Applying Genetic Algorithms to Finding the Optimal Gene Order in Displaying the Microarray Data

Huai-Kuang Tsai

Dept. of Computer Science and Information Engineering National Taiwan University, Taipei, Taiwan

d7526010@csie.ntu.edu.tw

Jinn-Moon Yang

Dept. of Biological Science and Technology & Institute of Bioinformatics, National Chiao Tung University, Hsinchu, Taiwan

moon@cc.nctu.edu.tw

Cheng-Yan Kao[1,2]

[1] Dept. of Computer Science and Information Engineering National Taiwan University, Taipei, Taiwan

[2] Bioinformatics Center, National Taiwan University, Taipei, Taiwan

cykao@csie.ntu.edu.tw

Abstract

In this paper the *Family Competition Genetic Algorithm* (FCGA) is applied to analyze DNA-microarray data. DNA Microarray technology is a significant impact on genomics study. The proposed approach consists of global and local strategies by integrating the family competition, edge assembly crossover, and neighbor-join mutation. Experiments are performed to compare the FCGA with several methods in some real-world biological data sets. Numerical results indicate that FCGA performs very robustly and is very competitive with other approaches. Using FCGA, we are able to find a gene order to display the microarray data in a meaningful way.

1 INTRODUCTION

DNA microarray technology can be applied to many biological domains, such as drug discovery, molecular diagnosis, and toxicological research. During the past few years, the development of DNA-microarray technology had provided the means to monitor the expression levels of a large number of genes simultaneously.

In the microarray experiments, messenger RNAs (mRNA) are extracted from the cell culture. Complementary DNAs (cDNA) are generated from the RNAs, amplified, labeled and then hybridized to a large array of DNA probes immobilized on a solid surface. The array is then scanned by a laser to obtain the signal for each probe region. From the signal strengths of the probes from a particular gene, one can infer the expression level of the gene in the cell type under study. Fig. 1 is the schematic procedures for monitoring gene expression using DNA microarray. With many chips, the expression data can be represented by a real-valued expression matrix X where X_{ij} is the measured expression level of gene i in experiment j.

However with thousands of genes and hundreds of experiments, it is difficult to evaluate the immense amount of gene expression profiles. A large number of approaches have been developed for analyzing the huge microarray data. For examples, clustering, classification, and genetic network analysis are usually adapted for analyzing these data. In any case, it is important to display microarry data in a meaningful way to best illustrate trends in gene expression.

An intuitive way to display microarray data is to find an optimal order of genes such that genes with similar expression profiles are blocked together. However, it is NP-complete to find an optimal order of genes [1]. Several approaches have been proposed for solving this problem. For example, the hierarchical clustering approach, a widely used tool [2-6], has been used to approximate the solution. Since the constructing process of the hierarchical tree is greedy, this approach may get stuck at local minima. Some approaches have been proposed to improve the solution quality of hierarchical clustering approach, such as flipping the internal nodes in the tree [7] and neural networks [8]. In this paper, finding an optimal order of genes is formulated as a travel salesman problem (TSP). Evolutionary approaches (EAs) are one of promising directions for solving TSPs.

Evolutionary approaches have been successfully applied to optimization problems that are inherently computationally complex [9-11]. EAs are an adaptable concept for problem solving and especially well suited for solving difficult optimization problems. They have been used to solve problems involving large search spaces, where traditional optimization methods are less efficient.

In this paper, we propose the *family competition genetic algorithm* (FCGA) to find the optimal order of genes with expression profiles. The FCGA combines a family competition, the neighbor-join mutation (NJ), and the edge assembly crossover (EAX) [12]. The family competition, derived from (1+λ)-ES and Lin-Kernigan heuristic, had been successfully applied to several continuous parameter optimization problems, such as protein docking [13] and thin-film coatings [14]. In our pervious studies [15], we had successfully integrated the family competition and EAX for solving traveling salesman problems (TSPs). In order to balance exploration and exploitation, we also designed the neighbor-join mutation [16] to cooperate with the EAX. The main difference in methodology between the present work and our previous studies is the integrations of these mechanisms.

