한국퍼지 및 지능 시스템학회 논문지 1998, Vol. 8, No. 5. (1998) 61 ~ 71

Parallel Genetic Algorithm based on a Multiprocessor System FIN and Its Application to a Classifier Machine

Myung-Mook Han

KyungWon University

ABSTRACT

Genetic Algorithm (GA) is a method of approaching optimization problems by modeling and simulating the biological evolution. GA needs large time-consuming, so it had better do on a parallel computer architecture. Our proposed system has a VLSI-oriented interconnection network, which is constructed from a viewpoint of fractal geometry, so that self-similarity is considered in its configuration. The approach to Parallel Genetic Algorithm (PGA) on our proposed system is explained, and then, we construct the classifier system such that the set of samples is classified into several classes based on the features of each sample. In the process of designing the classifier system, we have applied PGA to the Traveling Salesman Problem and classified the sample set in the Euclidean space into several categories with a measure of the distance.

1. Introduction

Genetic Algorithm (GA) is search technique which was proposed by John Holland in 1975, and has been demonstrated to be robust in searching very large spaces in a wide variety of application [1-3]. The search techniques may be classified into three classes: *Calculus-based techniques*, *Guided random search techniques*, and *Enumerative techniques* [4, 5]. *Calculus-based techniques* utilize a set of conditions which are satisfied by the solutions of an optimization problem. *Enumerative techniques* search one point at a time until every point associated with an specific domain space is examined. *Guided random search techniques*, which are further classified into simulated annealing and evolutionary algorithms, use not only enumerative methods but also additional information to conduct the search. Simulated annealing utilizes a thermodynamic process to converge to the global optimum. On the other hand, evolutionary algorithms are based on natural selection principles. These techniques subdivide into evolution strategies, evolutionary programming, and genetic algorithm.

GA maintains a fixed size N population of individuals, to which standard, basic GA operators such as selection, crossover, and mutation are applied. The entire current population of N individuals is replaced by a new population created by applying the three operators described above to the current population. The initial population consists of N randomly generated individuals. Each individual is then evaluated for fitness. With fitness assigned, the three operators can be applied to create the next generation. This process is repeated until some stopping criterion is reached.

Due to increasing demand of GA in the application of complex real-world problems, there is a growing interest for fast and flexible implementation. Parallel processing is the natural way to consider.

PGA uses the characteristics of parallel computers to design a model that is similar to natural evolution through the concept of *spatial locality*. The traditional GA selects parents and replacements from the entire population of individuals.

In a realistic model, however, a species of creatures in nature is usually not arranged as a large, entire population. Rather a species of creatures is spread into several subpopulations, each of which can evolve independent to other subpopulations. Since the communication costs are greatly reduced in the parallel architecture, it is necessary for a parallel computer to implement the PGA. Besides, PGA improves two aspects. First, parallel processing increases the execution speed. Second, a more flexible improvement comes from the spatial

relationships among the population of individuals.

The Fractal geometry-based Interconnection Network(FIN) we have proposed is constructed by connecting the basic units recursively [6]. The basic unit of the FIN is the structure composed of the processor, memory, and I/O ports. FIN has a VLSI-oriented interconnection network, which is constructed from a viewpoint of fractal geometry so that self-similarity is considered in its configuration. The network has the property of a parallel processing and several parallel algorithms have been proposed, and can be regarded as a member of the star graph. Even though a great deal of research has been directed towards studying the topological properties of hypercubes, there are papers which report the topological superiority of the star graph over the hypercube including the analysis of the topological, communication and computation merits of star graph configured parallel processors.

Since the PGA is designed to utilize a parallel architecture, we shall consider the communication among the processors which is performed through message passing. A n dimensional hypercube is made up of $N=2^n$ processors such that each processor is connected to n other processors. Therefore, it is possible that the distance between any two processors is at most n communication links long. This indicates that information can be passed in (log N) time steps in a cube. On the other hand, a n dimensional star graph consists of $N=n!$ processors such that each processor is connected to $n-1$ other processors. Also information can be broadcast in $\left\lfloor \frac{3(n-1)}{2} \right\rfloor$ time steps, where the greatest integer less than or equal to x is represented by $[x]$. We can consequently see that based on the communication merit, the star graph is superior to the comparable n dimensional hypercube for the same number of processor. It is considered that FIN, having a recursive nature and a hierarchical structure, is suitable to implement GA efficiently because the proposed methods can quickly lead to the global optima and spread through the whole population. That is, the characteristics of the proposed methods are (1) since the selection operator is performed in a variable neighborhood, good mate can be chosen to a wide extent, (2) by the proposed crossover, good individuals spread rapidly through the entire population. Also, the recursive and the hierarchical structure is desirable for many computation and communication problems.

