Journal of Mathematics and Informatics
Vol. 12, 2018, 63-77
ISSN: 2349-0632 (P), 2349-0640 (online)
Published 19 March 2018
www.researchmathsci.org
DOI: http://dx.doi.org/10.22457/jmi.v12a7

Journal of
Mathematics and Informatics

Optimization of Biclustering Algorithm Based on Greedy Randomized Adaptive Search Procedure

Chun-de Yang[1], ***Ling Zhao***[1], ***Kun-xian Shu***[2] **and** ***Ling Wang***[2]

[1]School of science, Chongqing University of Posts and Telecommunications
Chongqing – 400065, Chongqing, China. E-mail: yangcd@cqupt.edu.cn
[2]Chongqing Key Laboratory of Big Data for Bio Intelligence
Chongqing University of Posts and Telecommunications, Chongqing – 400065
Chongqing, China. E-mail: shukx@cqupt.edu.cn

Received 1 March 2018; accepted 18 March 2018

Abstract. The biclustering algorithm based on greedy randomized adaptive search procedure (GRASP) has two main steps, firstly, bicluster seeds are generated by K-means clustering; secondly, the greedy randomized adaptive search procedure is used to expand these bicluster seeds so that to get some better biclusters. However, K-means clustering assumes that each gene belongs to only one cluster, which is not biologically valid, because some genes may not belong to only one category, and K-means clustering requires multiple runs to determine the number of "clusters," all of which will affect the identification of bicluster seeds. This paper optimizes GRASP-based bIclustering algorithm. In this paper, the single column vector clustering is used to generate biclusters seed, and then the bicluster seeds are added more rows and columns to generate the final biclusters. Experiment shows that this algorithm can generate a large number of biclusters with coexpression level, and the algorithm also finds more biclusters.

Keywords: biclustering algorithm, greedy randomized adaptive search procedure, mean squared residuals

AMS Mathematics Subject Classification (2010): 90C72

1. Introduction

The rapid development of gene chip technology has produced a large amount of biological data. How to find useful information from these massive data has become a key and difficult point in the field of system biology. Gene expression data is usually stored in a matrix where each row represents the expression of a gene under all experimental conditions and each column of the matrix represents the expression of all genes under a certain experimental condition. Gene expression data can be used to determine which genes have similar expression patterns, which genes expression have changed, and how gene activities are affected under different conditions. Traditional clustering algorithm is clustering based on all attributes of data, and only to find non-overlapping clustering genes, and a large number of biological information is hidden in these local information, but traditional clustering can not find local information.

The biclustering algorithm proposed by Hartigan [1] is clustering based on the rows and columns of gene expression matrices simultaneously. The biclustering algorithm is a new clustering method, it is to find local similarity in gene expression matrix [2,3,4,5]. Cheng et al. [6] proposed the well-known CC algorithm, and the biclustering algorithm is applied to gene expression data for the first time, the CC algorithm can quickly get the user specified number of biclustering, But the flaw is obvious, substitution of random numbers will change the original data, which leads to inaccurate results and and Can not find overlapping biclustering. CC algorithm has been improved by Dharan et al. [7,8]. They proposed a biclustering algorithm based on greedy random adaptive search procedure (GRASP) to find better biclustering. The algorithm consists of two main steps: Firstly, high quality bicluster seeds are generated by K-means clustering, secondly, these seeds are grown by the greedy random adaptive search procedure. Although GRASP-based biclustering algorithm can find overlapping bicluster, however, the final result of biclusters and the number of biclusters are largely dependent on the bicluster seeds. These bicluster seeds are respectively clustered in the rows and columns of the gene expression matrix, and then they are combined to generate different bicluster seeds.

In this paper, a single column vector clustering method is used to produce high quality bicluster seeds. This method can generate more bicluster seeds, and use the greedy randomized adaptive search procedure to expand these seeds. In this paper, we use the mean square residue to evaluate the effect of the obtained biclusters, and compare it with the Dharan et al. [7,8] proposed Biclustering algorithm, and the validity of the algorithm is verified. The results show that the algorithm in this paper can produce better biclusters.

2. Biclustering algorithm

2.1. The definition of bicluster

Definition 1. Set A as the gene expression matrix, and the gene expression matrix is the expression value of a group of genes under some experimental conditions. $G=\{g_1, g_2, \cdots, g_n\}$ and $C=\{c_1, c_2, \cdots, c_m\}$ respectively represent gene set and conditional set. a_{ij} is an element in the gene expression matrix A, which indicates the expression value of the i th gene under the j th condition. Bicluster is a subset of genes that have some similar pattern of expression under certain experimental conditions or under all experimental conditions. The Bicluster is a submatrix of gene expression matrix, which can be expressed as A_{IJ}, where $I \subseteq G$ and $J \subseteq C$.