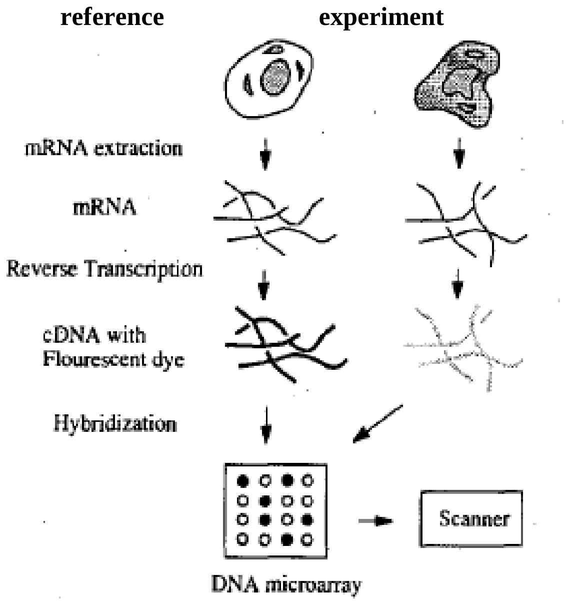


Figure 1. Schematic procedures for monitoring gene expression using DNA microarray

We illustrate features of FCGA by some TSPs benchmarks and biological data sets. The TSPs were used to verify the performance of FCGA by comparing with several methods [12][17-19]. Three biological data sets are tested to shown that FCGA is superior to the existing heuristic methods of gene order, including hierarchical clustering [2] and self-organizing map (SOM) [20]. Experimental results demonstrated that the FCGA is an encouraging approach for finding the optimal order of genes in expression profiles.

This paper is organized as follows. Section 2 describes the problem of ordering genes in expression profiles. Section 3 introduces the evolutionary nature of the FCGA. In Section 4, some experimental results are presented to illustrate the performance of the FCGA. We also compare the FCGA with various approaches on three biological problems and discuss the biological meanings. Concluding comments are drawn in Section 5.

2 PROBLEM DEFINITION

One important issue in the microarray data analysis is to display the data in a meaningful way that best illustrates the trends in gene expression. The problem can be formulated as follows: find an optimal order of genes such that genes with similar expression profiles are close together. Different criteria result in different objective functions: such as distances between gene expression profiles [1] or distances between both adjacent genes and block similarities [21]. In this paper, we used the sum of distances of adjacent genes as our fitness function defined as

$$\sum_{i=1}^{M} D(g_{\pi_i}, g_{\pi_{i+1}}), \qquad (1)$$

where g_i denote a gene, $1 \le i \le n$, π denote a gene order, M is number of genes and $D(g_i, g_j)$ is the distance of two genes g_i and g_j. This problem is the same as to determine the shortest route passing through a set of M cities under the condition that each city is visited exactly once. This so-called traveling salesman problem is well known to be NP-complete [22].

Some methods have been proposed to define the distance $D(g_i, g_j)$, or called similar, between two genes, such as *Pearson* correlation, *absolute* correlation, *Spearman Rank* correlation, *Kendall's Tau*, and *Euclidean* distance. In this paper, we applied *centered Pearson correlation* which is widely used in DNA microarray. Let $X = x_1, x_2, ..., x_k$ and $Y = y_1, y_2, ..., y_k$ be the expression levels of two genes (prepared in log-transformed data) observed over a series of k conditions. Based on *Pearson correlation* the distance of genes X and Y can be given

$$D(X,Y) = 1 - s_{X,Y}\,. \qquad (2)$$

$s_{X,Y}$ is the centered Pearson correlation defined as

$$s_{X,Y} = \frac{1}{k}\sum_{i=1}^{k}\left(\frac{x_i - \overline{X}}{\sigma_X}\right)\left(\frac{y_1 - \overline{Y}}{\sigma_Y}\right) \qquad (3)$$

where $\overline{X}$ and σ_X is the mean and standard derivation of the expression levels. The value of σ_X is

$$\sigma_X = \sqrt{\tfrac{1}{k}\sum_{i=1}^{k}(x_i - \overline{X})^2}\,. \qquad (4)$$

According to the above steps, the problem finding an optimal order of genes can be formulated as a TSP. Then we applied our method to solve this problem.