In this paper, we discuss the PGA using the self-similarity structure of FIN. The performance of the Simple Genetic Algorithm (SGA) and the PGA are examined on the Traveling Salesman Problem (TSP) and compared to the performance of our proposed methods. We also construct the classifier system so that the set of samples are classified into several classes based on the feature belonging to each sample. In the process of designing the classifier system, we have applied PGA to the TSP and classified the sample set in the Euclidean space into several categories with a measure of the distance. Finally, we conclude the paper.

2. A Multiprocessor System with Fractal geometry-based Interconnection Network (FIN) and Genetic Algorithm (GA)

2.1 FIN

FIN is a multiprocessor system which has an interconnection network constructed from the viewpoint of fractal geometry so that its configuration has the characteristics of self-similarity. In other words, FIN is composed of the plural processing element (PE) parts and the communication links to connect each PE. The Fractal geometry-based Interconnection Network is the network where each local network part is reduced from the global part network recurrently. Each PE is able to fulfil the basic instruction which consists of arithmetic-logic operation, comparison, conditional divergence, and transmission-reception operation with the optional register. It is expected to be the case that each PE has an enough local memory and its register has a sufficient bit-length. The communication link is connected with the input-output port of each PE, and each PE can exchange the data with its connected PE through this port to transmit data streams in parallel. We define the connective number as the number of links of each PE, which is represented as n.

In this paper, we define FIN-1 as the FIN with the connective number three, and explain the system

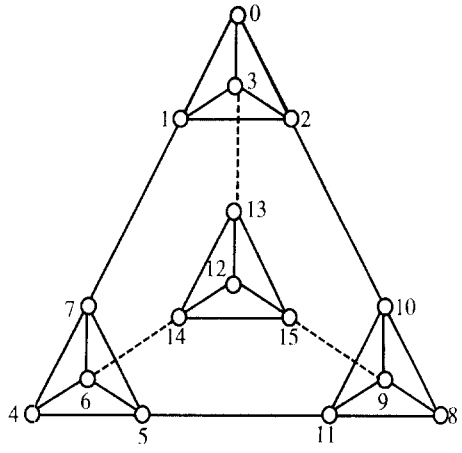

Fig. 1. FIN of level 2 with n=4.

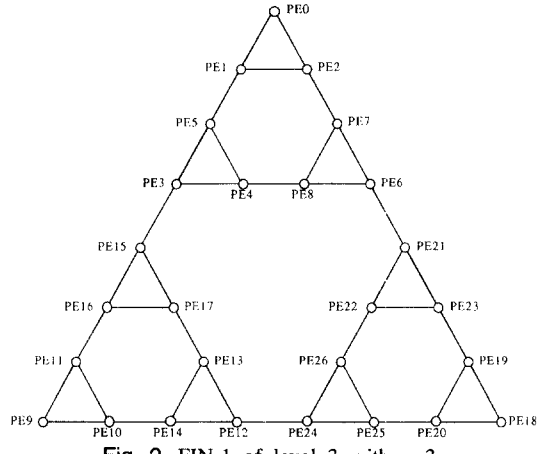

Fig. 2. FIN-1 of level 3 with n=3.

using FIN-1. Since the network is constructed recursively, we introduce concepts of level and a block which consists of some blocks or PEs. A block of level 1 consists of three PEs interconnected. A block of level $l(l>=1)$ consists of three blocks of level $l-1$ interconnected by communication links. For example, Fig. 1 shows FIN of level 2 with connective number 4, which is constructed with the connection of 4 blocks of level 1. In addition, let a block of level 0 be one PE for convenience. The connection between blocks means that a PE of the block connects with a PE of the other block. The PEs included in the block of level l, connecting to the other block of level l, are referred to corner PEs of the block of level l. We refer the blocks of level l containing corner PEs of block of level l+1 to corner blocks of level l and the PE whose address is the smallest in the block to the leader PE of the block.

In a block of level l, there are three blocks of level $l-1$ except in the case of l=0. We identify these blocks by the block address. The block address of one corner block of level l is 1, and another is 2. The block that is not a corner block is assigned 0 as its address. We assign the address of the block at level l to be b_l. However, same addresses are available; in a block of level l, there are three blocks of level $l-1$ and each of these may have blocks of level $l-2$, and the block address only varies 0 to 2, so that the block of level l have three blocks that have, say, 0 as the block address of level $l-2$.

Each PE has the unique address that is defined from the block addresses to which that PE is belonging, as follows:

$$PE_{address} = \sum_{l=1}^{l_{max}} b_l n^{l-1} \quad (1)$$

Here, b_l is a block address of level l which include that PE and $l_{max}=\log_n N$. We refer the PE of the address i to PE_i.

