An EA Framework for Biclustering of Gene Expression Data

Stefan Bleuler, Amela Prelić, and Eckart Zitzler
Computer Engineering and Networks Laboratory (TIK)
Swiss Federal Institute of Technology (ETH), Zürich
Email: {bleuler, aprelic, zitzler}@tik.ee.ethz.ch

Abstract— **In recent years, several biclustering methods have been suggested to identify local patterns in gene expression data. Most of these algorithms represent greedy strategies that are heuristic in nature: an approximate solution is found within reasonable time bounds. The quality of a biclustering, though, is often considered more important than the computation time required to generate it. Therefore, this paper addresses the question whether additional run-time resources can be exploited in order to improve the outcome of the aforementioned greedy algorithms. To this end, we propose a general framework that embeds such biclustering methods as local search procedures in an evolutionary algorithm. We demonstrate on one prominent example that this approach achieves significant improvements in the quality of the biclusters when compared to the application of the greedy strategy alone.**

I. Introduction

High-throughput technology has enabled molecular biologists to study genes and gene products of living organisms on a system level. Nowadays, it is possible to measure the expression levels of thousands of genes in a single experiment [1] and to determine protein interactions [2] and localizations [3] on a proteome-wide scale, to name only a few examples. The hope is that these experimental methods will allow us to gain a better understanding of biological systems, particularly in terms of structure and dynamics of the underlying gene and protein networks [4]. However, the vast amount of data produced by these methods render computational techniques and tools mandatory in order to analyze the experimental results and to extract useful information from it.

With respect to gene expression data, clustering represents the most commonly used analysis technique; the main goal is to identify genes with shared functions or shared regulatory mechanisms. Standard clustering methods such as k-means, hierarchical clustering [5], self organizing maps [6], partition the set of genes into disjoint groups according to the similarity of their expression patterns over *all* conditions. Thereby, they may fail to uncover processes that are active only over some but not all conditions. In contrast, *biclustering* aims at finding subsets of genes which are similarly expressed over a *subset* of conditions (see Figure 1), which better reflects biological reality. The usefulness of this concept in the context of microarray measurements has been demonstrated in different studies ([7]–[9]).

Among the various biclustering methods proposed in the literature, most approaches are based on greedy heuristics that iteratively refine a set of biclusters. These algorithms can be considered as local search methods which are fast but often yield suboptimal results. One has to take into account, though, that the time to compute a certain biclustering is often less critical than the quality of the outcome: In comparison to the amount of work required to perform the measurements, run-times of several minutes up to a couple of hours may be still acceptable if it can be justified by a substantial improvement in quality. Therefore, the question arises whether such an improvement is possible by integrating these greedy biclustering methods into a global search strategy, e.g., an evolutionary algorithm (EA). To this end, we will

- propose a general EA biclustering framework that can be coupled with several existing biclustering methods,
- present an implementation for one prominent biclustering method introduced by Cheng and Church [7], and
- apply and investigate the EA hybrid on data sets for yeast and *Arabidopsis thaliana*.

As we will show in the paper, the advantage of this approach is that the user can decide how much computation time he is willing to spend in order to improve the biclustering outcome; most of the current biclustering algorithms do not provide this flexibility. Furthermore, note that EAs and other blackbox optimization methods have been used for standard clustering, e.g., ([10], [11]), but to our best knowledge not in the context of biclustering.

The remainder of the paper is organized as follows. Section II formalizes the biclustering problem and briefly reviews existing biclustering algorithms in the context of gene expression data. Section III introduces the EA framework, discusses various implementation aspects, and describes the particular greedy biclustering techniques used in this study. Different EA implementations will be compared with the original method in Section IV, and the last section is devoted to concluding remarks.

II. Biclustering

Colloquially speaking, genes are the blueprints for proteins and some additional products, and mRNA is the first intermediate during the production of any genetically encoded molecule. The concentration of a specific mRNA molecule is usually called the *expression level* of the respective gene, and it serves as an indicator of the amount of end product that is currently being produced. Nowadays, the expression levels of

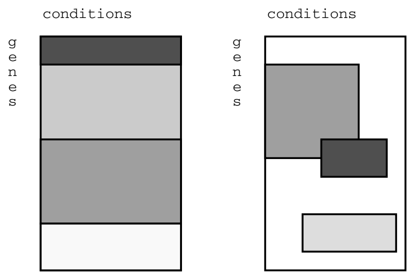

Fig. 1. Left: Traditional clustering searches a partition of all genes into k disjoint groups. Right: Biclustering searches for one or a set of blocks containing a consistent local pattern. Three biclusters are shown. (Note that it is not generally possible to display several biclusters at the same time as contiguous blocks.)

thousands of genes, possibly all genes in an organism, can be measured simultaneously in a single experiment using microarrays. Since most cellular processes are regulated by changes in gene expression, the result of a microarray measurement can be considered as an indicator of the current state of the cell. By performing measurements under different conditions and treatments, different cell states can be compared. A condition could for example consist of knocking out a specific gene or exposing the organism to a chemical. The outcome of such a series of microarray measurements is usually summarized in terms of an $m \times n$-matrix, E, where m is the number of considered genes and n the number of experimental conditions. A cell e_{ij} of E contains a real value that reflects the abundance of mRNA for gene i under condition j relative to a control experiment (absolute mRNA concentrations are seldom used).