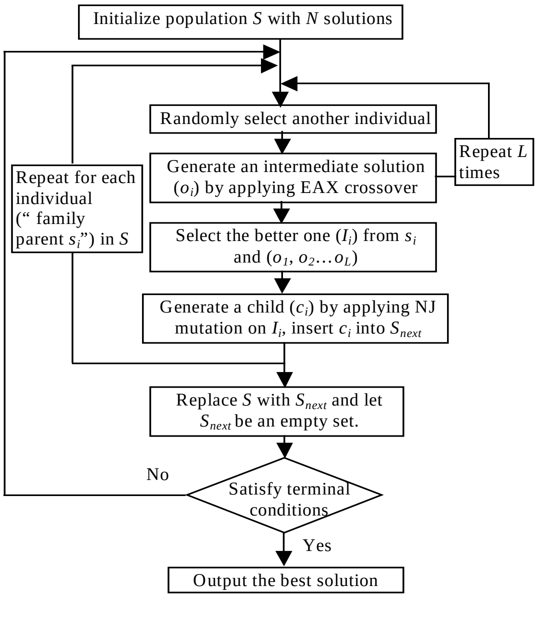

Initialize population S with N solutions
Randomly select another individual
Generate an intermediate solution (o_i) by applying EAX crossover
Repeat L times
Repeat for each individual (" family parent s_i") in S
Select the better one (I_i) from s_i and (o_1, o_2...o_L)
Generate a child (c_i) by applying NJ mutation on I_i, insert c_i into S_{next}
Replace S with S_{next} and let S_{next} be an empty set.
Satisfy terminal conditions
No
Yes
Output the best solution

Figure 2. The outline of FCGA

3 METHOD

In this section, the details of the proposed genetic algorithm, called family competition genetic algorithm (FCGA), for optimizing the gene order in gene expression data are presented. The FCGA has three major mechanisms, including a family competition, the edge assembly crossover (EAX), and the neighbor-join mutation (NJ). The EAX and the NJ mutation are genetic operators considered to be able to preserve and add good edges to generate a child. The family competition is a local search mechanism incorporated into the EAX and the NJ mutation. These three mechanisms have been studied to balance exploration and exploitation in the search space.

Fig. 2 shows the main steps of the FCGA. N solutions are generated as the initial population. Each solution is represented as a random permutation from 1 to M where M is the number of genes. After evaluating the fitness, each individual in the population sequentially becomes the "family father (s_i)" to produce L offspring, (o_1,...,o_L), by conducting the EAX and the family competition. The one with lowest fitness value from o_1,o_2...o_L and s_i becomes the intermediate offspring (I_i). The NJ is then applied to generate a child (c_i) by refining from the intermediated solution I_i. Each individual in the population sequentially executes the above steps to generate its child. These N solutions (c_1,...,c_N) become the new population of the next generation. Therefore, LN solutions are generated in one generation and N solutions are selected as the parent population of the next generation.

Our algorithm terminated when one of criteria is satisfied: 1) the maximum preset search time is exhausted, 2) all individuals of a population are the same, or 3) all of the children generated in continuous five generations are worse than their parents. Please note that both the crossover and mutation rates are 1.0. In the following subsections, the family competition, the EAX, and the NJ mutation are described.

3.1 REPRESENTATION

In the chromosome representation of our FCGA, each solution s_i represents a gene order π, where $1 \le i \le N$ and N is the population size. Assume there are M genes {g_1,…,g_M}, the solution s_i is represented as:

$$s_i = (g_{\pi_1}, g_{\pi_2}, ..., g_{\pi_M}) . \qquad (5)$$

The fitness function follows the equation (1).

3.2 FAMILY COMPETITION

The family competition, derived from (1+λ)-ES and Lin-Kernigan heuristic, is considered as a local search procedure in FCGA. In the family competition step, L offspring, (o_1,...,o_L), are generated by EAX crossover operator and after family selection, the one with best fitness from (o_1,...,o_L) and the family father (s_i) is survived. The procedure of the family competition is described as follows. Each individual (s_i) sequentially becomes the "family father." This "family father" and another solution (s_j) randomly chosen from the rest of the parent population are used as parents to do EAX crossover operation to generate an offspring (o_l). For each family father, such a procedure is repeated L times. Finally L solutions (o_1,...,o_L) are produced. After L solutions compete with "family father," only the one (I_i) with the best objective value survives. Since we create L solutions from the same "family father" and perform a selection, this is a family competition strategy. Because each individual sequentially becomes the "family father“, LN offspring are generated in one generation.

3.3 EDGE ASSEMBLY CROSSOVER

The EAX [10] is considered a powerful crossover operator [23]. It has two important features: preserving parents' edges with a novel approach and adding new edges with a greedy method, analogous to a minimal spanning tree. Several issues, such as the selection mechanism and heuristic methods, influencing EAX performance have been discussed [15][16][23][24]. In this paper the EAX is considered as the global search strategy in our proposed algorithm.