In order to measure the effect of bicluster, Cheng et al. [6] defined the average squared residue ($Hscore$) as a bicluster for quantitative analysis. The mean squared residue formula is as follows:

$$Hscore(I,J)=\frac{\sum_{i\in I, j\in J}(a_{ij}-a_{Ij}-a_{iJ}+a_{IJ})^2}{|I|\cdot|J|} \tag{1}$$

where $a_{ij} = \frac{\sum_{i \in I} a_{ij}}{|I|}$ is the average of the j th column, $a_{iJ} = \frac{\sum_{j \in J} a_{ij}}{|J|}$ is the average of the i th row, $a_{IJ} = \frac{\sum_{i \in I, j \in J} a_{ij}}{|I| \cdot |J|}$ is the average of A_{IJ}. If $\exists \delta > 0$, it satisfies $Hscore(I, J) \le \delta$, so that the bicluster A_{IJ} is defined as $\delta - bicluster$. Where δ is the maximum acceptable mean squared residue, the δ must be estimated in advance, and each data set has a different δ value.

2.2. Analysis of biclustering algorithm based on greedy randomized adaptive search procedure

Cheng et al. [6] have proved that the search for biclustering problem is NP-hard, and its operation time is increasing exponentially. The GRASP-based biclustering algorithm is a heuristic method [9,10,11], which requires much less computation than the exhaustive search method, and δ-*bicluster* can be found in a reasonable time . Finding a bicluster is an optimization problem [12,13], the purpose of which is to find a bicluster with the largest capacity but the $Hscore$ smaller one. The GRASP-based biclustering algorithm has two main steps. First of all, high quality bicluster seeds generated by K-means clustering; Secondly, more genes and experimental conditions are added to these seeds by the GRASP to expand the bicluster seeds until the average square residue ($Hscore$) of each bicluster reaches a predetermined threshold δ, and it is different for every dataset. Dharan et al. [7,8] have used K-means algorithm to generated bicluster seeds. during seed generation, the rows and columns of gene expression matrix are K-means clustered respectively. The gene expression matrix is divided into P gene clusters and Q condition clusters, if each gene cluster contains too many genes, each gene cluster is further divided into subclusters, they obtained X gene clusters and Q condition clusters. By combining these gene and condition clusters, $X \times Q$ disjoint submatrices is obtained, the $Hscore$ values of the entire $X \times Q$ submatrices are calculated, and the submatrices with $Hscore$ value less than the threshold are selected as the bicluster seeds. A bicluster seed is actually a small bicluster whose $Hscore$ value is less than the threshold δ, but genes and experimental conditions including in bicluster is not the largest.

During Seed growth, which includes construction phases and local search [14]. In the construction phase: firstly, a restricted candidate list is constructed, any one element in the restricted list is merged into the bicluster seed. If $Hscore$ of the bicluster seed is still smaller than the threshold δ, so the restricted candidate list is updated, repeat this step extended bicluster seeds. Local search starts from the Bicluster generated in the construction stage, where the local search finds a new bicluster in the field of the current bicluster and replaces the current bicluster when it has a lower $Hscore$. The algorithm pseudocode is shown in Table 1 [15,16].

Table 1: Algorithm of greedy randomized search procedure

```
Algorithm GRASP (Seed)
    $Solution=seed$ ;
    While <termination condition not met> do
    Solution ← Greedy_Randomized_ Construction (Current);
    Solution ← Local_Search (Solution);
    Current ← Solution;
End
```

The GRASP-based biclustering algorithm generates high quality bicluster seeds by the K-means clustering method, so the bicluster seeds play a decisive role in the final result of bicluster. It is well known that the K-means clustering algorithm is easy to operate, but it has its limitations. K-means Clustering assumes that each gene belongs to only one cluster, which is not true in biological sense, because some genes may not belong not to only one category, but also belong to several categories. The K-means clustering method specifies the number of "clusters" in advance, and each gene belongs to the best "cluster", but it is not always meaningful. And how to determine the number of "cluster", which will affect the determination of the bicluster seeds, and then affect the final bicluster result.

3. Optimization of selection method for bicluster seeds and expansion of bicluster seeds

3.1. Optimization of selection method for bicluster seeds

In this paper, the biclustering algorithm uses a single column vector clustering to generate the bicluster seeds; the bicluster seeds are extended through GRASP. The single column vector clustering is similar to individual dimension clustering (CLIC) [17] method. CLIC [17] is an effective clustering method that has been tested on larger microarray datasets. CLIC [17] clustering is more effective than traditional K-means clustering methods. The single column vector clustering is a class of genes with similar expressions under each condition, and more bicluster seeds can be found, each run found that the bicluster seeds are the same.