Biclustering goes back to the work of Hartigan [12] and tries to identify local subpatterns in an arbitrary data matrix. In this context, we try to identify local patterns of gene expressions, i.e., subsets of genes that behave similarly over a subset of conditions. Accordingly, a bicluster is defined as a pair (G, C) where $G \subseteq \{1, \ldots, m\}$ is a subset of genes and $C \subseteq \{1, \ldots, n\}$ a subset of conditions; it can be easily shown that the space X of all biclusters given a matrix E is exponential in both m and n, where $X = 2^{\{1,\ldots,m\}} \times 2^{\{1,\ldots,n\}}$. Roughly speaking, the optimization goal is to find one or several biclusters that are optimal with respect to their homogeneity and their size. These two objectives are usually competing, and the various biclustering approaches in the literature differ in the way how the conflicting goals are formulated and combined.

In general, one can distinguish between top-down and bottom-up biclustering approaches. Techniques of the first category, e.g., ([12], [13]), start with the entire gene expression matrix and iteratively partition it into smaller submatrices. Other methods operate in a bottom-up fashion, i.e., they start with a randomly or deterministically chosen set of biclusters that are iteratively modified, usually enlarged, until no local improvement is possible anymore (e.g., [8], [14], [15]). In the following, we will focus on the biclustering method introduced by Cheng and Church [7], which was among the first to be applied to gene expression data; the problem formulation used in the corresponding study will be briefly summarized here.

The optimization task in [7] is to find the largest bicluster that does not exceed a certain homogeneity constraint. The size $f(G, C)$ is simply defined as the number of cells in E covered by a bicluster (G, C), while the homogeneity $g(G, C)$ is given by the *mean squared residue score*. Formally, the problem can be stated as follows:

$$\begin{array}{lrcl} \max & f(G,C) & = & |G| \cdot |C| \\ \text{subject to} & g(G,C) & \leq & \delta \\ & (G,C) & \in & X \end{array}$$

where

$$g(G,C) = \frac{1}{|G||C|} \sum_{i \in G, j \in C} (e_{ij} - e_{iC} - e_{Gj} + e_{GC})^2$$

is called the mean squared residue score,

$$e_{iC} = \frac{1}{|C|} \sum_{j \in C} e_{ij}, \; e_{Gj} = \frac{1}{|G|} \sum_{i \in G} e_{ij}$$

are the mean column and row expression values for (G, C) and

$$e_{GC} = \frac{1}{|G||C|} \sum_{i \in G, j \in C} e_{ij}$$

is the mean expression value over all cells contained in the bicluster (G, C). The threshold δ needs to be set by the user and defines the maximum allowable dissimilarity within the cells of a bicluster. Roughly speaking the residue expresses how well the value of an element in the bicluster is determined by the column and row it lies in. A set of genes whose expression levels change in accordance to each other over a set of conditions can thus form a perfect bicluster even if the actual values lie far apart.

Based on this problem formulation, Cheng and Church proposed a greedy search heuristic that generates a suboptimal bicluster while meeting the homogeneity constraint. In order to find several biclusters, an iterative application of this algorithm was suggested where after each step the cells of the newly found bicluster are assigned random numbers; thereby, previously found biclusters are deleted from the matrix and will not be found again. Details of the algorithm will be given later. In the following, we will discuss how this greedy algorithm can be integrated as local search method into an evolutionary algorithm.

III. EA Framework

A. The Evolutionary Algorithm

The main idea is to use the evolutionary algorithm to explore the search space X. In the following we will describe the architecture and implementation of a general EA for biclustering.

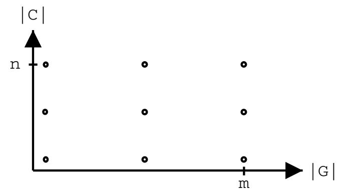

Fig. 2. Schematic drawing of the distribution of an initial population of nine biclusters. m and n are the total number of genes and conditions, respectively. $|G|$ and $|C|$ are the numbers of genes and conditions included in the bicluster.

1) Basic Implementation: Each individual represents one bicluster. Two alternatives for the encoding have been considered. One possibility is to use two variable length lists, one for the genes and one for the conditions included in the bicluster. An alternative solution uses bit representation with one bit string for the genes and one for the conditions. Since lists are more complex to handle than bit strings especially for large data matrices we have chosen to use a binary encoding. A bit is set to one when the corresponding gene or condition is contained in the bicluster. Biclusters containing only one gene or one condition are not interesting and are repaired whenever they appear by randomly adding a second gene or condition.