Fig. 1 shows FIN of level 2 with connective number 4, which is constructed with the connection of 4 blocks of level 1. Fig. 2 illustrates FIN-1 of level 3 with connective number 3, which is constructed with the connection of 3 blocks of level 2. In Fig. 3, several networks are illustrated with connective number and level. The first row is the basic element and several networks in the second

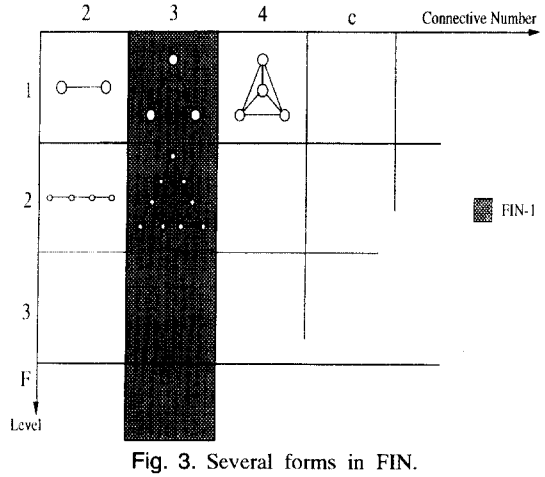

Fig. 3. Several forms in FIN.

row are constructed from their basic elements. In this paper, it is defined FIN-1 as having connective number three.

2.2 Parallel Genetic Algorithm

Following Holland's SGA proposal, considerable research involving PGA has been investigated in order to improve GA performance. PGA uses the characteristics of parallel computers and design a model that is similar to natural evolution with the concept of spatial locality. As Darwin already pointed out, the spatial environment is an important fact in evolution. By this consideration, the evolution process can be regarded as the parallel evolution process which is an interaction between the individuals and/or subpopulation and its environment. With regards to the PGA, there have been two types of development beyond the SGA. The first is the fine-grained PGA, in which one individual is assigned to each processor(subpopulation) and the genetic operator is performed near the individual which is activated [7-9]. The fine-grained PGA is designed to take advantage of machines with a large number of processors. In order to avoid the high communication overhead, the fine-grained PGA generally imposes some limitations on mating based on the distance. The processing of a generation is highly parallelizable, and each string in the next generation can be generated in parallel.

The potential drawback of the fine-grained PGA is that local optima can quickly spread the entire population. Since there is constant swapping between subpopulations, the possibility of independent evolutions may be hindered. This means that good individuals from one subpopulation quickly spread to other subpopulations, and the evolution no longer remains independent. The independent evolutions provide advanced method. For example, they are able to select the best evolution(including parameter values) among several subpopulations.

The second type is the distributed PGA, in which the set of individuals is assigned to each processor (subpopulation) and several subpopulations are evolved in parallel [10-12]. Each subpopulation is panmictic, so that individuals can only mate within their subpopulation. The subpopulations may communicate by migrations in which individuals move from one subpopulation to another. The merit of the distributed PGA is that each of the subpopulations runs a separate GA independently. That is, controlling interaction between subpopulations divides a large population into several subpopulations, in which one can perform several GA runs with smaller populations. One of the drawbacks of the distributed PGA is that the behavior of the distributed PGA tends to converge locally because the number of individuals of a subpopulation is small.

2.3 Parallel Genetic Algorithm on FIN

We present two kinds of PGA models that utilize the self-similarity structure of FIN.

The first is based on the fine-grained parallel model in which each individual is assigned to one PE and it is able to use the model in which the number of PEs is equivalent to the number of individuals to be required for the problem. For example, 64 individuals are performed on the FIN of level 4 with connective number 4. Also 81 individuals are carried out on the FIN of level 4 with connective number 3. The construction of initial solutions and the calculation of evaluation values can be performed simultaneously. Therefore, the n^{L-1} calculation can be done simultaneously in the level L with the connective number n. Besides, the genetic operator, such as crossover, can be operated in the plural place through the communication links concurrently.

Here, we shall define the crossover, the most important one of genetic operators, especially hierarchical crossover operation in FIN. The hierarchical crossover operation is performed for the model which has various forms depending on the connective number and the level of hierarchies.

The crossover of level l: The corner PEs of $l-1$ sub-blocks in the block of each level l hold parent individuals, and the individuals of the corner pair PEs which are linked to one another are crossovered concurrently.

For example, as for the crossover of level 3, the crossover operation is performed at 3 places (PE3 and PE15, PE6 and PE21, and PE12 and PE24)

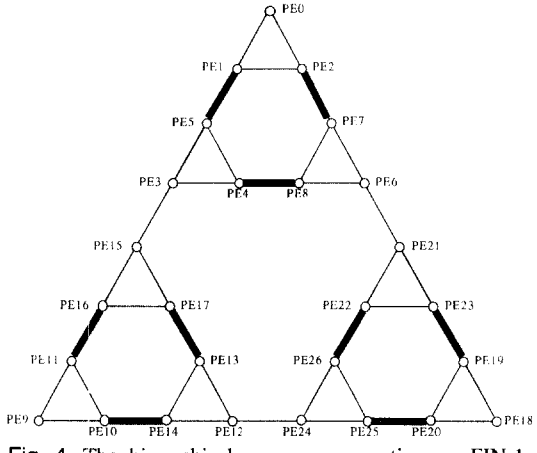

Fig. 4. The hierarchical crossover operation on FIN-1.

concurrently in Fig. 4. As for the crossover of level 2, the operation is carried out at 9 places (thick line) simultaneously. Also, as for the cross-over of level 1, the operation is done at 9 places(blocks of level 1) at the same time.