Firstly, a column vector is sorted according to the expression value from small to large, and the standard deviation sd_all of all gene expression values in the vector is calculated. In this paper, the same class of genes with the same clustering index value to replace its expression value. The index value of the gene in each cluster is $KI(KI \in Z^{+})$. The index value set of all clusters is K_{KI} , The number of genes accumulated in each cluster set is *count.* Initially K_{KI} is an empty collection, and $count = 0$. The gene i is placed in a cluster set and the standard deviation $sd(K_{KI,count})$ of the set is calculated, if $sd(K_{KI,count}) \le sd(K_{KI,count-1})$ and $sd(K_{KI,count}) \le sd\_all$ are satisfied, the gene i belongs to the set and replaces the expression value of the gene i with the clustering index KI , at the same time $count = count + 1$. On the contrary, if

$sd(K_{KI,count}) > sd(K_{KI,count-1})$ or $sd(K_{KI,count}) > sd\_all$ are satisfied, A new cluster is generated and its index value is $KI = KI + 1$, at the same time, the cluster index $KI = KI + 1$ is assigned to the gene i. Repeat this step until all the genes in the column vector are divided into a small cluster. Secondly, the above steps are repeated in m column vectors to obtain m column vectors with clustering index values, and the m column vectors are reordered according to the original gene order to obtain new m column vectors, $n \times m$ matrix that is recombined by the m cluster index vectors. Pseudocode for Single column vector clustering is shown in Table 2.

Table 2: The single column vector clustering

```
Input: $n \times m$ Gene Expression Matrix
Output: $n \times m$ Index matrix
The following processes are performed on $m$ column vectors respectively:
Step 1. The genes in each column are sorted from small to large, gene set
from $G = \{g_1, g_2, \cdots, g_n\}$ to $G' = \{g_1', g_2', \cdots, g_n'\}$.
Step 2. Initially, set the gene index $i = 1$ and set the cluster index from 1 to $KI$.
Step 3. Calculate the standard deviation of all gene expression values for one
condition, set it as $sd\_all$.
Step 4. When the clustered index is $KI$ and set $K_{KI} = NULL$. Let $K_{KI}$ be the gene
in the cluster.
    a. Set the number of genes accumulated in the cluster set, $count = 0$.
    b. $K_{KI} = K_{KI} \cup \{g_i'\}$
    c. The cluster index $KI$ is used to replace the expression value of the gene in
       the cluster set.
Step 5. If cluster $K_{KI} != NULL$, do:
    a. Set $count = count + 1$.
    b. Set $i = i + 1$.
    c. Set $K_{KI} = K_{KI} \cup \{g_i'\}$.
    d. When the number of genes in the cluster is $count$, the standard
       deviation $sd(K_{KI,count})$ is calculated.
    e. While $sd(K_{KI,count}) \le sd(K_{KI,count-1})$ and $sd(K_{KI,count}) \le sd\_all$, do:
        i. Set $i = i + 1$.
        ii. Set $K_{KI} = K_{KI} \cup \{g_i'\}$.
        iii. The cluster index $KI$ is used to replace the expression value of the
             gene in the cluster set.
    f. If $sd(K_{KI,count}) > sd(K_{KI,count-1})$ or $sd(K_{KI,count}) > sd\_all$, do.
        i. Set $KI = KI + 1$.
        ii. Set $K_{KI} = K_{KI} \cup \{g_i'\}$.
        iii. The cluster index $KI$ is used to replace the expression value of the
        gene in the cluster set.
        iv. Set $count = 0$.
Step 6. Repeat step 4 to step 5 until $i == n$.
Step 7. Each column of genes has a column index
vector $\{1,1,2,2,\cdots,KI-2,KI-1,KI\}$, The index vectors are arranged in the
order of the original gene $G=\{g_1,g_2,\cdots,g_n\}$.
Step 8. The $m$ column index vectors are recombined into a $n\times m$ matrix.
```

Bicluster seeds are determined from the $n\times m$ index matrix(as shown in Table 3). Genes of the bicluster seeds in each condition are labeled with the same cluster index. These genes in another condition can be labeled with either different or the same kinds of cluster indexes. If genes in the same conditions are labeled with the same cluster index, it means that these genes with the same cluster index have similar expression patterns under this condition; If genes in other conditions are labeled with the same kind of cluster index, it means that these genes expression patterns are similar in the original condition and other conditions. In other words, genes show similar expression patterns over these conditions. These bicluster seeds are more relevant than the bicluster seeds selected by K-means cluster, because genes with similar expression under each condition are subdivided into many clusters, the number of bicluster seeds is found sufficiently large and comprehensive. Moreover, the same bicluster seeds can be determined even if the algorithm is run multiple times.

