Distributed Genetic Algorithm using Automated Adaptive Migration

Hyunjung Lee, Byonghwa Oh, Jihoon Yang*, and Seonho Kim

Abstract—**We present a new distributed genetic algorithm that can be used to extract useful information from distributed, large data over the network. The main idea of the proposed algorithm is to determine how many and which individuals move between subpopulations at each site adaptively. In addition, we present a method to help individuals from other subpopulations not be weeded out but adapt to the new subpopulation. We apply our distributed genetic algorithm to the feature subset selection task which has been one of the active research topics in machine learning. We used six data sets from UCI Machine Learning Repository to compare the performance of our approach with that of the single, centralized genetic algorithm. As a result, the proposed algorithm produced better performance than the single genetic algorithm in terms of the classification accuracy with the feature subsets.**

I. INTRODUCTION

Nowadays, data (in some domain) are generated continuously in different sites in the networked world. For instance, customers' bank transactions keep occurring and results huge amounts of data at each branch office. If we want to mine such data, there are two choices: *centralized* or *distributed* computation. In other words, we can either move all the data to the central site and mine the data globally, or mine the data at each site locally and combine the results. Though the former is expected to produce the best mining result over the entire data under certain criteria (e.g. training accuracy), it causes a tremendous overhead in bringing all the data to the central site, which makes the approach infeasible in many real-world applications. Even in the case the data movement is possible, the centralized approach has high chance of overfitting since all the data are considered in training. On the other hand, the latter requires only the model (or data structure) movement which is significantly less expensive than the data movement. In addition, the latter has lower chance of overfitting than the former since only the local data are considered at each site. Distributed data mining or analysis is thus of great significance [1].

The Genetic Algorithm (GA) is one of the global search heuristics inspired from biology and has been applied to a variety of optimization problems [2, 3, 4]. The standard GA runs with data at a single site, which requires the expensive data movement as described above. Moreover, GA incurs high complexity in both time and space since it requires all the data should be in memory and repeats the evolutionary computation iteratively [5]. Though the total amount of time and space in both the centralized and distributed versions of GA would be the same, the former has a serious difficulty in mining large data in a timely fashion which can be handled appropriately by the latter.

The Distributed Genetic Algorithm (DGA) is an approach to mitigate the aforementioned overhead and limitations, and to enhance the generalization capability of the centralized GA (CGA). The island model is a typical model of DGA [5, 6, 7, 8]. In this model, multiple GAs are executed in parallel each of which runs independently with its own subpopulation and allows individuals to move from one subpopulation to another. This move of an individual is called *migration* and the individual in migration is called a *migrant*. Since the size of each subpopulation is typically smaller than the population used by CGA, we expect that DGA should converge faster. In addition, the DGA is capable of mining huge data under limited time and space constraints, while CGA is not. Furthermore, DGA is expected to provide higher generalization accuracy over GA by considering only the local data at each site.

In addition to the standard parameters (e.g. crossover and mutation probabilities, selection strategies), DGA needs extra parameters for the migration policy such as the migration interval, the number of migrants, the selection and replacement methods of migrants, and the network topology of the environment [5]. The issues on the migration strategy have been studied by many researchers [7, 8, 9]. Simply, the best individual or a randomly selected one could migrate. However, if the best individual at each subpopulation is sent to another subpopulation, the global best individual may dominate in all subpopulations. This is called the conquest (domination) problem [7]. On the other hand, if migrants are chosen randomly, their possibly low fitness could make them ignored in the new subpopulation. Therefore, we need to devise an algorithm that can guarantee the avoidance of the conquest problem as well as the survival of migrants (during the minimal life span) at their new places. Also, finding the appropriate settings for the various parameters and allowing them to change dynamically (as in dynamic DGA [10]) is of importance. Against this background, we propose a new DGA that defines a set of relevant parameters on migration and automatically adjusts their values with regard to the environment.

