Annals of University of Craiova, Math. Comp. Sci. Ser.
Volume 33, 2006, Pages 132–140
ISSN: 1223-6934

Elitist Generational Genetic Chromodynamics as a Learning Classifier System

Catalin Stoean and Dumitru Dumitrescu

Abstract. A recently developed multimodal evolutionary algorithm [8] is used as engine for a learning classifier system. The evolutionary classifier is applied for two largely used data sets and the obtained results are compared with others of different algorithms. A noticeable gain over other computational techniques is that, besides the outcome for objects in the test set, the algorithm may also provide simple rules that led to that classification decision.

2000 Mathematics Subject Classification. 68T05, 68T20, 92D10.
Key words and phrases. evolutionary computation, learning classifier system, genetic chromodynamics.

1. Introduction

Proposed learning classifier system (LCS) is applied for the classification of two data sets that come from the UCI repository of machine learning databases [6]; one of them is *Pima Indians Diabetes Database* and the other one is the *Iris Plants Database*.

The classifier is based on a recently developed evolutionary technique, applied for solving multimodal problems, called Elitist Generational Genetic Chromodynamics (EGGC) [8]. The algorithm belongs to the family of a well-known framework, Genetic Chromodynamics (GC) [1]. GC was also used as an engine for a LCS applied for long-term placement of the elderly [4], as well as for text categorization [10]. The results proved to be very promising.

EGGC was recently applied to the optimization of several multi-dimensional functions, as well as for a classification problem and results were compared to the ones obtained by applying plain GC or other algorithms ([8], [9]).

The two data sets were chosen accordingly in order to classify different types of data: the diabetes data set contains only two classes - a patient may be either ill or healthy - while the iris data has three classes, representing three types of iris flowers: Setosa, Versicolour and Virginica, respectively. While the iris data set contains only real values for its four attributes, the diabetes data set has eight, not all real valued, attributes; exceptions are, for instance, the *age* attribute or the *number of times pregnant*.

The paper is organized as follows: section 2 introduces some basics of LCSs and section 3 presents the GC framework, the EGGC algorithm, in particular; the ground is thus laid for section 4, where EGGC is integrated into a LCS. Applications for the diabetes and iris data sets are encompassed in sections 5 and 6. Last section consists of conclusions and ideas for future work.

Received: 28 November 2005.

2. Learning Classifier Systems

A learning classifier system represents a machine learning system that uses an evolutionary algorithm as a rule discovery component ([2], [5]). The rules (or productions, which are simple if-then rules) represent a population that is evolved by an appropriate evolutionary algorithm; the rules cover the space of possible inputs and they are then evolved in order to successfully be applied to the problem to be solved - the problems may range from data mining to robotics.

There are two important families of LCSs: the Pittsburgh and Michigan approaches. In a Pittsburgh-type LCS, each individual represents an entire set of rules. The individuals compete among themselves and only the strong ones survive and reproduce. This is achieved by means of natural, proportional selection and variation operators; therefore, Pittsburgh approach uses a typical evolutionary algorithm (EA) for the learning problem. What remains to be solved is the representation problem and the way individuals adapt to their environment. Usually, in a chromosome there also appear operators from propositional logic, like disjunction and/or conjunction.

In a Michigan-style LCS, each individual of the population represents a unique, distinct rule, so the EA evolves a set of rules; the population represents the rule set needed to solve the problem. The goal here is not to obtain the best individual, but to find the best set of chromosomes (rules) in the end of the algorithm. Usually, chromosomes representation is divided into two parts - one is the condition part and contains the values for the attributes that appear in the condition of the rule and the other part consists of the conclusion of the rule. A credit assignment system is used in order to reward the better rules in a higher proportion or, at the same time, to penalise the worse rules. When new entities (rules) enter the population through mutation and/or recombination, usually crowding methods are utilised in order to introduce them; in this way, they replace only very similar individuals in the population. This technique is especially used because, in this problem case, not the best chromosome is desired most, but a set of chromosomes (rules) that, when applied, gain an optimum result for the problem to be solved.

Section 4 presents a LCS based on the Michigan approach but, instead of using a typical EA accompanied by a credit assignment system, it uses the EGGC algorithm described in section 3.

3. Genetic Chromodynamics

Analyzing the results obtained by applying GC to various problems like function optimization, clustering or classification ([1], [3], [8], [10]), it can be affirmed that it has a clearly settled place within the multimodal optimization techniques. Next subsection gives a brief introduction to the evolutionary computation.