The initial population should be generated such that a high diversity of bicluster is attained. A simple strategy for example which sets each bit to 1 with a probability of 0.5 produces a set of biclusters containing different genes and conditions but all biclusters will have similar sizes. The proposed procedure deterministically chooses the number of genes and conditions to include in each bicluster such that the biclusters are uniformly distributed in the plane spanned by the number of genes and the number of conditions contained in a bicluster (see Figure 2). Which of the genes or conditions are included is then randomly chosen. This strategy assures that the full matrix is always part of the initial population.

Different variation operators are applicable. After some preliminary tests we decided to apply independent bit mutation to both strings with mutation rates chosen such that the expected number of bits flipped is the same for both strings. Uniform crossover is preferred to one point crossover because one point crossover would prohibit certain combinations of bits to be crossed over together.

If the EA operates alone without any local search method the following fitness function is used:

$$F(G,C) = \begin{cases} \frac{1}{f(G,C)} & \text{if } g(G,C) \leq \delta \\ \frac{g(G,C)}{\delta} & \text{else} \end{cases}$$

Note that fitness is to be minimized here. This function assigns a fitness smaller than 1 to all individuals which fulfill the residue constraint while those violating it are assigned a fitness greater than 1. In the case where the local search procedure is used the fitness function reduces to

$$F(G,C) = \frac{1}{f(G,C)}$$

since the local search method guarantees that the residue constraint is met.

The following selection operators have been chosen: For the environmental selection a (N,N) strategy was used where the offspring population replaces the parent population. A tournament selection is used as mating selection.

2) Diversity Maintenance: As a whole population of biclusters is evolved simultaneously it is possible not only to optimize one bicluster but also to find a set of biclusters which fulfill a desired property like total coverage or maximum overlap. To this end special diversity maintenance mechanisms need to be applied. In the following we describe one such mechanism which is implemented as a particular kind of environmental selection which is later compared to the simple (N,N) strategy. The general idea of the algorithm is as follows (see Algorithm 1 for details). First the biggest bicluster is selected and the elements which are contained in this bicluster are marked. In each following step the algorithm selects the bicluster which contains the largest number of unmarked cells. When several biclusters contain the same number of unmarked cells the algorithm selects the bicluster which contains the largest number of elements which are only marked once. This comparison continues for higher levels until a difference is found. These steps are iterated until enough individuals have been selected.

Algorithm 1 Environmental selection for diversity maintenance

```
$N$: number of individuals to select
$taken[]$: matrix of the same dimensions as input matrix
set all values in $taken[]$ to zero.
for $s = 1$ to $N$ do
  for all individuals in population do
    allocate a vector $levels$ and set all values to 0
    go through $taken$ and increment $levels[k]$ whenever
      $taken[i][j]$ == $k$ and $e_{ij}$ is part of the bicluster.
  end for
  get first individual $ind$ in population which was not yet
  selected
  $best$ = $ind$
  for all other individuals $ind$ in population do
    if $ind$ not yet selected then
      starting at $r = 0$ find first level where $levels[r]$ are
      not equal for $ind$ and $best$
      if $levels[r]$ is higher for $ind$ than for $best$ then
        $best$ = $ind$
      end if
    end if
  end for
  select $best$ and increment $taken[i][j]$ whenever $e_{ij}$ is part
  of the bicluster.
end for
```

B. Local Search Algorithm

Since the algorithm proposed in [7] is a greedy strategy it may happen that it takes decisions which are negative for the outcome, e.g., by removing a gene from the bicluster based on dissimilar values in some conditions which are later anyway removed from the bicluster. In such cases it could be beneficial to start with a specifically chosen submatrix instead of the whole matrix. The local search strategy is included in the EA with this effect in mind.

When using a local search procedure two options exist after the local search has been performed and its solution is returned: Either this solution is only used to determine the objective values of the original individual while the original individual remains unchanged or the new solution replaces the original individual. Some results concerning this question will be presented in Section IV.

1) Basic Cheng and Church Algorithm: This section describes the biclustering algorithm which we incorporated into the described framework. For a more in-depth description please be referred to the original paper [7]. The problem formulation has already been introduced in the previous section. The algorithm starts with the full matrix and consists of three steps:

1) In the first step multiple nodes (genes or conditions) are removed in each iteration. This step is only performed if the number of genes or conditions in the bicluster is above a certain threshold (default = 100). It works as follows: First calculate e_{iC} for all $i \in G$, e_{Gj} for all $j \in C$, e_{GC} and $g(G, C)$. If $g(G, C) \leq \delta$ return (G, C). Then remove all genes $i \in G$ with

$$\frac{1}{|C|}\sum_{j \in C}(e_{ij} - e_{iC} - e_{Gj} + e_{GC})^2 > \alpha g(G, C).$$

Recalculate all means and perform the corresponding operation on the conditions. Iterate until no improvement is possible any more.

2) The second step performs single node deletion. In each iteration the one node is removed with the largest

$$d(i) = \frac{1}{|C|}\sum_{j \in C}(e_{ij} - e_{iC} - e_{Gj} + e_{GC})^2.$$

The equation for the conditions is analogous. This step is iterated until the mean squared residue drops below δ.

3) In the last step all genes and conditions which are not contained in the bicluster are tested and whenever one can be added without increasing the mean squared residue it is added. This is iterated until no node can be added anymore.