Since the crossover operations are done in the sub-blocks of level l simultaneously, the crossover operations of level $l(l>1)$ in the level L of FIN-1 can be implemented 3^{L+1-l} concurrently. Besides, it can correspond hierarchically with the problem to be divided into the sub-problems. We applied the model, which uses the hierarchical crossover operation in FIN, to several problems and obtained good results [13].

The second is based on the type combining the above model with the characteristics of the distributed PGA model. That is, each individual was processed as the type of the above model but the migration operator was performed between sub-populations. It belongs to a category that has not yet been explored.

3. Modeling and Implementation to the Traveling Salesman Problem

In order to evaluate the performance of the four kinds of GAs (SGA, PGA, fine-grained PGA on FIN, and distributed PGA on FIN), we applied each to the Traveling Salesman Problem (TSP). The TSP is considered an NP-hard problem in which computation time for a solution increases exponentially with the size of the problem. To implement the experiment, the procedure is to perform in a direction that minimizes the fitness function, hoping that good solution will be reached with the GA.

The TSP can be described as follows.

Let $S=\{s_1, s_2, \cdots, s_n\}$ be the cities and $\forall s_i, s_j \in S$, $D(s_i, s_j)$ denotes the distance between the city i and city j.

Then the genotype is a permutation of S

$$GT = (s_{\Pi(1)}, s_{\Pi(2)}, \cdots, s_{\Pi(n)}) \tag{2}$$

and the phenotype is

$$PT = \sum_{i=1}^{n-1} D(s_{\Pi(i)}, s_{\Pi(i+1)}) + D(s_{\Pi(n)}, s_{\Pi(1)}) \tag{3}$$

The statement is: if n cities are randomly located on the 2 dimensional Euclidean space, find out the shortest path that visits each city exactly once. In this paper, we used the individual as the order of the visiting city number. Based on the Grefenstette's method [1], we used the crossover in order to avoid the fatal individual which may come out through a single crossover. With this method, it is possible that the phenotype corresponds to the genotype by 1 to 1 and the genes of offsprings can receive the information of the city linkage from the parent. To generate the initial population, we selected a city at random and then searched its nearest city among the nonvisited cities. We continued the above way until the whole cities were visited. Because it happens to make the different traveling paths by selecting the initial city, we distribute the different initial city to the entire population to have different kinds of individuals as much as possible. The method to choose the initial population of the TSP by Grefenstette is regarded as the excellent one for using the specific heuristics of a problem. The method is described as follows: start with a randomly chosen city, add an edge to the nearest one in a random sample of unvisited cities, and repeat until all cities have been visited. This method is based on the belief that, among the cities, the nearest one is involved plentifully in the optimal solution of the TSP, but it is impossible to create many kinds of the traveling paths as the initial population. However, the optimal

solution in practice is not only the nearest one among the cities but also the cities around the nearest one. Therefore, we used the method to choose the initial population as follows: start with a randomly chosen city, select an edge with the probability that is in proportion to a reciprocal number of distance in unvisited cities and connect it with the path until all cities have been visited. While the genotype corresponds to the phenotype by 1 : 1 in the coding method, the evaluation function can be measured as the length of the traveling path. The genetic operators, such as crossover and mutation, were performed. Crossover played a primary role in the operation of GA. Through sporadic and random alternation of the bits of individuals, mutation played a secondary operator with the role of restoring lost genetic material.

3.1 SGA applied to the TSP

To derive a solution to a problem, the SGA creates an initial single population which has 64 individuals in this experiment. The individuals in the single population are selected and reproduced by a roulette wheel method. That is, the probability that an individual is chosen is proportional to its fitness value. An individual with high fitness value may have several copies in the population. Crossover provides a powerful exploration ability by exchanging parts of two parents. The crossover rate, the probability that each mating pair executes crossover, was set 0.6, 0.7, 0.8 and 0.9 in our tests. As one of many domain specific local search techniques for the TSP, we used the heuristic mutation operator (inversion) that simply reverses a randomly chosen subtour. The mutation rate was set 0.06, 0.07, 0.08 and 0.09. All 16 different possible combinations of these parameter values were tested on the SGA with 64 individuals of population. Each simulation was run five times for 10000 generations and took the average of the results. Among the simulations, we found the best result in the experiment in which the crossover rate and mutation rate was 0.9 and 0.06, respectively, which is shown in Fig. 5. There is a great deal of research of parameter settings. But there are no conclusive results on what is best. Most people use what has workded well in previously reported case. De Jong performed an early study of how varying parameters affected the GA's performance on a small suite of test functions. Since De Jong's experiments indicated that the best population size was 50-100 individuals, 64 individuals of population were used.