3.1. Evolutionary Background. In order to solve a problem, an environment is created and is filled with a population of individuals. Learning is viewed as a process of continuous adaptation of an individual to the initially unknown environment. The adaptation of the individuals in the population is achieved through reproductions between them and mutations they suffer. The fitness of the individuals is closely related to how well they adapted to the environment and represents their chances of surviving and multiplying. The individuals that form the population represent a collection of candidate solutions for the problem considered. To outline the correspondences between natural evolution and problem solving, then the environment represents the problem, individuals are candidate solutions and the fitness corresponds to the quality of the solution. The quality of a candidate determines the chance that the considered individual has to be used as a seed for building new candidate solutions that will form the next generation.

Given an environment that can host only a limited number of individuals and the capacity to reproduce added to them, selection is inevitable if one does not want the population size to grow exponentially. Obviously, natural selection favours those individuals that adapt to the environment condition best - survival of the fittest - so the best ones survive and reproduce and the evolution progresses step by step. Occasional mutations take place in order to introduce new individuals to be tested. Thus, the constitution of the population changes as time passes and it evolves, offering in the end solution(s) for the problem considered.

3.2. GC Framework. GC is an evolutionary optimization technique that evolves multiple subpopulations that converge each to an optimum of the problem to be solved in the end of the algorithm.

In the beginning, the population size is high and, as the algorithm progresses, the number of solutions (chromosomes) can be reduced by each generation. In GC algorithm, each individual participates in the forming of the new generation: its mate is searched for by applying a local selection scheme; if a second chromosome is found within its local range (called the mating region), they recombine and the competition for survival of the fittest is held between the resulting offspring and the first parent only. If no chromosome can be found in the local range of the considered chromosome, it will be mutated. Consequently, it is said that GC uses a stepping-stone search mechanism in connection with a local interaction principle.

The removal of less fit solutions is performed by introducing a special operator that merges very similar chromosomes into one chromosome; this chromosome is usually considered to be the fittest one of the chromosomes to be merged.

Some changes are applied to the classical algorithm in GC framework, in order to achieve a better accuracy and, at the same time, a speed up of the convergence ([8]).

3.3. EGGC Algorithm. In this new algorithm, the selection for replacement strategy is generational; the offspring obtained after crossover does not replace any of the parents particularly, but the worst chromosome (with respect to fitness values) within its replacement radius, which is a new parameter of the algorithm. The stepping stone mechanism does not appear in this algorithm, the first parent being selected randomly. The local interaction principle and merging still hold.

A number of n chromosomes are considered for the initialization of the population; the genes values are randomly taken from their intervals.

In order to find the chromosomes from the mating region of the current one c, the distance between c and all the other chromosomes in the population is computed. The mating region of c contains all chromosomes that are at a distance from it of less than a given threshold that represents the mating radius.

One offspring is obtained from crossover between two parents. The offspring fights to enter the current generation with all chromosomes in its personal *replacement region* (Figure 1). Therefore, weak chromosomes are removed more aggressively (alongside with the effect merging has in this respect) from the current population. An important aspect of the algorithm is the choice of the replacement radius value. If picked properly, this new parameter may lead to improved convergence speed. Mutation causes only minor perturbation to a chromosome. The stop condition can refer to

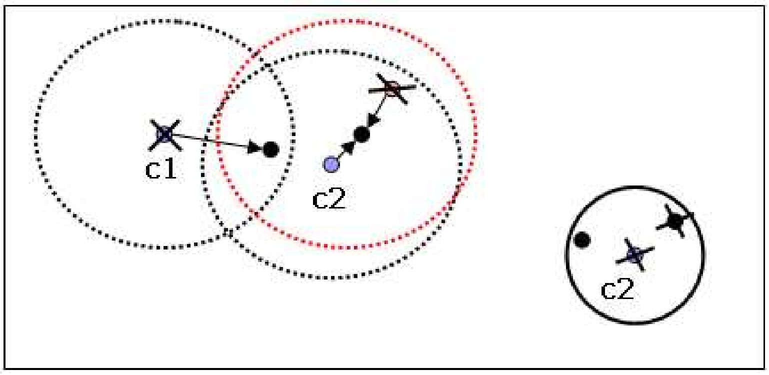

Figure 1. Mating (left) and merging (right) within EGGC. c_1 and c_2 each produce one offspring. This time, the second offspring replaces its other parent - one of the chromosomes in its replacement area (red dotted circle) - because the latter is the worst chromosome within its replacement region. During merging, two chromosomes are removed, c_2 and one offspring, because now three chromosomes are within merging radius from c_2.

a given number of generations (value of t) or to the case when there is no major improvement achieved after many generations, as in present paper.