The rest of the paper is organized as follows: Section II briefly describes the background knowledge on GA and DGA. Section III includes a detailed description on our novel DGA

The authors are with the Data Mining Research Laboratory at Sogang University, Seoul, Korea. (e-mail: luckyhj777@naver.com, mr-five@hanmail.net, yangjh@sogang.ac.kr, shkim@lex.yonsei.ac.kr)

* This work was supported by the Special Research Grant of Sogang University.

978-1-4244-2959-2/09/$25.00 ©

algorithm with automated adaptive migration and its application to the feature subset selection task. Section IV presents the experimental results on various real-world data to evaluate the performance of our approach. Section V concludes with summary and discussion of some directions for future research.

II. Distributed Genetic Algorithm

This section briefly introduces background knowledge and some of the work related to our DGA algorithm.

GA is a kind of a stochastic search method developed by Holland [2]. GA is also one of the biologically-inspired algorithms, motivated by the principle of "the survival of the fittest". Like other biologically-inspired algorithms (e.g. Ant Colony Optimization [11], Particle Swarm Optimization [12, 13]), GA is mainly applied to optimization problems. GA attempts to find the best solution through evolutionary processes of crossover and mutation among the individuals in the search space. Though effective, GA is computationally expensive due to its algorithmic characteristics requiring iterative computation of multiple candidate solutions.

Since GA is naturally appropriate to the parallel computing environment, the parallel GA has been studied by many researchers [5, 9]. One of the most widely known types of parallel GA is DGA [5, 8, 9]. In DGAs, the population can be divided into several subpopulations, among which the migrations of individuals are allowed. The basic structure of DGA is as follows:

1. Generate at random the population P of chromosomes
2. Divide P into SP_1, …, SP_{Ns} subpopulations
3. Define a neighborhood structure for SP_i, $i = 1,..,N_s$
4. For SP_i, $i = 1,..,N_s$
Execute in parallel the next steps
Apply, during f_m generations, the selection mechanism and the genetic operators.
Send n_m chromosomes to neighboring subpopulations
Receive chromosomes from neighboring subpopulations
Until the stop criterion is satisfied

As mentioned earlier, there exist parameters for migration in DGA. Some of the key parameters include:

1) Migration interval
2) Number of migrants
3) Selection strategy
4) Replacement strategy
5) Network topology

III. DGA using Automated Adaptive Migration

The objectives of our automated migration process are to maintain the genetic diversity by preventing the conquest (or premature convergence) problem while allowing the survival of migrants, and to reduce the difficulty in determining the parameters. In other words, our DGA attempts to find a good solution (with comparable and improved generalization performance over the CGA) but even faster, through the parallel computation in the subpopulations of smaller sizes and the automated migration without any user-defined parameters.

For the remaining part of this section, we start with the introduction of our algorithm. We then provide a detailed description on the automated migration process and the selection mechanism using *aging*. Finally, we present the application of our algorithm to the feature subset selection of distributed data.

A. The algorithm

First, the whole population (of CGA) is split into several subpopulations based on the network topology. (The number of subpopulations could be determined by the number of repositories of distributed data or processors, as well.) In each subpopulation, GA is performed independently. In every generation, each offspring's age is initialized to zero and increased by one whenever the individual survives in next generation.

After a fixed number of generations (i.e. migration interval), the individuals in a randomly selected subpopulation migrates to its neighbor sites (i.e. directly connected subpopulations) based on the network topology. The individuals for migration are chosen automatically according to the average quality of the subpopulations. To avoid the conquest problem and maintain diversity in subpopulations, outstanding individuals (i.e. with very high fitness) do not migrate from better subpopulation (i.e. with higher average fitness) to worse one (i.e. with lower average fitness). At the same time, in order to select individuals with reasonably good fitness and to make them contribute to the evolution, the individuals that have higher fitness (but not outstanding) than the average fitness of the subpopulation they belong migrate. (Details on the migration procedure are given in Section III-B.) The algorithm replaces individuals with the lowest fitness in the subpopulation with the immigrated, new individuals, keeping the constant size at all times. For new immigrants, their fitness (in terms of the classification accuracy against the data at the new site) can be low since they have been generated with respect to the data at the previous sites. Blind application of such fitness values may cause their extermination after the migration. In order to avoid this, aging technique is adopted in the fitness evaluation. (Details on aging are given in Section III-C.) Through this approach, each subpopulation is able to produce improved performance without losing the diversity after migration.

