bryan K, Cunningham P, Bolshakova N (2006) Application of simulated annealing to the biclustering of gene expression data. In: IEEE Transactions on information technology on biomedicine, 10(3): 519–525
14. Cano C, Adarve L, Lopez J, Blanco A (2007) Possibilistic approach for biclustering microarray data. In: Computers in biology and medicine, 37, pp 1426–1436
15. Cheng KO, Law NF, Siu WC, Liew AW (2008) Identification of coherent patterns in gene expression data using an efficient biclustering algorithm and parallel coordinate visualization. BMC Bioinformatics 9(210):1282–1283
16. Cheng Y, Church GM (2000) Biclustering of expression data. In: Proceedings of the eighth international conference on intelligent systems for molecular biology. AAAI Press, pp 93–103
17. Cheng Y, Church GM (2006) Biclustering of expression data. Technical report (supplementary information)
18. Christinat Y, Wachmann B, Zhang L (2008) Gene expression data analysis using a novel approach to biclustering combining discrete and continuous data. IEEE/ACM Trans Comput Biol Bioinform 5(4): 583–593
19. Dharan A, Nair AS (2009) Biclustering of gene expression data using reactive greedy randomized adaptive search procedure. BMC Bioinform 10(Suppl 1):S27
20. Dimaggio P, Mcallister S, Floudas C (2008) Biclustering via optimal re-ordering of data matrices in systems biology: rigorous methods and comparative studies. BMC Bioinform 9(1):458
21. Divina F, Aguilar-Ruiz JS (2007) A multi-objective approach to discover biclusters in microarray data. In: GECCO '07: Proceedings of the 9th annual conference on genetic and evolutionary computation. ACM, New York
22. Gallo CA, Carballido JA, Ponzoni I (2009) Microarray biclustering: A novel memetic approach based on the pisa platform. In: EvoBIO '09: Proceedings of the 7th European conference on evolutionary computation, machine learning and data mining in bioinformatics. Springer, Berlin, pp 44–55
23. Hartigan JA (1972) Direct clustering of a data matrix. J American Statistical Association 67(337):123–129
24. Jiang D, Pei J, Ramanathan M, Lin C, Tang C, Zhang A (2007) Mining gene-sample-time microarray data: a coherent gene cluster discovery approach. Knowl Inf Syst 13(3):305–335
25. Lehmann EL, D'Abrera HJM (1998) Nonparametrics: statistical methods based on ranks. Prentice-Hall, rev. ed. Englewood Cliffs, NJ, pp 292–323
26. Liu J, Li Z, Hu X, Chen Y (2009) Biclustering of microarray data with MOSPO based on crowding distance. BMC Bioinform 10(S–4)
27. Liu J, Wang W (2003) Op-cluster: clustering by tendency in high dimensional space. IEEE Int Conf Data Min. ISBN 0-7695-1978-4, pp 187–194
28. Liu JW, Li ZJ, Liu FF, Chen YM (2008) Multi-objective particle swarm optimization biclustering of microarray data. In: IEEE international conference on bioinformatics and biomedicine(BIBM 2008). IEEE Computer Society, Washington, pp 363–366
29. Liu X, Wang L (2007) Computing the maximum similarity bi-clusters of gene expression data. Bioinformatics 23(1):50–56
30. Luan Y, Li H (2003) Clustering of time-course gene expression data using a mixed-effects model with b-splines. Bioinformatics 19:474–482
31. Madeira SaraC, Oliveira ArlindoL (2004) Biclustering algorithms for biological data analysis: A survey. IEEE/ACM Trans Comput Biol Bioinform (TCBB) 1(1):24–45
32. Madeira SC, Oliveira AL (2009) A polynomial time biclustering algorithm for finding approximate expression patterns in gene expression time series. Algorithms Mol Biol 4:8
33. Maulik U, Mukhopadhyay A, Bandyopadhyay S (2009) Combining pareto-optimal clusters using supervised learning for identifying co-expressed genes. BMC Bioinform 10:27
34. Mitra S, Banka H (2006) Multi-objective evolutionary biclustering of gene expression data. Pattern Recogn 39(12):2464–2477
35. Myers JL, Arnold DW (2003) Research design and statistical analysis
36. Okada Y, Okubo K, Horton P, Fujibuchi W (2007) Exhaustive search method of gene expression modules and its application to human tissue data. In: IAENG international journal of computer science, 34, pp 1–16