Note that there is a small difference to the implementation in [7]. In step three the original algorithm tries to add each row or it's inverse where each element is multiplied with -1 while we do not consider the inverse here.

Beside δ there is one additional parameter to set for this algorithm: α determines how often multiple node deletion is used. A higher α leads to less multiple node deletion and thus in general requires more CPU time. Ideally the size of the resulting bicluster should increase with increasing α since the single node deletion is more accurate. In some tests, however, the size varied a lot and sometimes it decreased significantly with increasing alpha. For the present study we have thus chosen to stick to the value 1.2 which was used in [7].

2) Randomized Version: As we will show in Section IV the EA combined with the local search performs better than the Cheng and Church algorithm itself. Motivated by the question whether it is possible to improve the Cheng and Church algorithm itself by adding some flexibility we have implemented a randomized version of the Cheng and Church algorithm. It performs the deletion and addition operations in step 1 and step 3 only with a certain probability p. In step 2 it does not always remove the best node (row or column) but removes the best one with probability p, the second best with probability $(1-p)p$ and the k-th best node with probability $(1-p)^{k-1}p$.

All the algorithms for the EA as well as for the local search procedure have been implemented in C++.

IV. Simulation Results

In the simulation runs the following two questions were investigated both for the task of finding one bicluster as well as for finding a set of biclusters: How does the EA compare to the Cheng and Church algorithm? Are there any synergy effects when combining the two?

A. Data Preparation

In this study two different data sets have been used:

- The yeast expression data set from Cho et al. [16] that was used in [7].
- A set of expression values from *Arabidopsis thaliana*, a small plant. This data set was compiled from several biological studies ([17]–[19]).

The yeast data set contains 2884 genes and 17 conditions and the expression values denote relative abundance. The data have been used directly in the preprocessed form as provided by the authors of [7]. All values are integers in the range between 0 and 600. Following the procedure in [7] missing values are replaced by sampling a random number from a uniform distribution between 0 and 800.

The Arabidopsis data set contains 153 conditions and 1000 selected genes. The data contain absolute values which have been acquired using Affymetrix GeneChips and thus no missing values need to be handled. The data have been transformed in a similar fashion to the yeast data $x \rightarrow 100log(x+1)$ which results in real values in the range between 0 and 1165.

B. Finding One Bicluster

The application of biclustering algorithms might focus on finding a single optimal bicluster or on finding a set of biclusters with certain properties such as maximal overlap, total coverage, etc.. In this section we present the results setting the focus on finding a single bicluster while in the next section the focus is on sets of biclusters.

The standard parameter settings used in the following simulations are described in Table I. For the parameters of the Cheng and Church algorithm whenever possible the same

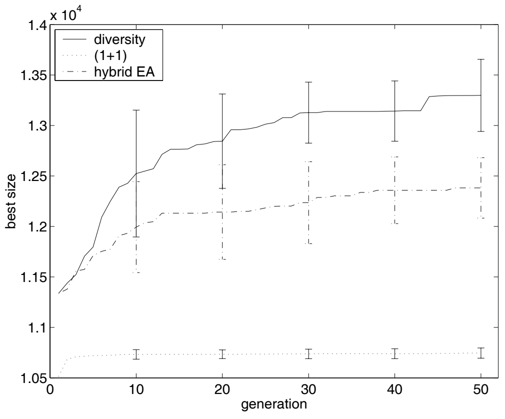

Fig. 3. Size of the biggest bicluster found up to the current generation for the yeast data set. For the (1+1) strategy one generation is equivalent to 100 function evaluations. (Mean over 10 runs with different random number generator seeds. The error bars have a total length of twice the standard deviation.)

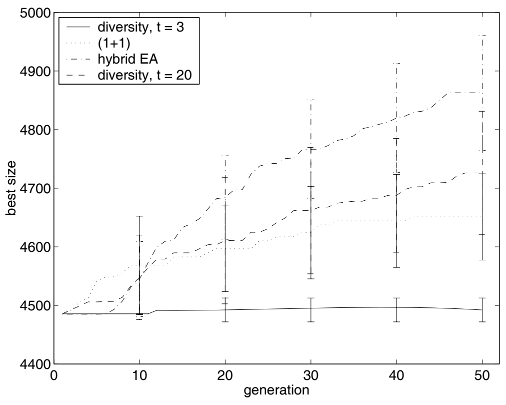

Fig. 4. Size of the biggest bicluster found up to the current generation for the Arabidopsis data set. For the (1+1) strategy one generation is equivalent to 100 function evaluations.(Mean over 10 runs with different random number generator seeds. The error bars have a total length of twice the standard deviation.)

values as in [7] have been chosen, i.e., δ, α and the number of clusters (= population size). The crossover rate refers to the percentage of parents involved in crossover. The mutation rate is the probability for bit flips in the independent bit mutation. Crossover and mutation rates were set to the values which yielded the best results in a few preliminary runs.

TABLE I

DEFAULT PARAMETER SETTINGS FOR THIS STUDY.