3.2 PGA applied to the TSP

The traditional PGA requires local genetic interactions. Therefore, the performance has to be changed from global to local mechanisms. The population of individuals are held on a 2-dimensional grid in which edge elements wrap around, forming a ring. Each grid element has 8 adjacent elements on which genetic operators are performed. A grid element was selected at random, and with its immediate adjacent 8 individuals form subpopulation. Two individuals are selected as the parents from this subpopulation according to their fitness values. For our implementation, we selected the best of these 8, and performed a crossover with the individual at the grid node. If the offspring was better than the original, the offspring was kept. Otherwise the original individual was kept. Parameter setting was the same as one of the previous experiment and also each simulation was run five times for 10000 generations. The best result in the experiment had a crossover rate 0.9 and a mutation rate 0.06 and is shown in Fig. 5.

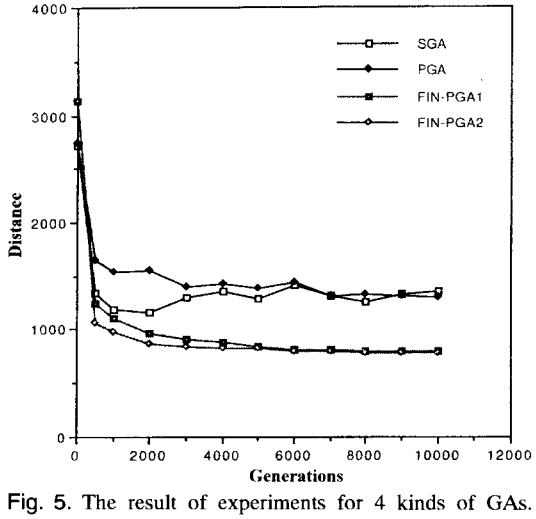

Fig. 5. The result of experiments for 4 kinds of GAs.

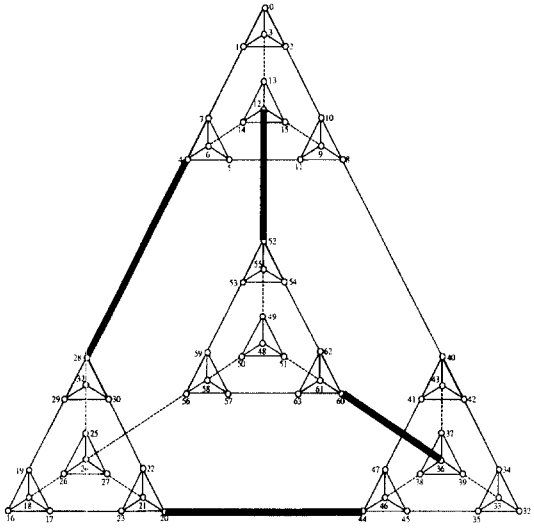

Fig. 6. The model of FIN-PGA1.

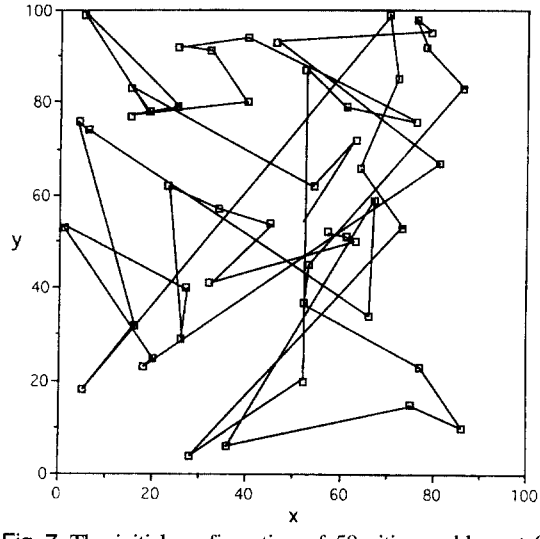

Fig. 7. The initial configuration of 50 cities problem at 0 generation.

3.3 The distributed-grained PGA using FIN applied to the TSP

We have mentioned four aspects for categorizing GA previously. In this section, we proposed a new model that has not yet been explored. Fig. 6 shows the model of FIN-PGA1 which has 4 sub-populations (0-15, 16-31, 32-47 and 48-63). Each sub-population was processed as the type of the PGA on FIN but the migration operator was performed between sub-populations (4-28, 20-44, 36-60 and 52-12). Therefore, each individual of the new model was assigned to a processor with local mating and pannmictic migration using FIN topology. This describes a fine-grained PGA with periodic migration. The use of panmictic migration enables interaction between distant demes periodically. The simulation was run ten times for 10000 generations. The migration operator selects the best individual from one sub-population and sends it to the next sub-population to replace the worst individual. The average values of 10 simulations is shown in Fig. 5.

3.4 The fine-grained PGA using FIN applied to the TSP

As the fine-grained parallel model, FIN-PGA2 fully utilizes the FIN architecture. Each individual was assigned to one PE and hierarchical crossover was performed. Each simulation was run ten times for 10000 generations and the average value is shown in Fig. 5.