EGGC was created for preserving the ability of GC to properly locate several or all optima within one go, and additionally speed up this process. It always accomplishes the first goal, the second one being achieved for more complex problems (for instance, for higher values for n in n-dimensional problems), due to changes that lead to better search space exploitation.

The EGGC algorithm is illustrated in Algorithm 1 ([8]).

4. EGGC Learning Classifier System

An LCS inspired from the Michigan family is presented. It is applied for two classification tasks: based upon some values patients have for their attributes, to predict whether they suffer of diabetes or not, on the one hand, and, again, based on the values of the attributes the iris flowers have, to predict their classes.

In each of the two cases, the data set containing patients or flowers, respectively, is divided into two parts - a *training set* and a *test set*. The LCS uses the objects in the training set and the values for their attributes to produce classification rules that are used in the decision-making process. The rules are finally used to predict the class for each of the objects (patient or flower, depending on the data set) in the test set.

Present model offers a easier alternative to the credit assignment system proposed by Holland in order to adequately solve the multimodal optimization problem within the Michigan approach. And proposed replacement is EGGC, since it has many times proven to be a very efficient way in determining multiple optima. Therefore, the combination between the Michigan approach and the EGGC engine seems like a good match in the domain of learning classifier systems.

Algorithm 1 EGGC Algorithm

```
$t = 0$;
initialize population P($t$);
repeat
   evaluate each chromosome;
   for $i = 1$ to $n$ do
      randomly choose a chromosome $c$;
      if mating region of $c$ is empty then
         apply mutation to $c$;
         if obtained chromosome is fitter than $c$ then
            replace $c$;
         end if
      else
         choose one chromosome from the mating region of $c$ for crossover by proportional
         selection;
         obtain and evaluate one offspring $d$;
         find worst chromosome $w$ within replacement radius of $d$;
         if $d$ has better fitness than $w$ then
            replace $w$;
         end if
      end if
   end for
   merging is applied to all chromosomes;
   $t$= $t$ + 1;
until stop condition

{Remark: The offspring $d$ obtained after crossover can replace a chromosome that does
not belong to any of the mating regions of its parents, thus performing faster separation
of chromosomes into clusters.}
```

5. Application to Diabetes Diagnosis

All patients in the data set are females of at least 21 years old, of Pima Indian heritage, living near Phoenix, Arizona, USA. There are eight attributes (either discrete or continuous) containing personal data, *e.g.* age, number of pregnancies, and medical data, *e.g.* blood pressure, body mass index, result of glucose tolerance test etc.

The last attribute is a discrete one and it offers the diagnosis, which is either 0 (negative) or 1 (positive). 34.9% of the patients in the dataset are assigned diabetes positive. The total number of cases is 768. The data is complete, according to its documentation; nevertheless, there are some zero values of attributes that were not reported as missing data, but look a bit strange. No replacement or deletion of these values was undertaken in present paper.

Each chromosome will encode an IF-THEN rule. A chromosome contains therefore nine genes, one for each attribute and one for the outcome; first eight genes are real valued, while the last is a binary one and it gives the output of the chromosome (conclusion of the rule encoded). Consequently, the condition of the rule is a conjunction of personal data and symptoms and its conclusion is the diagnosis.

The rules (chromosomes) are evolved against the training set. The fitness of a chromosome is computed as its distance to all patients in the training set that have the same outcome. The aim is to minimize distances, conceiving thus good rules for the patients diagnosis. A rule is considered of high-quality if it matches the condition part of the data in the training set with the same outcome as itself.

Having a chromosome $c = (c_1, c_2, , c_8, c_9)$ and a patient from the training set $p = (p_1, p_2, , p_8, p_9)$, the distance between c and p is computed by:

$$d(c,p) = \sum_{i=1}^{8} \frac{\mid c_i - p_i \mid}{b_i - a_i} \tag{1}$$

where a_i and b_i represent the lower and upper bounds of the i-th attribute. As the values for the eight attributes belong to different intervals, the distance measure has to refer to the interval bounds.

Convex crossover was used, with the coefficients biased by the fitness of the two parents involved. Mutation is with normal perturbation. Values for the parameters of the EGGC algorithm are specified in Table 2.

dif_i denotes the difference between the bounds of the interval corresponding to attribute i. Replacement radius was taken equal to the mating radius for both applications. The stop condition of the algorithm is given by a number of generations that can pass without any improvement for the solutions; this value was considered to be 10.

The ratio between training and test sets was set to 75%-25%, as established by Prechelt in [6] with respect to the diabetes task. Three kind of tests were conducted with different possibilities of choosing the data that would go into training and test, respectively. The two sets are obviously disjoint.