The EAX is briefly described here. Two individuals, denoted as A and B, were selected as the parents. The EAX first merges A and B into a single graph denoted G. The EAX travels G to generate many *AB-cycles* by alternately picking edges from parents A and B. According to the heuristic and random selection rules, some of *AB-cycles* are selected to generate a quasi solution which contains some disjointed subtours. Then, the EAX uses a greedy method to merge these disjointed subtours into a valid solution. This solution is returned if the fitness of this solution is better than its parents. Otherwise this procedure is repeated until a solution that is better than both A and B or K children are produced where K is the local search length.

3.4 NEIGHBOR-JOIN MUTATION

The neighbor-join (NJ) operator constructs a new solution by stealing edges from other individuals in the population or by considering the geometric information. Although the NJ is applied only on the single solution, the offspring is generated considering both the neighborhood information and knowledge from other individuals. Thus, the NJ operator is a genetic operator combining with the characteristics of local search, mutation, and recombination.

The NJ is inspired by the inver-over mutation [25] and by analyzing the TSP search space [26]. The main difference between the inver-over mutation and the other mutations is that it inherits edges both from parent and from other individuals in the current population. According to the analysis of the optimal tour of *att532*, we find that most of the links in the optimal tour of *att532* are the neighbor cities of each city.

The details of the NJ are described as follows. By given the input of an individual I_i and the search length K, the NJ applies K modifications from the start solution I_i'= I_i. In each modification, a gene c is randomly selected from I_i'. With equal probability, a gene c' is randomly selected either from the geometric nearest three neighbors of c or from the neighbor of c of an individual, which is randomly selected from the population. If the edge (c',c) does not appear in I_i', to reconnect c and c' together generates four possible types. The NJ generates four candidates by sequentially executing each type once. The one with lowest fitness from these four candidates and I_i' are selected as the parent of next loop. Above steps are executed K times.

In the four candidates, two are the simple invert operation to align c and c' together, the other two will result in two disjoint subtours. A greedy method is applied to merge two disjoint subtours into a valid solution. The greedy method works as follows: Let v_i represent a gene, (v_i, v_j), $i \neq j$, represents an edge, and $w(v_i, v_j)$ be the edge length of (v_i, v_j). At the same time, let (v_r, v_{r+1}) and (v_s, v_{s+1}) be the edges of the subtour G_r and the subtour G_s, respectively. We find a pair of edges (v_r, v_{r+1}) and (v_s, v_{s+1}) to connect these two subtours, G_r and G_s, into a legal tour by maximizing the value of the following equation:

$$w(v_r, v_{r+1}) + w(v_s, v_{s+1}) - w(v_r, v_{s+1}) - w(v_s, v_{r+1}) \quad (6)$$

$$\forall r, s;\ r \in G_r \text{ and } s \in G_s\,.$$

The new edges (v_r, v_{s+1}) and (v_s, v_{r+1}) are inserted to replace the original edges (v_r, v_{r+1}) and (v_s, v_{s+1}) to form the new solution.

4. EXPERIMENTAL RESULTS

In this section FCGA first was tested on some TSP benchmarks to verify the correctness and efficiency. Four efficient methods for TSPs were compared with FCGA to show the robustness of FCGA. FCGA is then applied on three biological data sets to find the optimal gene order. By comparing to the hierarchical clustering [2] and self-organizing map (SOM) [20], FCGA is superior to other approaches in: 1) minimizing the cost of order, 2) uncovering the correct cell cycle, and 3) most genes with the same group are aligned together. Finally we conclude this section by presenting biological results with visualized representation.

FCGA has been implemented in C++ and executed on a Pentium III 500MHz personal computer with single processor. As introduced in Section 3, the population size (N), the family competition length (L), and the local search length (K) are the main parameters in our algorithm. According to our previous study [15][16], the population size (N) is roughly set to the number of cities (for TSP problem) and the number of genes (for microarray data), while the family competition length (L) is set to 5 and the local search length (K) is set to 20 for the tradeoff between solution quality and convergence time.

4.1 RESULTS OF STSP PROBLEMS

Table I summarizes the results of our method and four other approaches, including nature crossover genetic algorithm (NGA) [18], ant colony system (ACS) [19], distance-preserving crossover genetic algorithm (DGA) [17], and EAX genetic algorithm (EGA) [12]. NGA integrated nature crossover and LK local search [27]; ACS is an ant colony system cooperated with 3-opt operator; DGA combined the distance-preserving crossover and 3-opt; and EGA used the EAX crossover. These four approaches perform well on these test problems according to our surveys. The results of first three methods were directly summarized from original papers. The average tour length and the average error of trails are used to measure the performance of comparative methods. The values in parentheses of the average tour length represent the percentages of error defined as $\frac{sol - optimum}{optimum}$, where *sol* is the experimental value and *optimum* is the optimum of a TSP problem.