Above steps are iterated until the predetermined stopping criteria are satisfied. The overall flow of the algorithm is illustrated in Figure 1.

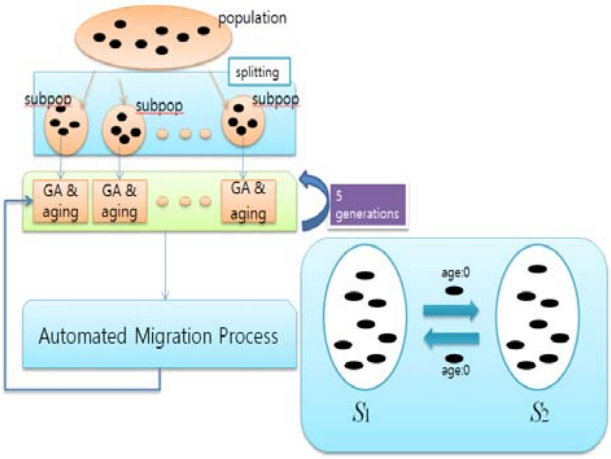


Fig. 1. The flow of our DGA algorithm

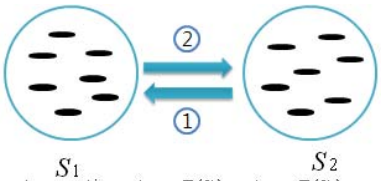
B. Automated migration process

Let S_1 and S_2 be the two directly connected subpopulations involved in migration. Let ivd_{ij} be the jth individual in subpopulation i. Then the average fitness of subpopulation *i* is

$$Avg\_F(S_i) = \frac{\sum_j F(ivd_{ij})}{|S_i|} \quad , \quad (1)$$

where $F(ivd_{ij})$ is the fitness of the ivd_{ij}.

The idea of the migration process was briefly explained in Section III-A and summarized in Figure 2. As shown in Figure 2, individuals migrate in two directions (i.e. both emigration and immigration). In Figure 2, ① is a migration from subpopulation with relatively bad quality (i.e. lower average fitness) to one with good quality (i.e. higher average fitness), and ② is the opposite case. The subpopulations with high average fitness will evolve through migration ①, and the subpopulations with low average fitness will evolve through migration ②. Since our approach performs bi-directional migrations, it enables both subpopulations to evolve. If ② is the only migration allowed, the domination problem can happen. Also, the migrants are chosen to have higher fitness than the average, which guarantees the average fitness of each subpopulation does not decrease. Note that different criteria were used to select the migrants for each direction. For ①, the individual with higher fitness than the average of the subpopulation it is joining can migrate without any restriction. For ②, however, the emigrant has higher fitness than the average fitness of the joining subpopulation but lower fitness than the average of the leaving subpopulation. This is to decrease the possibility of domination problem, while maintaining the diversity.



Assumption: $Avg\_F(S_1) > Avg\_F(S_2)$

① The direction of Migration: $S_2 -> S_1$

The Condition of migrants: $\forall ivd_{2i}, F(ivd_{2i}) \geq Avg\_F(S_1)$

② The direction of Migration: $S_1 -> S_2$

The Condition of migrants: $\forall ivd_{1i}, Avg\_F(S_2) \leq F(ivd_{1i}) \leq Avg\_F(S_1)$

Fig. 2. Automated migration process

C. Selection Mechanism using aging

Naturally, the length of the remaining life of individual human beings can be estimated by age, health, adaptability, and so on. Motivated by the natural laws, we consider two factors in terms of age for the selection process in our DGA algorithm: *biological* (or *functional*) age and *chronological* age.