First way of splitting data is by doing test-sample cross-validation. The first 75% of the cases made the training set and the last 25% composed the test set, according to [6]. The obtained mean accuracy for the test set in 100 runs was 75.06%.

Second, another test was done according to rules of splitting that should be used for this data set, as established by Prechelt's rules in [6]. The data set is sequentially split into 75% training - 25% test to give 4 different combinations of these two sets, *i.e.* first 75% data for training – last 25% for test, first 25% data for test – last 75% for training, first 50% data for training – next 25% for test – last 25% for training again, first 25% data for training – next 25% for test – last 50% for training again. The algorithm is subject to 100 trials again. The mean accuracy obtained for the test set was 69.672%.

Last, random cross-validation was performed, *i.e.* the training set containing 75% data and the test set consisting of 25% data were randomly generated in each run. The algorithm was applied 100 times and the obtained mean accuracy was of 69.515%.

However, in many tests, it was noticed that when the chromosome pool still has four chromosomes left and has not converged yet, a higher accuracy of 80% is obtained. This leads to the idea that in the structure of each of the two obvious clusters two other subclusters are included. Thus, with the best instead of last accuracy, better results can be obtained.

Results obtained by EGGC and those obtained by other algorithms applied to the same classification problem (a neural network with Prechelt's rules in [7] and an evolved neural network in [12]) are outlined in Table 1.

6. Application to Iris Data Set

Same LCS that uses EGGC as an engine is applied to another classification problem. The Iris Plants Database contains 3 classes (3 types of iris plants), with 50

TABLE 1. Results of comparable techniques for the diabetes task

Algorithm	Runs	Accuracy (%)
EGGC with test-sample cross-validation	100	75.06
EGGC with Prechelt's rules	100	69.672
EGGC with random cross-validation	100	69.515
EGGC best accuracy (4 rules)	100	80
Neural Network (NN) with Prechelt's rules	100	65.5
Evolved NN with test-sample cross-validation	30	77.6
Evolved NN best result	30	80.7

instances for each class and 4 numerical attributes which represent length and width of the petals and sepals, respectively. The three classes are equally distributed. According to database documentation, one class is linearly separable from the other two.

There are two ways of dividing the database into training and test sets: on the one hand, two thirds of each of the three classes were considered as training set and the rest as test set; on the other hand, the training set was randomly chosen and the test set consisted of the remaining instances. In the first case, the three classes are equally distributed into training and test sets.

Chromosome representation is similar to the one from the diabetes diagnosis problem, $c = (c_1, ..., c_4, c_5)$, where the first four genes correspond to the attributes of an iris plant and the last one embodies its class. Same distance as in (1) – with four instead of eight as the upper bound of the sum – was used to compute differences between individuals and same variation operators were considered; the values that were used for the parameters of the evolutionary algorithm are given below in Table 2.

TABLE 2. Parameters of the EGGC algorithm for both classification problems

Pop size	Mut. probability	Mut. strength	Mating radius	Merging radius
100	0.4	$dif_i/100$	0.3	0.03

The final result of the LCS has to consist of at least three rules (chromosomes), that is at least one for each class. The accuracy for the first mode of fixing the training and test sets varies between 94% and 98%, while for the second way of selecting the two disjoint sets the accuracy ranges from 88% to 96%. Besides accuracy from the third row in Table 3, where the percent is the best found in 100 runs (obtained even during these runs and not necessarily at the end of them), all other values in the third column are obtained by computing the average for the final accuracies at the end of each of the 100 runs.

Unfortunately, in some literature approaches that used this database there were no clear descriptions of the way training and test sets were chosen, so direct comparisons between their results and those obtained by EGGC are not very accurate. For instance, in [13], a genetic programming model was tested on the Iris database, while in [11], some neural networks algorithms (backpropagation algorithm, denoted by BP, and cascade-correlation algorithm, CCA) were applied in this respect; results are outlined in Table 3.

Table 3. Results of different techniques for the Iris Plants Database in comparison to EGGC

Algorithm	Runs	Accuracy (%)
EGGC with equally distributed cross-validation	100	95.76
EGGC with random cross-validation	100	92.84
EGGC best accuracy instead of last	100	98
Genetic programming ([13])	100	92.7
NN BP with random cross-validation	10	93.2
NN CCA with random cross-validation	10	92.6
NN modified CCA with random cross-validation	10	97

7. Conclusions and Future Work

A multimodal evolutionary algorithm called Elitist Generational Genetic Chromodynamics has been used as an engine for a LCS; the obtained classifier system was applied to two classification problems and proved its suitability for these tasks.