δ	300
α	1.2
mutation rate (relative to string length l)	$3/l$
crossover rate	0.5
tournament size	3
population size	100
number of generations	50

1) Comparison of Basic Algorithms: In order to determine whether an evolutionary algorithm without any local search procedure is able to find good biclusters the fitness function described in Section III was used. The application of this simple evolutionary algorithm results in very small biclusters compared to the solution generated by the Cheng and Church algorithm. While the latter one finds a bicluster of size 10523 in the yeast data matrix the maximal bicluster found by the EA is between 342 and 4036 for 10 runs. This discrepancy is even bigger on the Arabidopsis data where the pure EA sometimes fails to find a bicluster which meets the residue constraint.

The hybrid EA integrates the Cheng and Church algorithm as local search procedure as described above and updates each individual using the corresponding bicluster found by the local search (see Section IV-B.2 for a discussion of this approach). As shown in Figures 3 and 4 (the additional curves in the plots will be explained later) this method significantly improves the size of the biggest bicluster compared to the solution found by the Cheng and Church algorithm alone which measure 4485 elements for yeast and 10523 elements for Arabidopsis. Note that in the case of the yeast data set a larger bicluster is already found in the initial population.

In a next step we investigated whether maintaining a population of potential solutions is necessary. To this end the hybrid EA was compared to the corresponding (1+1) strategy where in each generation one new individual is generated from one parent and the better of the two is designated as parent for the next generation. As depicted in Figures 3 and 4 the (1+1) strategy improves the size of the bicluster substantially in some cases and fails to do so in others. Maintaining a population of 100 individuals produces better results on both data sets.

Since the Cheng and Church algorithm is deterministic it always yields the same result when applied to the same data. As shown this result is often not optimal. This raises the question whether it is possible to improve the performance of the Cheng and Church algorithm by making it more flexible. This is attained by randomizing each step of the algorithm as described in Section III.

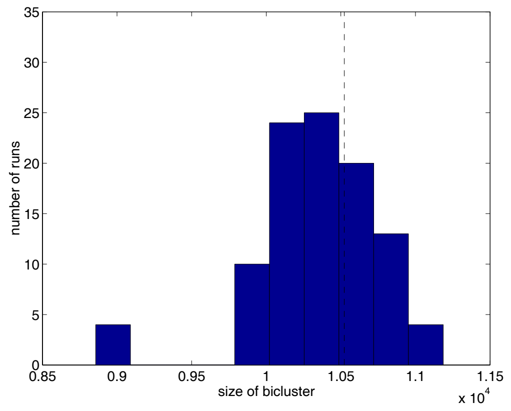

Fig. 5. Randomized Cheng and Church algorithm: Histogram of 100 runs on the yeast data set. Dashed line indicates the size of the bicluster found by the original algorithm.

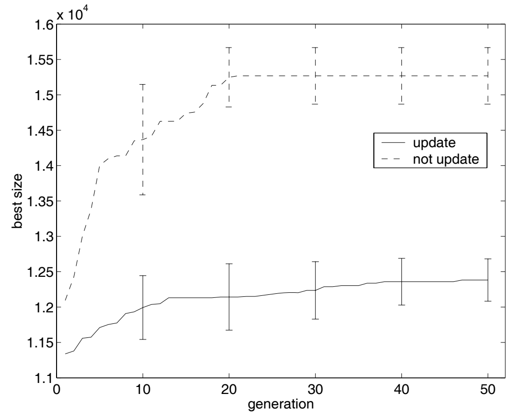

Fig. 6. Comparison of Lamarckian and Baldwinian evolution: Size of the biggest bicluster found up to the current generation for the yeast data set. (Mean over 10 runs with different random number generator seeds. The error bars have a total length of twice the standard deviation.)

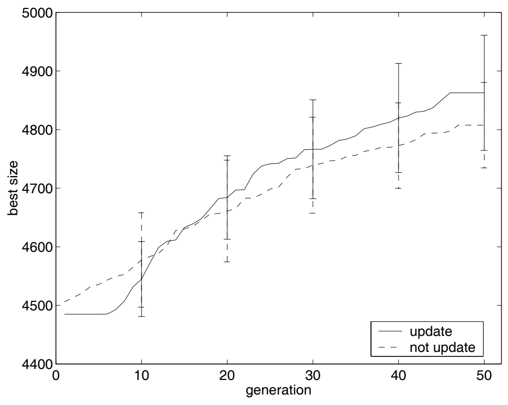

Fig. 7. Comparison of Lamarckian and Baldwinian evolution: Size of the biggest bicluster found up to the current generation for the Arabidopsis data set. (Mean over 10 runs with different random number generator seeds. The error bars have a total length of twice the standard deviation.)

Figure 5 shows the histogram of 100 runs for the yeast data using a probability of $p = 0.7$. Some improvement is possible although not as high as with the hybrid EA. Similar results are obtained for the Arabidopsis data. Whether the hybrid EA could be further enhanced by incorporating the randomized version of the local search is subject to ongoing research although some preliminary investigations indicate that this is not necessarily the case.