The former represents the conditions of an individual in the environment where they live, such as the health and the inherent structures of genes. In our algorithm, it is replaced by the fitness of each individual for the subpopulation they belong to. To calculate the latter factor, chronological age, we consider the following four cases: selection, crossover, mutation, and migration. At first, all individual's chronological ages set to zero. If some individuals are selected and survive in the next generation, their ages increase by one. The offsprings generated by the crossover and the mutation are zero year old. Finally, when an individual emigrates to other subpopulation, its age is also initialized to zero in the new environment. This is reasonable because migrants are not removed at their original subpopulations, but copied in the other subpopulation instead. As individuals who have adaptable gene combination can live for a long time and younger individuals generally have higher chance to survive at the next year in real world, the fitter and the younger individuals in our algorithm are more probable to be selected in the evolutionary process and thus to prosper. Our algorithm defines the following selection probability:

$$SP(ivd_{ij}) = \alpha \times \frac{I - A(ivd_{ij})}{\sum_k (I - A(ivd_{ik}))} + (1-\alpha) \times \frac{F(ivd_{ij})}{\sum_k F(ivd_{ik})}, \quad (2)$$

where $SP(ivd_{ij})$ is the selection probability of ivd_{ij}, $A(ivd_{ij})$ is chronological age of ivd_{ij}, I is total number of iteration in DGA, and α is the constant weight for aging.

Through this process, even though migrants may have low fitness in the new subpopulation they join due to the different data, their young age can relieve this weakness and participate in the evolutionary process successfully.

D. Feature selection using the algorithm

We applied our new DGA algorithm to the feature subset selection problem, one of the active research topics in machine learning and data mining, in order to verify its

feasibility and to evaluate the performance. Feature subset selection is about finding the optimal subset of features, among the full features, that renders the best performance in terms of well-defined criteria such as the classification accuracy in labeled data and the total cost associated with feature subsets. Due to its intrinsic difficulty in finding the best subset among the exponential number of candidates, a variety of approaches including GA have been used [14, 15, 16, 17]. In this paper, we apply our DGA for the task. In general, the fitness of feature subsets is evaluated by learning algorithms.

Each gene is represented as a binary string encoding a feature subset [17]. If the data has N features, a gene will be an N-bit binary string. If a bit is 1, the feature is chosen in the feature subset; if 0, it is not. Figure 3 illustrates this representation of genes with ten features. Each gene in the population is a candidate feature subset.

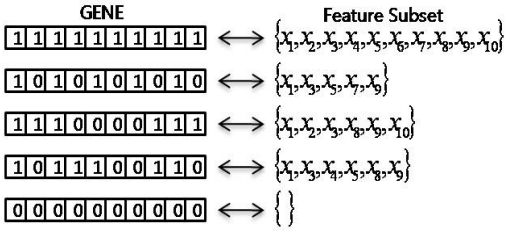


Fig. 3. Binary string representation of feature subsets

The overall flow of feature subset selection using our DGA (as a wrapper approach [15]) is depicted in Figure 4. At each site, GA is run independently. The feature subsets generated by GA are fed to the learning algorithm for fitness evaluation. For example, the learning algorithm can compute the classification accuracy of training data as the fitness for the feature subset. Based on the fitness of individuals in the subpopulation at each site, migration is used to determine the population of next generation, as described Section III-B. This process is repeated until the termination condition is met. The final output is the global best feature subset among the local best subsets produced at each site.

IV. Experiments

A. Experimental setup

The six data sets we used are from UCI machine learning archive [18]. 30% of the samples in each data set were used as test data and the rest as training data. For the distributed environment, we randomly split each training data into thirty or fifty equal-sized chunks considering the distribution of classes. The number of chunks is thus the number of subpopulations in DGA. (In experiments with networks with huge number of sites, we can use arbitrary sizes of the population and subpopulations, and arbitrary number of subpopulations in order to avoid making the population size of CGA too big.) For implementing the network topology, we constructed connected graphs. Each graph has nodes as many as the number of subpopulations. The data sets and the parameter settings used in our experiments are summarized in Table I.

As far as genetic operations, we chose simple genetic operators to generate new feature subsets with fixed, arbitrary probabilities: single-point crossover (with probability of 0.85) and bit-inversion mutation (with probabilities of 0.05).