We represent the progress of the solutions to the TSP for 50 cities randomly distributed in the 100× 100 square. Fig. 7 shows the configuration at 0 generation. The tour length of the initial path was 1509. Fig. 8 displays the final configuration at 3800 generations and the tour length of 594 was deduced from the final configuration. Next we extended the experiment to 100 cities randomly distributed in the

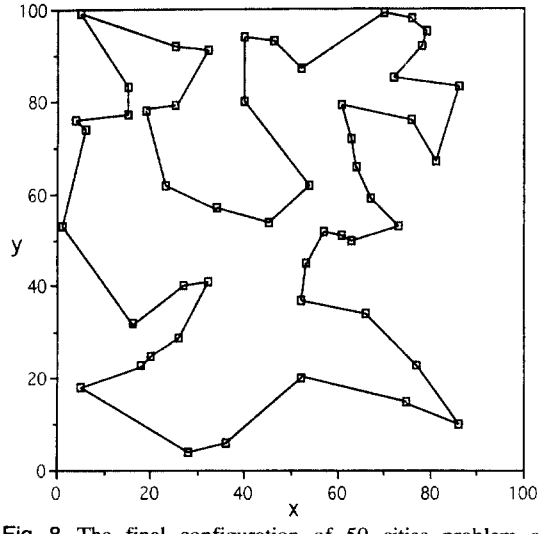

Fig. 8. The final configuration of 50 cities problem at 3800 generations.

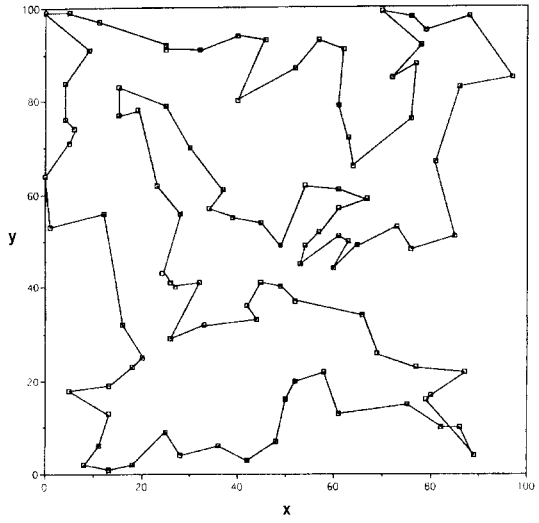

Fig. 9. The final configuration of 100 cities problem at 7500 generations.

same square. Path gradually decreased as the number of generations were increased. Fig. 9 illustrates the final configuration at 7500 generations and the tour length was reduced to 825.

4. Classifier System using the TSP

We usually regard the classification model as the one consisted of three parts: a transducer, a feature extractor, and a classifier.

The input data are perceived by the transducer and the dimensionality of the perceived data are reduced by the feature extractor. Finally, the classifier evaluates and classifies several classes in the feature space. In this chapter, we concentrate on the classifier only and use the input data in which the nation are regard as 2-dimensional object, that is, each nation have 2 feature values. We can regard the classifier design as dividing the feature space into several disjoint regions. Suppose there are n samples, W_1, W_2, $\cdots$, W_n, and these samples are expressed as points in the feature space Ω.

A classifier divides Ω into n regions,

$$\Omega = \cup_{i=1}^{n} \Omega_i^F, \quad \Omega_i^F \cap \Omega_j^F = \varnothing, \tag{4}$$

If the number of classes in the given problem is n, the time-complexity for the classifier to make a decision is $O(\log n)$. Therefore, when n is a very large number, the time-complexity is very critical. When a high speed approach in large set classification is considered, the parallel architectures are recommended. The main advantage of a classifier on FIN-1 is that the global decision can be divided into a number of local decisions at different levels of the FIN-1 and as a result can be made more quickly than the single stage classifier. Here, we may use the fact that samples which belong to the same class have relation to others in close location and samples which belong to the different classes have relation to distant place. If the sample to be given is thought of as a city and the shortest path among samples is found out, different classes are separated by using idea that the edges connecting different classes became longer than the edges in the same class. In order to find out the shortest path of traveling, the GA on FIN-1 would be run in order to find out the shortest path of travel. Because there are many cases where the set of samples have hierarchical structure, we take the above operation recurrsively on the local network of FIN-1. At the end, it repeats until a node of the network corresponds to a sample. It happens that the classifier machine is constructed on FIN-1 by this method. The algorithm to construct a Classifier System on FIN-1 is as follows.

[*The Construct Algorithm of a Classifier System*]

1. For the given samples, solve TSP on FIN-1.
2. Calculate the total distance of the path in step 1.
3. Select the maximum distance among distance.
   cutting distance $\Leftarrow$ *the maximum distance.*
   Cut the edge of the maximum distance.
4. *New distance* $\Leftarrow$ *total distance* $-$ *cutting distance*
   Divi $\Leftarrow$ *New distance* / n
   (n is the number of classifing classes and is equivalent to connective number.)
5. Add the sum of next citie's distance from the cutting one until it reaches around *Divi*.
6. Compare the distance of the edge included in *Divi* with distances before and after the edge. If the first or last edge is included in *Divi*, compare the distance of the edge included in *Divi* with distance after or before the edge only.
7. Select the maximum distance among them and cut that edge.
8. If it reaches n, Stop.