Table I shows that our algorithm performs robustly for testing symmetric TSPs. For each problem the proposed algorithm can find the best tour in almost each trial and the error rate is only 0.01% away from the optimal. Since the solution qualities of FCGA applied on these TSPs are good, we believe that it would also proper to optimize the gene order in gene expression data.

TABLE I

Comparisons of FCGA with other methods, including NGA [18], ACS [19], DGA [17], and EGA[12], on five TSP problems based on the average tour length and average solution qualities (error) in 30 trails. The percentages of error defined as $\frac{sol - optimum}{optimum}$, where *sol* is the experimental value and *optimum* is the optimum of a TSP problem. "N/A" represents not available in original papers.

Problems/ (optimum)	Methods: average tour length (error in %)				
	FCGA	ACS	DGA	NGA	EGA
Lin318 (42029)	42029.00 (0.000)	N/A	42033.44 (0.011)	42029.00 (0.000)	42041.23 (0.011)
pcb442 (50778)	50778 (0.000)	N/A	50778 (0.000)	50778 (0.000)	50778 (0.000)
att532 (27686)	27688.49 (0.0081)	27718.20 (0.112)	27697.58 (0.042)	27695.61 (0.035)	27696.33 (0.037)
rat783 (8806)	8806.00 (0.000)	8837.90 (0.362)	8806.00 (0.000)	8806.00 (0.000)	8806.00 (0.000)
pcb3038 (137694)	137700.19 (0.0032)	N/A	137760.55 (0.048)	137765.02 (0.052)	137703.77 (0.007)

4.2 RESULTS OF BIOLOGICAL DATA

After the robustness of FCGA is shown, we applied it to three biological data sets to find the optimal gene order.

FCGA was compared with four widely used methods, including hierarchical agglomerative clustering algorithm (single-linkage, complete-linkage, and average-linkage [2]) and self-organizing map (SOM) [20].

In the aspect of data, the first and second data set, cell cycle-cdc15 and cell cycle, are about 800 genes which are cell cycle regulated in *saccharomyces cerevisia* with deferent number of experiments [28]. Spellman et al. [28] assigned these 800 genes to five groups termed *G1*, *S*, *S/G2*, *G2/M*, and *M/G1*. These groups approximate the commonly used cell cycle groups in the literature. The authors used a '*phasing*' method which compare the '*peak expression*' for each unknown gene with the expression of genes that were known to belong to each of these group. Although the group assignment is not the real grouping, it is still meaningful to some degree. So, we also use this information to evaluate a order of genes. The third data set, yeast complexes, is from MIPS yeast complexes database [2]. All these three data sets can be found in [7]. Table II gives the brief descriptions of each data set.

TABLE II

The description of three tested biological data set, including the source, number of experiments, and number of genes of each set.

Data name	source	Num. of experiment	Num. of genes
Cell cycle cdc15	Spellman et al. [28]	24	782
Cell cycle	Spellman et al. [28]	59	803
Yeast complexes	Eisen et al. [2]	79	979
All these data can be download from [7]			

Two scoring systems are adapted here to verify the correctness of a gene order π. Assume $\pi = (g_{\pi_1}, g_{\pi_2} ... g_{\pi_M})$ is an order of genes, where *M* is the number of genes. The first score is the fitness function which is the sum of the distance between any two consecutive genes in π, denoted as *score₁*(π). The score *Score₁*(π) is defined as:

$$Score_1(\pi) = \sum_{i=1}^{M} D(g_{\pi_i}, g_{\pi_{i+1}}), \tag{7}$$

where $g_{\pi_{M+1}} = g_{\pi_1}$, $D(g_{\pi_i}, g_{\pi_{i+1}})$ is the distance between two genes (the distance measures are defined in section 2). In fact, this is just the fitness function (equation 1). The smaller the *score₁*(π) is, the better order of genes we would get.