We chose Naïve Bayes (NB) [19] as the learning algorithm to evaluate the fitness of feature subsets. Any inductive learning algorithms (e.g. C4.5 [20, 21], SVM [22]) can be used for this purpose. We simply chose NB due to its simplicity and efficiency to be employed in the wrapper for the feature subset selection task. In the fitness evaluation, 10-fold cross-validation was used to compute the average classification accuracy on training data. Test accuracy is computed with the test set, 30% of the samples that do not belong to the training data.

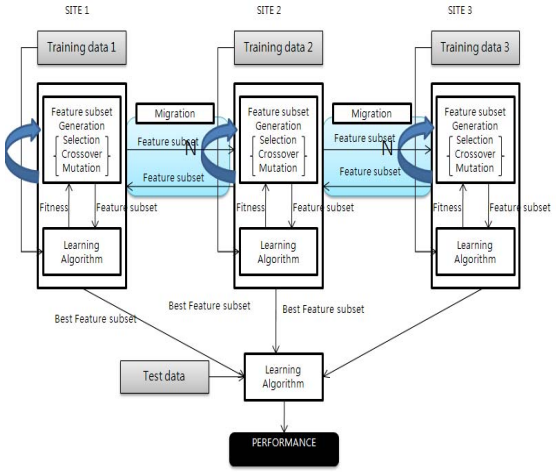


Fig. 4. Feature subset selection using DGA

TABLE I
DATASETS AND PARAMETERS USED IN EXPERIMENT

data set	# of features	# of classes	# of samples	#of subpopulations	total population size
Spambase	57	2	4601	50	500
Waveform	21	3	5000	50	500
Segment	19	7	2310	30	300
Splice	60	3	3190	50	500
Optdigits	64	10	3823	50	500
HDR Multifeature	649	10	2000	30	300

TABLE II
COMPARISON OF CLASSIFICATION ACCURACY BETWEEN CGA AND DGA

Dataset	Single(CGA)		DGA without aging			DGA using aging					
						age_weight=0.3			age_weight=0.7		
	training	test	test			test			test		
			best	average	stdev	best	average	stdev	best	average	stdev
Spambase	89.88	89.67	89.29(-)	82.68	5.10	88.85(-)	82.19	6.38	89.07(-)	83.13	**4.83**
Waveform	82.56	81.75	82.80(+)	80.78	1.26	82.34(+)	80.86	1.29	82.67(+)	81.17	**0.95**
Segment	88.84	88.42	89.14(+)	85.48	3.49	89.59(+)	85.65	3.44	89.15(+)	86.60	**1.65**
Splice	93.02	87.49	89.53(+)	86.63	2.15	90.56(+)	87.11	**1.66**	90.15(+)	86.57	2.21
Optdigits	92.17	91.17	91.52(+)	88.15	1.79	90.92(-)	87.87	2.02	90.83(-)	88.31	**1.75**
HDR Multifeature	96.46	92.67	92.51(-)	91.59	0.56	92.67(~)	91.59	0.58	92.67(~)	91.66	**0.48**