Table 3: Determination of bicluster seeds

```
Determination of bicluster seeds
In the $n\times m$ index matrix executing:
Step 1. Initialization, let bicluster seed set $S = NULL$
Step 2.Under each condition, for $s=1,\cdots,KI$ , do:
    a. Under each condition when the clustering index $KI$ is $s$, let $g(K_s)$
       for rows of genes.
    b. Set the conditions for biclusters seed set $CS = NULL$.
    c. Under condition $j=1,\cdots,m$ , do:
      i. When the gene is a $g(K_s)$ row and the condition is the $j$th column,
         let $g(K_{s,j})$ for the set of genes.
      ii. If the gene has the same clustering index in $g(K_{s,j})$, the
         set $CS = CS \cup \{c_j\}$.
    iii. If the elements $CS > 2$ in the set $CS$ .
The bicluster seeds set consists of $g(K_s)$ and $CS$
```

3.2. Expansion of bicluster seeds based on greedy randomized adaptive search procedure

In this paper, a greedy randomized adaptive search procedure [18,19,20] is used to expand the bicluster seeds, which includes the construction phase [21,22] and the local search. First, the candidate set C_1 (G_1) is initialized, the candidate set C_1 (G_1)is composed of all elements (except the elements in the bicluster seeds), second, expand bicluster seeds. Each iteration according to the greedy function g to evaluate each candidate in the candidate set C_1 (G_1),which is used as the basis for entry into the restricted candidate list(RCL).A randomly selected element from the RCL is added to a bicluster seed to calculate the $Hscore$ of the bicluster seed. If the $Hscore$ is still below the threshold, then the element belongs to this bicluster seed, Update the candidate set C_1 (G_1)and reevaluate each element in the candidate set C_1 (G_1). Each iteration of the candidate set C_1 (G_1) elements and greedy function values should be updated promptly to reflect their changes, which is the embodiment of the self-adaption function in the algorithm. The greedy function [7,8] in this paper is:

$$g = \{s \in C_1(G_1) \mid Hscore(S \cup s) \le S_{\min} + \alpha(S_{\max} - S_{\min})\} \tag{2}$$

with $S_{\min} = \min\{Hscore(S \cup s) \mid \forall s \in C_1(G_1)\}$

$S_{\max} = \max\{Hscore(S \cup s) \mid \forall s \in C_1(G_1)\}$, C_1 is condition candidate set, G_1 is gene candidate set, α is the parameter to build RCL $(\alpha \in [0, \cdots, 1])$, RCL consists of greedy functions from the candidate set to choose some high quality elements. The size of the RCL is an important factor affecting GRASP performance, the choice of α affects the size of the RCL , and the quality of the elements in the RCL largely depends on the parameter α .

At present, many studies [23,24] have proved that fixed α values can not get high quality feasible solutions. On the contrary, when the parameter α changes within a certain range during the iteration, a higher quality solution [11,24] can be obtained, and the algorithm has higher flexibility. In this algorithm, adaptive method is used to adjust α . Different α values are used in different iterations, and a α value is selected from discrete set $R = \{\alpha_1, \alpha_2, \cdots, \alpha_m\}$ every iteration. The average square residue of biclustering obtained from previous iterations is used as a guide for α selection. p_i is a probability related to the selection of $\alpha_i, i \in (1, \cdots, m)$, which initially makes all p_i equal to $\frac{1}{m}$, and periodically reestimates the selection probability p_i according to the value of the feasible solution during the iteration. Let A_i be the $Hscore$ average of the biclusters obtained during the construction of phase $\alpha = \alpha_i$. The probability p_i is updated regularly according to the following formula [24]:

$$p_i = \frac{q_i}{\sum_{i=1}^{m} q_i} \tag{3}$$

where $q_i = \frac{1}{A_i}, (i = 1, \cdots, m)$. Therefore, in the next iteration, α is reselected according to the updated probability p_i in order to produce a better bicluster. The algorithm of this paper is the same as the method of selecting α on GRASP-based biclustering algorithm, where $\alpha \in [0.25, \cdots, 0.60]$.

Usually, the quality of the bicluster obtained during the construction phase is not high, so the local search can improve the result of the biclusters and obtain a higher quality of biclusters. The basic idea of local search is to iteratively search for the optimal solution in the neighborhood of the existing solution, the optimal solution can be used to replace the current solution. Let S be the current solution, let $S^{'}$ be the local solution to the adjacent solution, if $Hscore(S^{'}) < Hscore(S)$, then use the adjacent solution $S^{'}$ to replace the current solution S.