37. Peddada SD, Lobenhofer EK, Li L, Afshari CA, Weinberg CR, Umbach DM (2003) Gene selection and clustering for time-course and dose-response microarray experiments using order-restricted inference. Bioinformatics 19:834–841
38. Pontes B, Divina F, Giráldez R, Aguilar-Ruiz JS (2007) Virtual error: a new measure for evolutionary biclustering. In: Evolutionary computation, machine learning and data mining in bioinformatics. pp 217–226
39. Prelic A, Bleuler S, Zimmermann P, Buhlmann P, Gruissem W, Hennig L, Thiele L, Zitzler E (2006) A systematic comparison and evaluation of biclustering methods for gene expression data. Bioinformatics 22(9):1122–1129
40. Schliep A, Schonhuth A, Steinhoff C (2003) Using hidden markov models to analyze gene expression time course data. Bioinformatics 19:i255–i263
41. Son YS, Baek J (2008) A modified correlation coefficient based similarity measure for clustering time-course gene expression data. Pattern Recognit Lett 29(3):232–242
42. Tanay A, Sharan R, Shamir R (2002) Discovering statistically significant biclusters in gene expression data. Bioinformatics 18:S136–S144
43. Tavazoie S, Hughes JD, Campbell MJ, Cho RJ, Church GM (1999) Systematic determination of genetic network architecture. Nat Genet 22:281–285
44. Teng L, Chan L (2008) Discovering biclusters by iteratively sorting with weighted correlation coefficient in gene expression data. J Signal Process Syst 50(3):267–280
45. Wei JM, Wang SQ, Yuan XJ (2010) Ensemble rough hypercuboid approach for classifying cancers. IEEE Trans Knowl Data Eng 22(3):381–391
46. Yang J, Wang H, Wang W, Yu P (2003) Enhanced biclustering on expression data. In: BIBE '03: Proceedings of the 3rd IEEE symposium on bioInformatics and bioengineering. IEEE Computer Society, Washington, p 321
47. Zhang Z, Teo A, Ooi BC, Tan KL (2004) Mining deterministic biclusters in gene expression data. Bioinformatic and bioengineering, IEEE international symposium on, pp 283–290
48. Zhao H, Liew A, Xie X, Yan H (2008) A new geometric biclustering algorithm based on the hough transform for analysis of large scale microarray data. J Theoretical Biol 251:264–274

Author Biographies

Wassim Ayadi received an Undergraduate Degree in Computer Science applied to Management in 2004 from the Higher Institute of Management, Tunis, Tunisia, and a Master's Degree in Computer Science in 2007, from the Faculty of Sciences of Tunis, Tunisia. He is currently a PhD Student in the Faculty of Sciences of Tunis, and in the University of Angers, France, where he is working his PhD degree in the area of biclustering algorithms. He is Teaching Assistant in the Higher School of Economic and Commercial Sciences, Tunis. He is also a member of research Unit of Technologies of Information and Communication (UTIC), Higher School of Sciences and Technologies of Tunis, University of Tunis, and of Laboratoire d'Etude et de Recherche en Informatique d'Angers (LERIA), University of Angers. His research interests are Computational Molecular Biology, and Knowledge Discovery and Data Mining.

Mourad Elloumi is currently Associate Professor in Computer Science, Faculty of Economic Sciences and Management of Tunis, University of Tunis-El Manar, and member of the research Unit of Technologies of Information and Communication (UTIC), Higher School of Sciences and Technologies of Tunis, University of Tunis, Tunisia. Professor Elloumi is the author/co-author of more than 30 publications in international journals and conferences. He was the guest editor of a special issue on biological knowledge discovery and data mining, Knowledge Based Systems Journal (Elsevier 2002), the co-editor, with Prof. Albert Y. Zomaya of the University of Sydney (Australia), of a book entitled Algorithms in Computational Molecular Biology: Techniques, Approaches and Applications (Wiley, to appear in 2011) and the co-editor of the proceedings of two international conferences. His research interests are Computational Molecular Biology, Algorithmics and Knowledge Discovery and Data Mining.

Jin-Kao Hao holds the title of Distinguished Professor at the Computer Science Department of the University of Angers (France) and is currently the Director of the LERIA Laboratory. His research lies in the design of effective algorithms for solving large-scale combinatorial search problems. He is interested in various application areas including bioinformatics, telecommunication networks, and transportation. He has authored or co-authored more than 140 peer-reviewed publications and edited four books. He has served as an Invited Member of more than 130 Program Committees of the Conferences and is on the Editorial Board of four International Journals.