A major improve may be obtained by changing the way fitness is computed for chromosomes; as all individuals represent rules, the fitness value of a certain chromosome c could be then given by:

$$f(c) = e^{\frac{x}{1+y}} \tag{2}$$

where x represents the number of objects (patients or flowers in the considered data sets) from the training set that are correctly classified by the rule (chromosome) c and y says how many objects are wrongly classified by the same rule; f is the fitness function. If c does not classify correctly any patient from the training set then its fitness is considered to be zero. At the end of the EGGC algorithm, obtained rules would be applied to the test set and accuracy computed.

This mode of computing fitness evaluation should to be more efficient as it seems more convenient that a chromosome is closer (regarding distance) to only a part of the patients from training (and consequently test) set(s) and not to all of them as computed in the proposed model. Curiously, this way of computing fitness for chromosomes did not lead to major improvements regarding accuracy on the test set; results were so far very similar to those outlined in present paper. A further improvement that is still to be done is to easily adjust the way merging takes place: it should occur only if the obtained set of rules provides a better accuracy on the training set.

References

[1] D. Dumitrescu, *Genetic Chromodynamics*, Studia Universitatis Babes-Bolyai Cluj-Napoca, Ser. Informatica, vol. 45, no. 1, pp. 39-50, 2000.

[2] D. Dumitrescu, B. Lazzerini, L.C. Jain, A. Dumitrescu, *Evolutionary Computation*, CRC Press, Boca Raton, Florida, 2000.

[3] D. Dumitrescu, Ruxandra Gorunescu, *Evolutionary clustering using adaptive prototypes*, Studia Univ. Babes - Bolyai, Informatica, Volume XL IX, Number 1, pp. 15 – 20, 2004.

[4] R. Gorunescu, P. H. Millard, *An Evolutionary Model of a Multidisciplinary Review Panel for Admission to Long-term Care*, ICCC 2004, Baile - Felix, Oradea, pp. 181 - 185, 2004.

[5] Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs*, 2nd Edition, Springer-Verlag, 1992.

[6] L. Prechelt, *Proben 1 - a set of benchmark and benchmarking rules for neural network training algorithms*. University of Karlsruhe, Institute for Program Structures and Data Organization (IPD), Tech. Rep. 21/94, 1994.

[7] R. Smithies, S. Salhi, N. Queen, *Adaptive Hybrid Learning for Neural Networks*, Neural Computation, vol. 16, no. 1, pp. 139-157, 2004.

[8] C. Stoean, M. Preuss, R. Gorunescu, D. Dumitrescu, *Elitist Generational Genetic Chromodynamics - a New Radii-Based Evolutionary Algorithm for Multimodal Optimization*, The 2005 IEEE Congress on Evolutionary Computation - CEC 2005, Edinburgh, UK, September 2-5, pp. 1839 - 1846, 2005.

[9] C. Stoean, R. Stoean, M. Preuss, D. Dumitrescu, *Diabetes Diagnosis through the Means of a Multimodal Evolutionary Algorithm*, Proceedings of the First East European Conference on Health Care Modelling and Computation - HCMC 2005, Craiova, Romania, pp. 277-289, 2005.

[10] C. Stoean, R. Gorunescu, M. Preuss, D. Dumitrescu, *An Evolutionary Learning Spam Filter System*, SYNASC 2004, Timisoara, Romania, Symbolic and Numeric Algorithms for Scientific Computing, International participation and committee, D. Petcu, V. Negru, D. Zaharie, T. Jebelean (Eds.), Mirton Press, pp. 512-522, 2004.

[11] J. Yang, V., Honavar, *Experiments with the Cascade-Correlation Algorithm*, Tehnical Report 91-16, Iowa State University, USA, 1991.

[12] X. Yao, Y. Liu, *A New Evolutionary System for Evolving Artificial Neural Networks*, IEEE Transactions on Neural Networks 8(3), pp. 694-713, 1997.

[13] C. J. Veenman, *Positional Genetic Programming: genetic algorithms with encoded tree structures*, Master's thesis, Vrije Universiteit, Amsterdam, 1996.

(Catalin Stoean) University of Craiova Faculty of Mathematics and Computer Science, Department of Computer Science Al.I. Cuza Street, No. 13, Craiova RO-200585, Romania
E-mail address: `catalin.stoean@inf.ucv.ro`

(D. Dumitrescu) Babes - Bolyai University Faculty of Mathematics and Computer Science, Department of Computer Science Str. Mihail Kogalniceanu, No. 1, Cluj-Napoca, 400084, Romania
E-mail address: `ddumitr@cs.ubbcluj.ro`