4. Simulation and analysis of experimental results

4.1. Dataset used

The biclustering algorithm in this paper is implemented in R software, and tested on Yeast Saccharomyces Cerevisiae cell cycle expression dataset. The data is stemmed from [25] and is based on Tavazoie et al. [26]. The data set contains 2884 genes and 17 experimental conditions (experimental conditions are time points), and there are 34 missing values in the data set, and -1 is a missing value. All expressions are integers in the range of 0 to 595. The value of δ is the maximum allowable dissimilarity between the genes and the conditions, while the higher δ indicates the decrease homogeneity. Therefore, the threshold $\delta = 200$ used in this paper is the same as the δ value in the GRASP-based biclustering algorithm.

4.2. Comparison of results between this algorithm and GRASP-based biclustering algorithm

In this paper, the method of seed selection based on GRASP-based biclustering algorithm [7,8] is optimized, that is, how to obtain high quality bicluster seeds. In this paper, we carry out a single column vector clustering for each condition. It clustered the genes with very similar expression level under each condition, and similar genes are given the same clustering index. Finally, the m column index vector is merged into the matrix of $n \times m$, and the bicluster seeds are determined according to the genes of the same or different index in the merging matrix. The algorithm in this paper and GRASP-based biclustering algorithm [7,8] are adding more genes and conditions to a group of bicluster seeds to expand these seeds ,until the $Hscore$ of the bicluster reaches a given threshold δ.

Assessing the quality of biclusters individuals usually depends on whether they are close to the four basic bicluster models [27]. Aguilar-Ruiz et al. [28,29] have demonstrated that the mean squared residuals can only find additive models and no other models can be found. The mean square residue $(Hscore)$ is a better way to measure whether the result is close to the addition model [30], and it is also the usual measure of measurement at the present stage. The size of a bicluster S (bicluster contains the number of elements) is also a criterion for judging the quality of bicluster. Therefore, there are many $Hscore$ evolution forms, such as $Hscore/S$ can effectively measure the quality of bicluster. The smaller the $Hscore/S$ value, the better the bicluster result. Multiple biclusters are found on the whole matrix, so it is also an important evaluation criterion to cover the comprehensiveness of the matrix elements and the overlap of the biclusters. It is found that overlapping biclusters should be considered in designing biclustering algorithms, it is also important that the bicluster results cover all the elements in the matrix, the biclustering algorithm in this paper can find overlapping biclusters and cover some elements in the matrix.

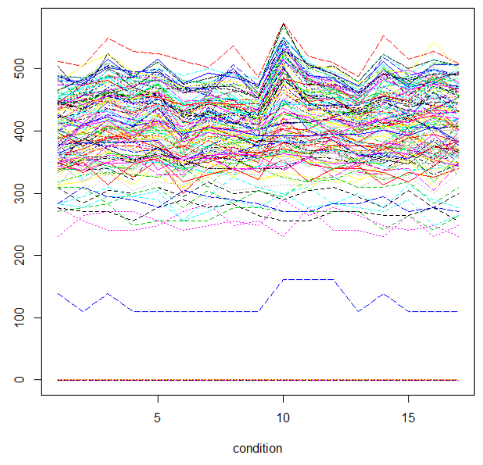
Table 4 summarizes the results of the algorithm in the yeast gene expression dataset. The algorithm in this paper can find 205 biclusters, which is far greater than the number of biclusters individuals found by the GRASP-based biclustering algorithm[7,8]. Therefore, the gene clusters with similar expression found by the algorithm in this paper is more comprehensive. In the biclusters discovered by the algorithm, on average, each bicluster contains 150.5317 genes, with an average of 16.67 experimental conditions per bicluster.

To verify the comprehensiveness of biclusters discovered by this algorithm, we calculate the coverage ratio of biclusters in the data matrix. If a gene or condition is contained in at least one bicluster, the gene or condition is covered by biclusters. Check the 2884×17 data matrix, the 205 biclusters found in this paper covers 34.77809% of the genes and all the conditions in the data set, however, the GRASP-based biclustering algorithm[7,8] not only found fewer biclusters, but also found that the biclusters only covered 9.604716% of the genes and 95.29412% of the condition in the data set. In this article, most of the conditions in the dataset are contained in at least one bicluster.