The second score, *score₂*(π), is to measure the overall group distribution in π. As mentioned earlier in this section, the first and second data have descriptions about gene group information. By this information, the score *Score₂*(π) is defined as:

$$Score_2(\pi) = \sum_{i=1}^{M} G(g_{\pi_i}, g_{\pi_{i+1}}), \tag{8}$$

where $g_{\pi_{M+1}} = g_{\pi_1}$, and $G(g_{\pi_i}, g_{\pi_{i+1}})$ is defined as:

$$\mathrm{G}(\mathrm{g}_{\pi_i}, g_{\pi_j}) = \begin{Bmatrix} 1, \textit{if } \mathrm{g}_{\pi_i} \textit{ and } g_{\pi_j} \textit{ are in the same group} \\ 0, \textit{if } \mathrm{g}_{\pi_i} \textit{ and } g_{\pi_j} \textit{ are not in the same group} \end{Bmatrix}$$

In a gene order π, if genes with the same groups are aligned next to each other, *score₂*(π) would be higher. In summary, we use FCGA to get the optimal gene order by minimizing the fitness function *score₁*(π). After the optimal gene order π is got, we hope the magnitude of *score₂*(π) as larger as possible. Fig.3 shows an example of calculating the score$_1$(π) and score$_2$(π).

Table III and IV summarizes the comparisons on score$_1$(π) and score$_2$(π) of our method and these four approaches. Both tables show that the FCGA performs more robustly than comparative methods for testing sets in both score$_1$(π) and score$_2$(π). By counting the factor:

$$\frac{score_2(\pi)}{number\ of\ genes}, \tag{9}$$

we found that almost 70-80% genes with the same groups are aligned next to each other. In other words, two neighbor gene in the optimal gene order π are almost in the same group. In summary, FCGA provides a way to reorder the genes in a meaningful order and aligns genes with the same group together.

Gene order: π = (2,3,6,1,7,4,5)

score₁(π) = *D*(2,3)+*D*(3,6)+ *D*(6,1)+*D*(1,7) + *D*(7,4)+*D*(4,5)

= 0.5+0.5+0.6+0.4+0.5+0.4+0.8 = 3.7

score₂(π) = *G*(2,3)+*G*(3,6)+ *G*(6,1)+*G*(1,7) +*G*(7,4)+*G*(4,5)

=1+0+0+0+0+1+0 = 2

distance matrix

	1	2	3	4	5	6	7
1	0	0.5	0.3	0.2	0.5	0.6	0.4
2	0.5	0	0.5	0.3	0.8	0.3	0.2
3	0.3	0.5	0	0.4	0.4	0.5	0.2
4	0.2	0.3	0.4	0	0.4	0.5	0.5
5	0.5	0.8	0.4	0.4	0	0.6	0.6
6	0.6	0.3	0.5	0.5	0.6	0	0.3
7	0.4	0.2	0.2	0.5	0.6	0.3	0

group category

gene	Group
1	1
2	1
3	1
4	2
5	2
6	3
7	3

Figure 3. An example of calculating the score$_1$(π) and score$_2$(π).

Table III

Comparisons of our method (FCGA) with other methods including single-linkage, complete-linkage, average-linkage [2], and self-organizing map (SOM) [20], on three gene expression data sets based on $\boldsymbol{score_1}(\pi)$.

	Cell cycle cdc15	Cell cycle	Yeast complexes
FCGA	137.347	219.233	308.801
Single-linkage	655.483	599.329	621.311
complete-linkage	227.828	486.717	435.314
average-linkage	244.792	398.15	459.529
SOM	363.453	530.635	623.169

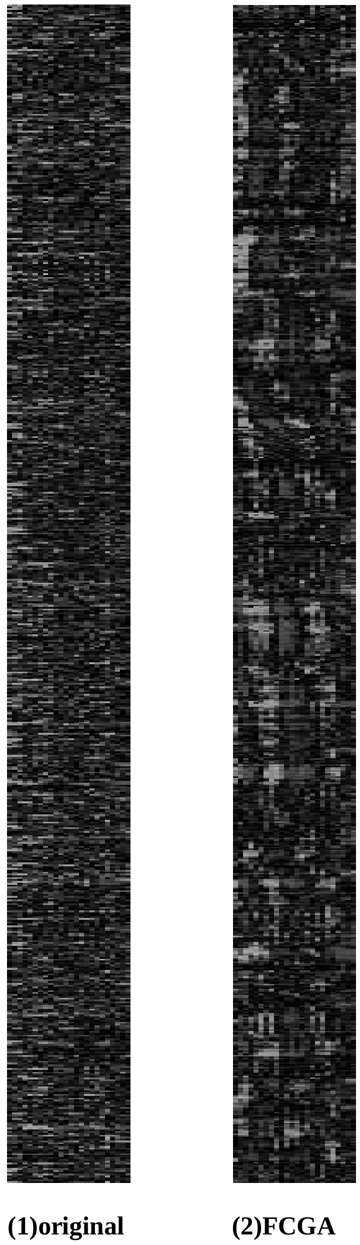
Table IV

Comparisons of our method (FCGA) with other methods including single-linkage, complete-linkage, average-linkage [2], and self-organizing map (SOM) [20], on two gene expression data sets based on $\boldsymbol{score_2}(\pi)$.