	X: Political alignment	Y: Economic development
U.S.A.	.290	.334
U.K	.263	.310
West Germany	.261	.326
France	.175	.223
Israel	.168	.108
Japan	.120	.115
South Africa	.069	.268
Greece	.065	-.092
Spain	.189	-.038
Brazil	.212	-.186
Mexico	.196	-.156
Ethiopia	-.012	-.296
India	-.064	-.295
Indonesia	-.052	-.301
Congo	-.086	-.283
Egypt	-.215	-.220
China	-.403	-.114
Cuba	-.283	-.080
Yugoslavia	-.234	.051
Poland	-.308	.126
U.S.S.R	-.353	.201

Fig. 10. Stimulus coordinates on the X, Y dimensions.

$n \Leftarrow n+1$

Go to 4.

As the experimental data of the example, we use the statistic from "Differences in perceived similarity of nations" [14]. The interpretation *political alignment* given to X dimension was based on the fact that nations aligned with the U.S.A. (the pro-Western nations) and nations aligned with U.S.S.R. or China (the pro-Communist nations) are at opposite extremes. Since the more developed nations generally project higher on the vertical dimension, dimension Y was labeled *economic development* shown in Fig. 10. The countries are presented by the X, Y coordinates. Fig. 11 displays the shortest path among samples. It is classified recursively by using the above algorithm, and Fig. 12 shows the overall performance of these steps and we obtain the classifier system on FIN-1 as illustrated in Fig. 13. The Classifier System has classified the set of nations

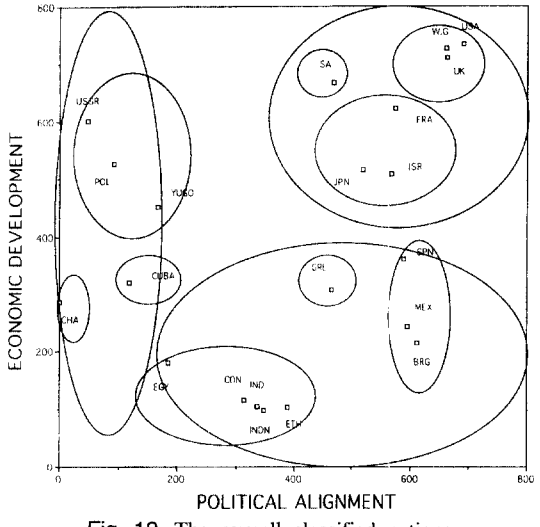

Fig. 12. The overall classified nations.

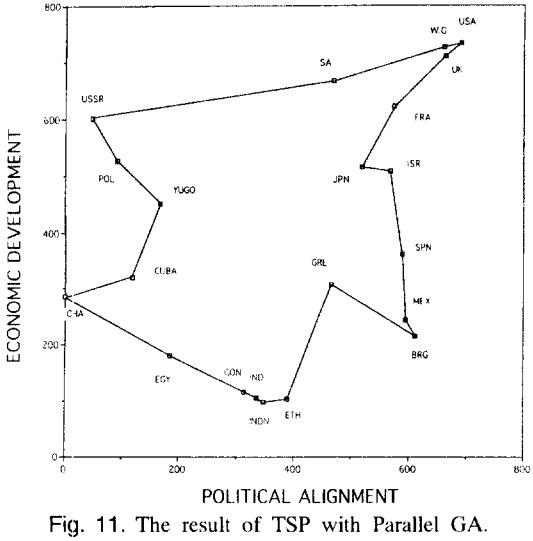

Fig. 11. The result of TSP with Parallel GA.

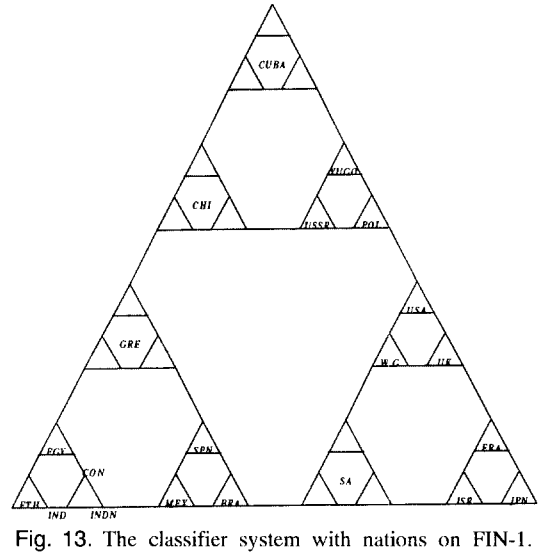

Fig. 13. The classifier system with nations on FIN-1.

according to the features which belong to each nation. The result of simulation is making satisfactory progress generally. The nations belonging to the same class have similar characters politically and economically. When the certain data have only feature values, it will be classified on FIN-1 hierarchically. The performance of the Classifier System is well when the number of sample is large. Because the above algorithm compares three distances generally, it has to be considered in detail when the number of sample is small.