Table 4: Bicluster results comparison

method	The number of biclusters	Gene coverage ratio	Condition coverage ratio	The average size of the biclusters contains the conditions	The average size of the biclusters contains the gene
The method of this paper	205	0.3477809	1.0000000	16.67	150.5317
The GRASP -based biclustering method	5	0.09604716	0.9529412	16.2	162.4

As shown in the table 4, the GRASP-based biclustering algorithm[7,8] finds less number of biclusters, and genes contained in some biclusters found in this paper are slightly less than the genes contained in the biclusters found in the GRASP-based biclustering algorithm[7,8]. Although the algorithm of this paper found more biclusters, but these biclusters covered only one-third of the genes in the data matrix. Analysis of reasons: First, the algorithm starts with clustering all the genes under each condition, which itself is data coverage instead of partitioning. The discovered bicluster seeds is not duplicated. Second, through the growth of the bicluster seeds, we get the final biclusters. If a gene or condition may belong to several biclusters, almost every bicluster will have overlapping parts, so the coverage ratio of the whole dataset is low. In addition, we can only find the bicluster of the additive model, so this is one of the reasons why the gene coverage ratio is low. Two biclusters in this paper will be depicted in Figure 1.



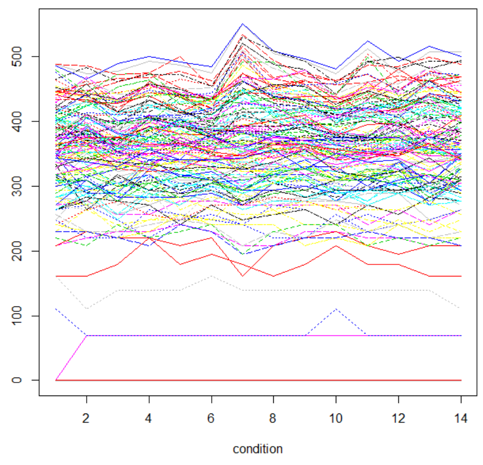
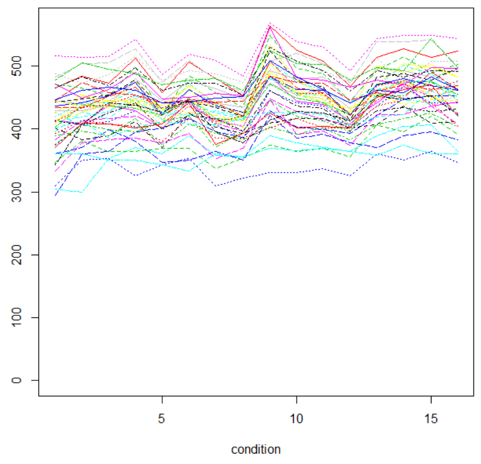
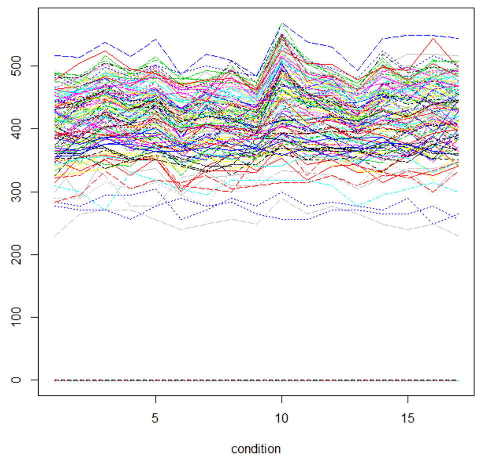
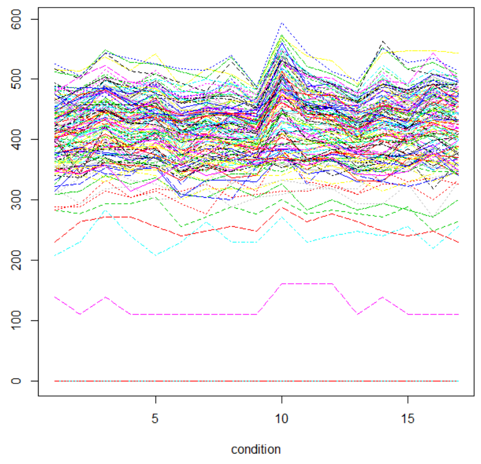


Figure 1: The 205th and 110th seed grow into biclusters

Figure 1 is two biclusters, they are grow from 205th seed and 110th seed. They contain 134 genes, 17 conditions and 149 genes 14 conditions respectively. Figure 2 shows several other biclusters generated by this algorithm, from these figures can be seen that the results of the bicluster is satisfactory, it may find some coexpression of the biclusters; At the same time, the algorithm uses a greedy randomized adaptive search procedure, can quickly find some smaller bicluster, that is, a better coexpression level of bicluster. Moreover, the biclusters found by this algorithm not only covers all the conditions, but also some biclusters covers fewer conditions, that is, the bicluster individuals with higher levels of coexpression can be found.