	Cell cycle cdc15	Cell cycle
FCGA	521	627
Single-linkage	251	336
complete-linkage	498	598
average-linkage	500	581
SOM	461	578

4.3 VISUALIZED RESULTS

To further understand the efficiency of our approach, we show the results by a visualized graph. Fig.4 is the gene expressions of the cell cycle cdc15 data [28] whose gene order is reordered by 1) original (random permutation) and 2) FCGA. For each gene in the figure, the expression profiles are represented as lines of color boxes and each box corresponding to one experiment. Comparing to the original data (random permutation), genes with similar expression profiles are grouped together by using FCGA. Some genes are not connected to their groups because: 1) the global minimization forces some genes to separate from their original group; 2) the missing values and the distance metric affect the overall ordering of genes. (more data results are available at http://bioinfo.csie.ntu.edu.tw/~survivor/ordering.)



(1)original **(2)FCGA**

Figure 4. The visualized gene expressions results. This figure shows the gene order of the cell cycle cdc15 data whose gene order is 1) original (random permutation) and 2) reordered by FCGA. For each gene, the expression profiles are represented as lines of color boxes and each box corresponding to one experiment. As we can see, (2) is more organized and most neighbor genes in the order have similar expression profiles.

5 CONCLUSION

This study presents that FCGA has successfully applied to solve the problem of displaying the microarray data in an optimal order. FCGA keeps the population diversity via the family competition and efficiently search the solution space via incorporating EAX crossover and NJ mutation. Experiments of the TSPs verify that the proposed approach is very comparative with other evolutionary algorithms. Using FCGA on the biological data can recover the correct cell cycle and group similar genes in an optimal order. We believe that the flexibility and robustness of the FCGA make it an effective tool of analyzing microarray data.

In the future, we will: 1) test more biological data set to reveal new biological facts; 2) use different distance metrics to produce better results; and 3) investigate different objective functions for nonparametric clustering via FCGA.

References

[1]. Biedl, T., Brejova, B., Demaine, E. D., Hamel, A. M., and Vinai, T., "Optimal Arrangement of leaves in the tree representing hierarchical clustering of gene expression data," Technical report, Nov. 2001.

[2]. Eisen, M. B., Spellman, P. T., Brown, P. O., and Botstein, D., "Cluster analysis and display of genome-wide expression patterns," *Proc. Natl. Acad. Sci.*, pp. 14863–14868, 1998.

[3]. Alizadeh, A. A., Eisen, M. B., et al., "Distinct types of diffuse large B-cell lymphoma identified by gene expression profiling," *Nature*, 403(6769), pp. 503-511, 2000.

[4]. Kawasaki, S., Borchert, C., et al., "Gene expression profiles during the initial phase of salt stress in rice," *Plant Cell*, 13(4), pp. 889-906, 2001.

[5]. Khodursky, A. B., Peter, B. J., et al., "DNA microarray analysis of gene expression in response to physiological and genetic changes that affect tryptophan metabolism in Escherichia coli," *Proc. Natl. Acad. Sci.*, pp. 12170-12175, 2000.

[6]. Schaffer, R., Landgraf, J., et al., "Microarray Analysis of Diurnal and Circadian-Regulated Genes in Arabidopsis," *Plant Cell*, 13(1), pp. 113-123, 2001.

[7]. Bar-Joseph, Ziv., Gifford, D. K., and Jaakkola, T. S., "Fast optimal leaf ordering for hierarchical clustering," *Bioinformatics*, vol. 17, suppl. 1, pp. s22-29, 2001, *http://www.psrg.lcs.mit.edu/clustering/ismb01/optimal.html*.

[8]. Herrero, J., Valencia, A., and Dopazo, J., "A hierarchical unsupervised growing neural network for clustering gene expression patterns," *Bioinformatics*, vol. 17, pp. 126-136, 2001.