5. Conclusion

GA has been reputed to be a novel search method for various fields including the optimization problem. However, the GA requires a large amount of computation to find a good solution and GA is inherently parallel that all individuals in a population evolve simultaneously. To achieve their full ability, GA must be implemented on a parallel computer architectures. In this paper, we have proposed the PGA using a multiprocessor system FIN to have a self-similarity network. At first, we have surveyed GA briefly and explained the position where the proposed methods are placed. Then, the SGA and the PGA were compared with the proposed methods on the TSP. We noted that, with alternative population structures such as grid, FIN approach performs better than the SGA, even without taking actual parallel execution into account.The power of our approach came from its topology which manages the traverse of the search space at several levels. This topology allows a rapid flow of genetic information, which is desirable in problems as good solution can rapidly propagate. Other interesting GA techniques in our methods include the hierarchical corssover and the fine-grained PGA with periodic migration. On the basis of solving the TSP, we have designed the classifier system on FIN-1. Finally, with an affinity between FIN and GA, the proposed PGA have produced valid solutions for larger problem.

References

[1] Lawrence Davis Ed., *Genetic Algorithms and Simulated Annealing*, Morgan Kaufmann Publishers, 1987.

[2] David E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison Wesley, 1989.

[3] L. Davis Ed., *Handbook of Genetic Algorithms*, Van Nostrand Rein hold, New York, 1991.

[4] Jose L. Ribeiro Filho, Philip C. Treleaven, and Cesare Alippi, "Genetic Algorithm Programming Environments", *Computer*, pp. 28-43, June 1994.

[5] M. Srinivas and Lalit M. Patnaik, "Genetic Algorithms: A Survey", *Computer*, pp. 17-26, June 1994.

[6] M. Sugaya, T. Maeba, S. Tatsumi and K. Abe, " Parallel Algorithms on a multiprocessor system with a Fractal geometry-based Interconnection Network (in Japanese)", *IEICE Trans*, Vol. J73-D-1, No. 11, pp. 847-855, 1990.

[7] Bernard Manderick and Piet Spiessens, "Fine-Grained Parallel Genetic Algorithms", in *Proc. Third Int. Conf. on Genetic Algorithms*, J. David Schaffer, Ed., pp. 428-433, 1989.

[8] Robert J. Collins and David R. Jefferson, "Selection in Massively Parallel Genetic Algorithms", in *Proc. Fourth Int. Conf. on Genetic Algorithms*, R. K. Belew and L. B. Booker, Eds., pp. 249-256, 1991.

[9] Yuval Davidor, Takeshi Yamada, and Ryohei Nakano, "The ECOlogical Framework II: Improving GA Performance at Virtually Zero Cost", in *Proc. Fifth Int. Conf. on Genetic Algorithms*, S. Forrest, Ed., pp. 171-176, 1993.

[10] Reiko Tanese, "Distributed Genetic Algorithms", in *Proc. Third Int. Conf. on Genetic Algorithms*, J. David Schaffer, Ed., pp. 434-439, 1989.

[11] H. Muhlenbein, M. Schomisch and J. Born, "The Parallel Genetic Algorithm as Function Optimizer", in *Proc. Fourth Int. Conf. on Genetic Algorithms*, R. K. Belew and L. B. Booker, Eds., pp. 271-278, 1991.

[12] V. Scott Gordon and Darrell Whitley, "Serial and Parallel Genetic Algorithms as Function Optimizers", in *Proc. Fifth Int. Conf. on Genetic Algorithms*, S. Forrest, Ed., pp. 177-183, 1993.

[13] Myung-Mook Han, Shoji Tatsumi, Yasuhiko Kitamura and Takaaki Okumoto, "Parallel genetic algorithms based on a multiprocessor system FIN and its application", in *IEICE Trans. Fundamentals*, Vol. E78-A, No. 11, pp. 1595-1605, Nov. 1995.

[14] Romney, Shephard and Nerlove, *Multidimensional Scaling*, Seminar Press, 1973.

한 명 묵(Myung-Mook Han)

received the B.E. in Ceramic Engineering from YonSei University, Seoul, Korea, in 1980, and the M.S. in Computer Science from New York Institute of Technology, New York, U.S.A., in 1987. He received Ph.D from Osaka City University, Osaka, Japan, in 1997. From 1997 to 1998 he was at YonSei University as a Post-Doc. researcher with a Korea Research Foundation Fellowship. He is currently a Full-time Lecturer in the Department of Computer Science of the College of Engineering, KyungWon University. His research interests included genetic algorithms, machine learning, parallel processing, and data mining.