2) Baldwinian vs. Lamarckian Evolution: As mentioned before, two different strategies can be adopted for the local search procedure: Either the solution found by the local search is used only to determine the objective value of the individual or the original solution is updated. The latter one is often called Lamarckian Evolution because Jean-Baptiste Lamarck argued in 1801 that an animal could inherit characteristics which were acquired by it's parents during their live time. Baldwinian evolution, in contrary, claims that acquired properties cannot be passed on to the next generation.

While the updating strategy leads to substantially lower sizes of the best biclusters on the yeast data, it performs better than the Baldwinian strategy on the Arabidopsis data (see Figures 6 and 7). Considering that the update strategy keeps the average size of the individuals small by reducing each individual to a bicluster which fulfills the residue constraint it comes as no surprise that the running time[1] is higher for Baldwinian evolution compared to Lamarckian evolution: 631 s vs. 404 s on the yeast data and 4500 s vs 814 s on the Arabidopsis data. This effect can be expected to get stronger with an increasing difference between the size of the input matrix and the average size of biclusters that fulfill the residue threshold. Based on these considerations the following simulations are performed using the update strategy.

3) Diversity Preservation: The diversity maintenance mechanism which consists of a particular environmental selection as presented in Section III has some strong effects on the size of the biggest biclusters found in the data (see Figures 3 and 4). While this strategy clearly outperforms the standard hybrid EA with the yeast data set it fails to improve the best bicluster from the initial population in the case of the Arabidopsis data. In contrast, the average size in the population even decreases slightly. A potential explanation for this effect is the following: probably there exists one big bicluster in the yeast data and a lot of minor variations of it that still have non overlapping regions of considerable size. As a consequence the EA might focus on the same region in the matrix despite the strong selection pressure for diversity while in the Arabidopsis data set the diversity maintenance feature hinders the EA from improving the biggest bicluster. An increased pressure towards large biclusters was introduced by using a tournament size of 20 instead of three in the mating selection. This results in significant increase of the resulting bicluster size. However, compared to the hybrid EA without the diversity maintenance mechanism the best size after 50 generations is considerably smaller. Note that on the one hand runs using the diversity mechanism can increase the CPU time by a factor of ten or more. On the other hand diversity maintenance enables the EA to improve the size of the best individual for much longer than the hybrid EA without diversity maintenance (see Figure 8).

C. Finding a Set of Biclusters

The original algorithm by Cheng and Church uses an iterative approach for finding a set of b biclusters. Each time a bicluster is found the corresponding elements in the data matrix are replaced by random numbers using the same method as for missing values.[2] This results in a set of biclusters which have little overlap. The hybrid EA, however, generates a set of b biclusters in one run where b equals the population size. These biclusters often show considerable overlap.

[1]All running times were measured on a Pentium 4 2.8 GHz CPU.

[2]For the Arabidopsis data random numbers are chosen uniformly between 0 and 1200.

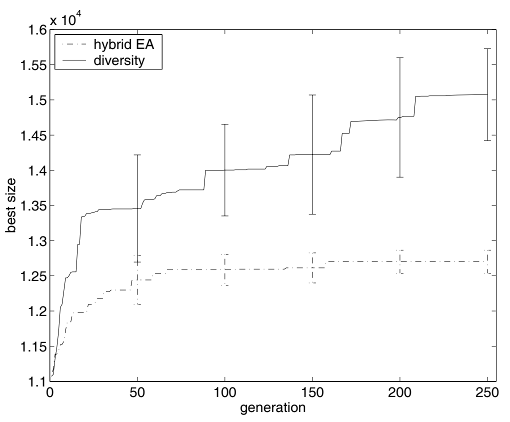

Fig. 8. Size of the best bicluster found up to the current generation for the yeast data set. (Mean over 5 runs with different random number generator seeds. The error bars have a total length of twice the standard deviation.)

One obvious way to compare two sets of biclusters is to count how many biclusters of one set are completely contained in any bicluster of the other set. This, however, was never the case in our simulation results. Thus, we need to define some kind of score based on the size of the biclusters. The average size is not an adept option since it would strongly prefer a solution that contains only copies of the largest bicluster. On the one hand too much overlap should not be rewarded with a high score, on the other hand some overlap might be biologically reasonable. We therefore focussed on the total number of elements, in the matrix E, covered by the biclusters.

In Figures 9 and 10 the coverage is depicted, i. e., the number of cells that are covered by the first k biclusters. The sequence of biclusters is chosen as follows: first pick the largest bicluster, then in each iteration pick the bicluster which contains the largest number of uncovered cells. It can be clearly seen that the hybrid EA with the diversity maintenance mechanism covers the largest part of the matrix while the EA without this pressure on diversity covers the smallest part. The set of biclusters found by the Cheng and Church algorithm lies in between. Note that the tradeoff between diversity in the population and consequently low coverage and the size of the best bicluster is nicely visible in the case of the EA with diversity mechanism on the Arabidopsis data: A higher tournament size in the mating selection leads to increased size of the best individual but decreased coverage.