[9]. Goldberg, D. E., *Genetic algorithms in search, optimization & machine learning.* Reading, MA: Addison-Wesley, 1989.

[10]. Chu, P. C. "A Genetic Algorithm for the Multidimensional Knapsack Problem," *Journal of Heuristics*, vol. 4, pp. 63-86,1998.

[11]. Dandekar, T. and Argos, P., "Folding the main chain of small proteins with the genetic algorithm," *J. Mol. Biol.*, vol. 236, pp. 844- 861, 1994.

[12]. Nagata, Y. and Kobayashi, S., "Edge assembly crossover: A high-power genetic algorithm for the traveling salesman problem," in *Proceeding of the seventh international Conference on Genetic Algorithms* (*ICGA*), 1997, pp. 450-457.

[13]. Yang, J. M. and Kao, C.Y. "A Family competition evolutionary algorithm for automated docking of flexible ligands to Proteins," *IEEE Trans. on Information Technology in Biomedicine*, vol. 4, no. 3, pp. 225-237, 2000.

[14]. Yang, J. M., Horng, J.T, Lin, C. J., and Kao, C.Y. "Optical coating designs using an evolutionary algorithm," *Evolutionary Computation*, vol. 9, no.4, pp. 421-443, 2001.

[15]. Tsai, H. K., Yang, J. M., and Kao, C. Y. (2001) "A genetic algorithm for traveling salesman problems," *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2001)*, pp.687-693.

[16]. Tsai, H. K., Yang, J. M., and Kao, C. Y., "Solving Traveling Salesman Problems by Combining Global and Local Search Mechanisms," *Proceedings of the Congress on Evolutionary Computation* (CEC), 2002, *to appear*.

[17]. Dorigo, M. and Gambardella, L. M. (1997) "Ant colony system: A cooperative learning approach to the traveling salesman problem," *IEEE Trans. on Evolutionary Computation*, vol.1, no.1, pp53-66.

[18]. Freisleben, B. and Merz, P. (1996) "New genetic local search operators for the traveling salesman problem," *In Parallel Problem Solving from Nature IV*, Springer-Verlag, pp. 890-899.

[19]. Jung, S. and Moon B. R. (2000) "The nature crossover for the 2D Euclidean TSP," *Genetic and Evolutionary Computation Conference (GECCO 2000)*, pp. 1003-1010.

[20]. Tamato,P., Slonim, D., Mesirov, J., Zhu, Q., Kitareewan, S., Dmitrovsky, E., LANDER, E. S., and GOLUB., T. R., "Interpreting patterns of gene expression with self-organizing maps: methods and application to hematopoietic differentiation," *Proc. Natl. Acad. Sci.*, vol 96, pp. 2907-2912, 1999.

[21]. Amir, B. D., Ron, S., and Zohar, Y. "Clustering Gene Expression Patterns," *journal of computational biology*, vol. 6, pp. 281-297, 1999.

[22]. Garey, M. R. and Johnson, D. S., "Computers and Intractability: A Guide to the Theory of NP-Completeness," *Freeman*, 1979.

[23]. Watson, J., Ross, C., Eisele, V., Denton, J., Bins, J., Guerra, C., Whitely, D., and Howe, A. (1998) "The traveling salesrep problem, edge assembly crossover, and 2-opt," *In Parallel Problem Solving from Nature V, A. E. Eiben et al, eds. Springer-Verlag,* pp.823-832.

[24]. Nagata, Y. and Kobayashi, S. (1999) "An analysis of edge assembly crossover for the traveling salesman problem," *IEEE International Conference on Systems Man and Cybernetics*, pp. 628-633.

[25]. Tao, G. and Michalewicz, Z. (1998) "Inver-over Operator for the TSP," *In Parallel Problem Solving from Nature V,* Springer-Verlag, pp.803-812.

[26]. Padberg, M. and Rinaldi, G. (1987) "Optimization of a 532-city symmetric traveling salesman problem by branch and cut," *Operation Research Letters*, vol. 6, pp.1-7.

[27]. Lin, S. and Kernighan B. (1973) "An effective heuristic algorithms for the traveling salesman problem," *Operations Research,* Vol.21, pp.498-516.

[28]. Spellman, T. S., Sherlock, G., & et al., "Comprehensive identification of cell cycle-regulated genes of the yeast *saccharomyces cerevisia* by microarray hybridization," *Mol. Biol. of the Cell*, vol. 9, pp. 3273-3297, 1998.