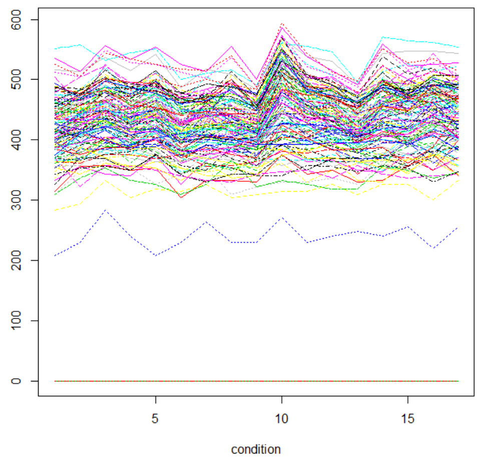


Figure 2: The 204th, 14th, 42th, and 194th seed generated biclusters

4.3. GO function enrichment analysis

GO enrichment analysis[31] can investigate the biological relevance of the acquired biclusters. GO function analysis is used to enrich the results of biclusters. This paper uses SGD GO Termfinder [32] for functional enrichment analysis. The bicluster results were analyzed by function enrichment, which included three functions: GO biological processes (GO BP), GO molecular functions (GO MF), GO cellular components (GO CC). By analyzing the association between genes and functions in the biclusters, false positives (FDR) are used to multiple tests to correct the estimated p-value to avoid the more the number of genes contained in the bicluster, the greater p-value leads to its feature richness. The results of GO analysis showed that the FDR of bicluster was zero in biological process analysis, molecular function analysis and cell composition analysis, which indicated that the biclusters found in this paper had almost no false discovery rate. In addition, the p-values obtained by the bicluster GO analysis found in this paper are very small, which shows that the proposed algorithm has high accuracy. The biclusters found in this paper are more abundant in functional terms than biclusters found in GO BP, GO MF, GO CC and GRASP-based biclustering algorithms [7,8].

5. Conclusion

In this paper, the bicluster seeds are crucial to the bicluster results. So it is very important to find suitable and comprehensive bicluster seeds. A single column vector clustering is used to obtain bicluster seeds to solve the problem of finding biclusters. This method is more stable than K-means clustering and does not require multiple runs to determine the bicluster seeds. The experimental results also show that the proposed algorithm is superior to the GRASP-based biclustering algorithm.

This article focuses on the bicluster expression level. In this paper, a single column vector clustering method is used to generate the bicluster seeds, and the bicluster seeds are added with rows and columns to generate the final biclusters. With the discovery of the biclusters drew mapping, the biclusters result are compared with the biclusters result of Dharan et al. [7,8]. Experimental result shows that the algorithm finds more biclusters, and computing time is also acceptable. However, it also explains a problem, although the algorithm in this paper finds more biclusters, it means that some genes and conditions in these biclusters are highly overlapped. At the same time, we can only find the additive model bicluster. In future studies, further consideration may be given to discovering the bicluster of other models so that a more efficient biclustering algorithm can be obtained.

REFERENCES

1. J.A.Hartigan, Direct clustering of a data matrix, *Journal of the American Statistical Association*, 67 (1972) 123-129.
2. M.Charrad and M.B.Ahmed, Simultaneous clustering: a survey, *International Conference on Pattern Recognition and Machine Intelligence*, 6744 (2011) 370-375.
3. H.A.Majd, A.R.Baghestani, S.M.Tabatabaei, S.Shahsavari, M.R.Tavirani and M.Hamidpour, Biclustering algorithm in gene expression data: A review article, 12 (2015) 3864-3871.
4. B.Pontes, R.Giráldez and J.S.Aguilar-Ruiz, Biclustering on expression data: A review, *Journal of Biomedical Informatics*, 57 (2015) 163-180.
5. S.Roy, D.K.Bhattacharyya and J.K.Kalita, Analysis of gene expression patterns using biclustering, *Humana Press*, (2015) 91-103.
6. Y.Cheng and G.M.Church, Biclustering of expression data, *Proceedings International Conference on Intelligent Systems for Molecular Biology*, 8 (2000) 93-103.
7. S.Dharan and A.S.Nair, Biclustering of gene expression data using greedy randomized adaptive search procedure, TENCON 2008 - 2008 IEEE Region 10 Conference, 10 (2008) 1-5.
8. S.Dharan and A.S.Nair, Biclustering of gene expression data using reactive greedy randomized adaptive search procedure, *BMC Bioinformatics*, 10 (2009) 1-10.
9. A.Duarte, J.Sánchez-Oro, M.G.C.Resende, F.Glover and R.Martí, Greedy randomized adaptive search procedure with exterior path relinking for differential dispersion minimization, *Information Sciences*, 296 (2015) 46-60.