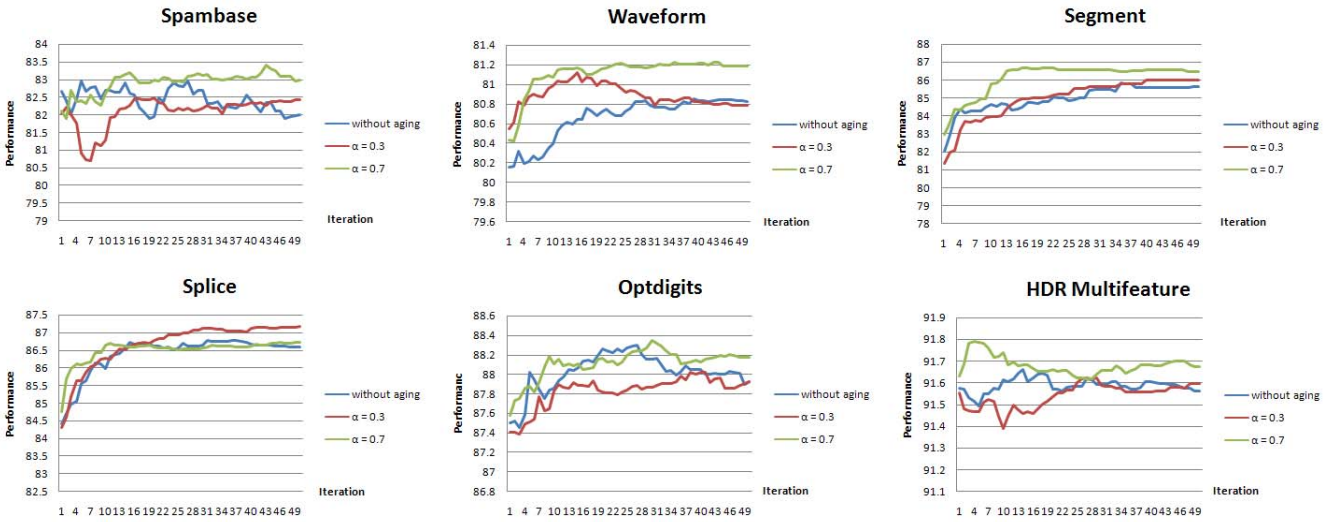


Fig. 5. Average learning curves of best feature subsets in DGA

B. *Resuls*

We applied our DGA to feature subset selection task as described in Section III-D and compared its performance with that of single GA. In both experiments, each algorithm went through 30 generations. For DGA, the migration was performed in every 5 generations.

Table II shows the performance of the CGA and DGA in terms of the classification accuracy. For CGA, both the training, test accuracies are shown. For DGA, three cases are shown. The first DGA used our automated migration but its selection probability was determined by the fitness only like the case of CGA. In the remaining two DGAs, both our migration process and the selection mechanism were used. The selection mechanism was performed with two different weights of the age: 0.3 and 0.7. In each DGA, *best* is the best of all subpopulations' accuracies, *average* is the average accuracy, and *stdev* is standard deviation of them. Most of DGAs outperformed CGA for given datasets. However, the difference between DGA and CGA is fairly small (less than 2%). Therefore, we can take advantage of DGA in its parallel computation with comparable performance as CGA. Furthermore, we noticed that all the locally best individuals produced consistent performance with small standard deviations in [0, 3.49] for all data set except the *Spambase* data. This verifies that most subpopulations evolved successfully. In terms of the difference between the training and test accuracies, there is no clear difference or tradeoffs between DGA and CGA (though the difference was either small (less than 2) or big (greater than 4) in DGA).

Figure 5 displays the average learning curves (in test accuracy) of best feature subsets in DGA for 50 iterations. We can see the accuracy of each subpopulation generally keeps increasing as learning progresses. This verifies that DGA guides subpopulations into the right direction during the evolution. We can also see that DGA using our selection process with aging with the weight of 0.7 outperforms both

DGA without aging and with aging with the weight of 0.3 for most datasets except Splice.

V. Conclusion and Future Works

The analysis of distributed, large data requires parallel algorithm that can be run on multiple processors. We proposed a novel distributed genetic algorithm (DGA) to deal with this situation. DGA makes use of automated migration which allows individuals in each subpopulation move to different sites and join the population without relying on any user-defined parameters. Our algorithm guarantees continuous improvement of fitness in all subpopulations as well as the diversity among individuals.

Performance of our approach was verified in experiments conducted with real-world datasets for feature subset selection. DGA produced comparable performance to the single genetic algorithm with even faster speed. DGA also yielded quality solutions in all subpopulations with minute deviations in performance. It was also verified to induce appropriate evolution process for its participants.

There are some avenues for future research. First, our current migration algorithm use fixed size of subpopulations, which can be extended to adjust dynamically considering the environment. As a variant, migrate-and-copy can be added to the current migrate-and-delete process. Similarly, the migration gap can be dynamic as well. The experiments in this paper were conducted with limited parameter settings (e.g. migration interval, aging weight). Extensive experiments with different parameter settings on additional data sets can help evaluating the algorithm more fairly and thoroughly. Finally, our DGA algorithm can be applied to various applications beyond the feature subset selection problem.