While Figures 9 and 10 show the increase of coverage the average size of the first k biclusters is given in Figures 11 and 12. For the hybrid EA without any diversity maintenance all biclusters have similar size while the non-overlapping area quickly decreases. The Cheng and Church algorithm on the other extreme finds biclusters whose size rapidly decreases with the number of biclusters found.

Since the hybrid EA is able to improve the results of the Cheng and Church algorithm when just the largest bicluster in the population is considered, in principle the EA could be run b times in order to find b biclusters each time randomizing the bicluster entries like in the Cheng and Church algorithm. This, however, was not investigated here, as the run-time increases rapidly. Furthermore, we consider it a specific advantage of the EA to be able to find b biclusters in one run.

D. Summary of the results

With respect to the two main goals of finding one large bicluster and of finding a set of biclusters which cover a maximal part of the input matrix the results can be summarized as follows:

- The evolutionary algorithm without a local search procedure cannot in general find biclusters of a similar size as the Cheng and Church algorithm.
- The EA in combination with the local search method manages to significantly increase the size of the largest bicluster compared to the results of the Cheng and Church algorithm alone.
- A randomized version of the Cheng and Church algorithm can find larger biclusters than the original version but it does not perform as well as the hybrid EA.
- The hybrid EA with the diversity maintenance mechanism finds set of biclusters which cover a substantially larger part of the matrix than the biclusters found by the Cheng and Church algorithm.
- In contrary to the yeast data set, a tradeoff between diversity and the size of the best bicluster is clearly visible on the Arabidopsis data set.

V. Conclusions

This paper has introduced a general global search framework for biclustering of gene expression data. As global optimizer, an evolutionary algorithm was used that can be coupled with various greedy biclustering methods proposed in the literature. For one prominent example—the biclustering algorithm by Cheng and Church [7]—we have demonstrated that the quality of the outcome can be substantially improved by this hybrid approach. While the evolutionary algorithm alone failed to produce results comparable to those generated by Cheng and Church's algorithm, both algorithms together clearly outperformed the pure greedy strategy in terms of the quality of the resulting biclustering. Although the improvement in quality comes at the expense of additional computation time, we argue that running times of half an hour are well acceptable, especially in comparison to the time needed to do the biological experiments. Now, the biologist has the choice when to stop the algorithm; after three hours running time still improvements of the biclustering could be observed.

Acknowledgment

We would like to thank Wilhelm Gruissem and his group for providing the gene expression data for *Arabidopsis thaliana* and the REP group for many helpful discussions. We thank the anonymous referees for the detailed reviews and precise comments. This work has been supported by the SEP program at ETH Zürich under the poly project TH-8/02-2.

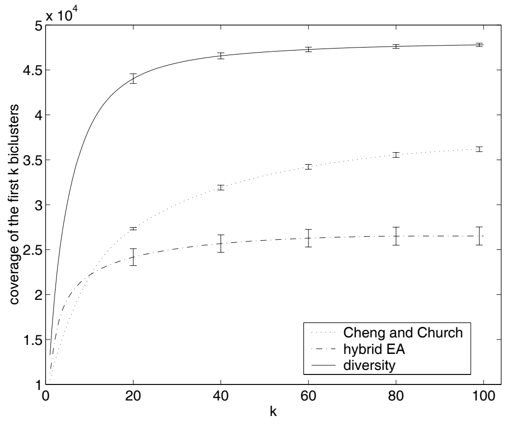

Fig. 9. Coverage for the yeast data set. Shows the number of elements in the expression matrix covered by any of the first k biclusters. (Mean over 10 runs with different random number generator seeds. The error bars have a total length of twice the standard deviation.)

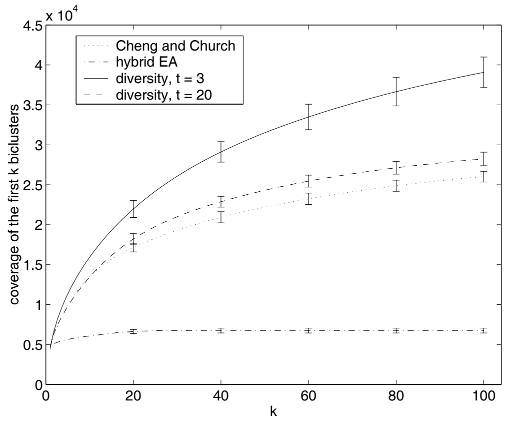

Fig. 10. Coverage for the Arabidopsis data set. Shows the number of elements in the expression matrix covered by any of the first k biclusters. (Mean over 10 runs with different random number generator seeds. The error bars have a total length of twice the standard deviation.)