10. R.Figueiredo, P.G.Silva and M.Poss, Transmission expansion planning with re-design - a greedy randomized adaptive search procedure, *Energy Systems,* 1 (2013) 113-139.
11. M.Prais and C.C.Ribeiro, Reactive GRASP: an application to a matrix decomposition problem in TDMA traffic Assignment, *Informs Journal on Computing*, 12 (2000) 164-176.
12. A.Layeb, A new greedy randomized adaptive search procedure for solving the maximum satisfiability, *International Journal of Operational Research*, 17 (2013) 1-17.
13. A.Layeb, A.Boudra, W.Korichi and S.Chikhi, A new greedy randomized adaptive search procedure for multiobjective RNA structural alignment, *International Journal in Foundations of Computer Science & Technolog,* 3 (2013) 1-14.
14. S.Mancini, A greedy randomized adaptive search for the surveillance patrol vehicle routing problem, *Recent Developments in Metaheuristics*, 11 (2018) 305-317.
15. C.Faycal, M.E. Riffi and B. Ahiod, Hybrid genetic algorithm and greedy randomized adaptive search procedure for solving a nurse scheduling problem, *American Journal of Cardiology*, 104 (2015) 55D–56D.
16. S.Das and S.M.Idicula. Application of greedy randomized adaptive search procedure to the biclustering of gene expression data, *International Journal of Computer Applications*, 2 (2011) 6-13.
17. T.Yun, T.Hwang, K.Cha and G.S.Yi, CLIC: clustering analysis of large microarray datasets with individual dimension-based clustering, *Nucleic Acids Research,* 38 (2010) W246.
18. H.Akrout, B.Jarboui, A.Rebaï and P.Siarry, New greedy randomized adaptive search procedure based on differential evolution algorithm for solving no-wait flowshop scheduling problem, *International Conference on Advanced Logistics and Transport,* (2013) 327-334.
19. H.Lei and K.Qin, Greedy randomized adaptive search procedure for analog test point selection, *Analog Integrated Circuits & Signal Processing*, 79 (2014) 371-83.
20. V.C.Yepes, C.Torres-Machi, A.Chamorro and E.Pellicer, Optimal pavement maintenance programs based on a hybrid greedy randomized adaptive search procedure algorithm, *Statyba*, 22 (2016) 540-550.
21. A.H.Yin, T.Q.Zhou, J.W.Ding, Q.J.Zhao and Z.P.Lv, Greedy randomized adaptive search procedure with path-relinking for the vertexp-center problem, *Journal of Computer Science and Technology,* 32 (2017) 1319-1334.
22. H.Mokhtari and A.Salmasnia, Fitting the three-parameter weibull distribution by using greedy randomized adaptive search procedure, *International Journal of Engineering*, 30 (2017) 425-432.
23. T.Gevezes and L.Pitsoulis, A greedy randomized adaptive search procedure with path relinking for the shortest superstring problem, *Journal of Combinatorial Optimization*, 29 (2015) 859-883.
24. M.G.C.Resende, Greedy randomized adaptive search procedures (GRASP), *Journal of Global Optimization*, 6 (1999) 109-133.
25. Yeast Saccharomyces cerevisiae cell cycle expression dataset, http://arep.med.harvard.edu/biclustering.
26. S.Tavazoie, J.D.Hughes, M.J.Campbell, R.J.Cho and G.M.Church, Systematic

determination of genetic network architecture, *Nature Genetic*, 22 (1999) 281-285.

27. D.Yan and J.Wang, Biclustering of gene expression data based on related genes and conditions extraction, *Pattern Recognition*, 46 (2013) 1170-1182.
28. J.S.Aguilarruiz, Shifting and scaling patterns from gene expression data, *Bioinformatics*, 21 (2005) 3840-3845.
29. D.Bozdağ, A.S.Kumar and U.V.Catalyurek, Comparative analysis of biclustering algorithms, *In Proceedings of the First ACM International Conference on Bioinformatics and Computational Biology,* 8 (2010) 265-274.
30. B.Pontes, R.Girldez and J.S.Aguilarruiz, Quality measures for gene expression biclusters, *Plos One*, 10 (2015) e0115497.
31. J.A.Nepomuceno, A.Troncoso and I.A.Nepomuceno-Chamorro, Integrating biological knowledge based on functional annotations for biclustering of gene expression data, *Computer Methods & Programs in Biomedicine*, 119 (2015) 163-180.
32. Stanford University, Saccharomyces Genome Database. http://db.yeastgenome.org/cgi-bin/GO/goTermFinder.