References

[1] H. Kargupta, and P. Chan, *Advanced in Distributed and Parallel Knowledge Discovery*, AAAI Press/MIT Press, California, 2000.

[2] J. H. Holland, *Adaptation in Natural and Artificial systems*, Ann Arbor: The University of Michigan Press, Michigan, 1975.

[3] D. E. Goldberg, *Genetic algorithms in Search*, *Optimization and Machine Learning*, Addison-Wesley Longman, Inc, Boston, 1989.

[4] D. V. Michael, *The Simple Genetic Algorithm: Foundations and Theory*, MIT Press, California, 1999.

[5] E. Alba, *Parallel Metaheuristics: A New Class of Algorithms*, John Wiley & Sons, Inc., New Jersey, 2005.

[6] D. Whitley, S. Rana, and R. B. Heckendorn, "The island model genetic algorithm: On separability, population size and convergence", *Journal of Computing and Information Technology*, 1999, Vol.7, pp 33-47.

[7] S.-C. Lin, W. F. Punch III, and E. D. Goodman, "Coarse-grain parallel genetic algorithms: Categorization and new approach," *Proceedings. 6th IEEE Parallel and Distributed Processing*, IEEE Press, New York, 1994, pp. 28–37.

[8] F. Herrera and M. Lozano, "Gradual distributed real-coded genetic algorithms", *IEEE Transactions on Evolutionary Computation*, 2000, Vol.4, No. 1, pp. 43-63.

[9] E. Cant-paz, "A survey of parallel genetic algorithm", *Calculateurs Paralleles*, 1998, Vol.10, pp. 141-171.

[10] W. Yi, Q. Liu, and Y. He, "Dynamic distributed genetic algorithm", *Proceedings of the 2000 Congress on Evolution Computation*, 2000, Vol.2, pp1132-1136.

[11] M. Dorigo and T. Stützle, *Ant Colony Optimization*, MIT Press, 2004.

[12] J. Kennedy and R. Eberhart, "Particle swarm optimization", *Proceeding soft the Conference on Neural Networks*, 1995, pp.1942-1948.

[13] J. Kennedy and R. Eberhart, R, *Swarm Intelligence*, Morgan Kaufmann, 2001.

[14] A. Jain and D. Zongker, "Feature Selection: Evaluation, Application, and Small Sample Performance", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1997, Vol. 19, No. 2, pp. 153-158.

[15] J. Yang and V. Honavar, "Feature subset selection using a genetic algorithm", *IEEE Intelligent Systems*, 1998, Vol. 13, No. 2, pp. 44-49.

[16] C. Yun and J. Yang, "Experimental comparisons of feature subset selection methods", *Proceedings of the Workshops on International Conference on Data Mining,* 2007, pp.367-372.

[17] M.J Martin-Bautista and M-A Vila, "A Survey of Genetic Feature Selection in Mining Issues", *Proceedings of the Congress on Evolutionary Computation,* 1999, Vol. 2, pp. 1314-1321.

[18] A. Asuncion and D. Newman, UCI Machine Learning Repository[http://www.ics.uci.edu/~mlearn/MLRepository.html], Irvine, CA: University of California, School of Information and Computer Science, 2007.

[19] P. Domingos and M. Pazzani, "On the optimality of the simple Bayesian classifier under zero-one loss", *Machine Learning*, 1997, Vol. 29, pp. 103-137.

[20] J. Quinlag, "Induction of decision tree", *Machine Learning*, Vol.1, No.1, 1986, pp.81-106.

[21] J. Quinlan, C4.5: Programs for Machine Learning. Morgan Kaufmann, 1993.

[22] B. Boser, I. Guyon, and V. Vapnik. "A training algorithm for optim margin classifier", *Proceedings 5th ACM Workshop on Computational Learning Theory*, ACM, 1992, pp. 144-152.