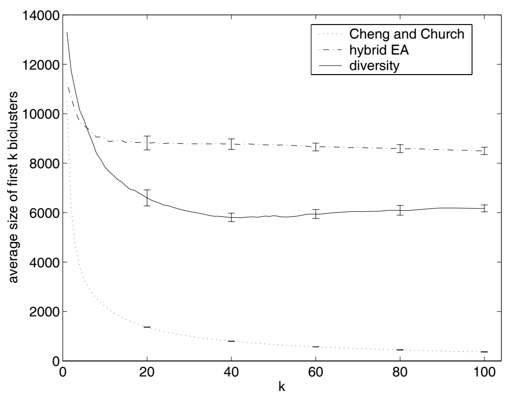

Fig. 11. Average size of the first k biclusters for the yeast data set. (Mean over 10 runs with different random number generator seeds. The error bars have a total length of twice the standard deviation.)

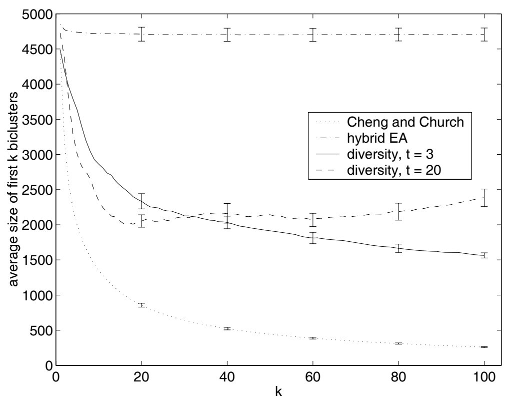

Fig. 12. Average size of the first k biclusters for the Arabidopsis data set. (Mean over 10 runs with different random number generator seeds. The error bars have a total length of twice the standard deviation.)

References

[1] J. L. DeRisi, V. R. Iyer, and P. O. Brown. Exploring the metabolic and genetic control of gene expression on a genomic scale. *Science*, 278(5338):680–686, October 1997.

[2] T. Ito, T. Chiba, R. Ozawa, M. Yoshida, M. Hattori, and Y. Sakaki. A comprehensive two-hybrid analysis to explore the yeast protein interactome. *PNAS*, 98(8):4569–4574, 2001.

[3] A. Kumar et al. Subcellular localization of the yeast proteome. *Genes & Development*, 16(6):707–719, 2002.

[4] H. Kitano. Systems biology: A brief overview. *Science*, 295(5560):1662–1664, 2002.

[5] M. B. Eisen, P. T. Spellman, P. O. Brown, and D. Botstein. Cluster analysis and display of genome-wide expression patterns. *PNAS*, 95:14863–14868, December 1998.

[6] P. Tamayo et al. Interpreting patterns of gene expression with self-organizing maps: Methods and applicataion to hematopoietic differentiation. *PNAS*, 96:2907 – 2912, March 1999.

[7] Y. Cheng and G. M. Church. Biclustering of gene expression data. In *ISMB 2000*, pages 93–103, 2000. http://cheng.ecescs.uc.edu/biclustering.

[8] A. Tanay, R. Sharan, and R. Shamir. Discovering statistically significant biclusters in gene expression data. *Bioinformatics*, 18(Suppl. 1):S136–S144, 2002.

[9] E. Segal, A. Battle, and D. Koller. Decomposing gene expresision into cellular processes. In *Pacific Symposium on Biocomputing*, pages 8:89–100, 2003.

[10] P. Merz and A. Zell. Clustering gene expression profiles with memetic algorithms. In *Parallel Problem Solving from Nature (PPSN VII)*, number 2439 in LNCS, pages 811–820, 2002.

[11] E. Falkenauer. *Genetic Algorithms and Grouping Problems*. John Wiley & Sons, 1998.

[12] J. A. Hartigan. Direct clustering of a data matrix. *Journal of the Amercian Statistical Association*, 67(337):123–129, March 1972.

[13] G. Getz, E. Levine, and E. Domany. Coupled two-way clustering analysis of gene microarray data. *PNAS*, 97(22):12079–12084, 2000.

[14] A. Ben-Dor, B. Chor, R. Karp, and Z. Yakhini. Discovering local structure in gene expression data: The order-preserving submatrix problem. In *International Conference on Computational Biology*, pages 49–57. ACM Press, 2002.

[15] J. Yang, H. Wang, W. Wang, and P. Yu. Enhanced biclustering on expression data. In *IEEE Symposium on BioInformatics and BioEngineering (BIBE'03)*. IEEE, March 2003.

[16] R. J. Cho et al. A genome-wide transcriptional analysis of the mitotic cell cycle. *Mol Cell*, 2(1):65–73, July 1998.

[17] O. Laule, A. Fürholz, H. S. Chang, T. Zhu, X. Wang, P. B. Heifetz, W. Gruissem, and B. M. Lange. Crosstalk between cytosolic and plastidial pathways of isoprenoid bio synthesis in arabidopsis thaliana. *PNAS*, 100(11):6866–6871, 2003.

[18] L. Hennig, M. Menges, J. A. H. Murray, and W. Gruissem. Arabidopsis transcript profiling on affymetrix genechip arrays. *Plant Molecular Biology*, 53(4):457–465, November 2003.

[19] M. Menges, L. Hennig, W. Gruissem, and J. A. H. Murray. Genome-wide gene expression in an arabidopsis cell suspension. *Plant Molecular Biology*, 53(4):423–442, November 2003.