An implementation of genetic-based learning classifier system on a wet clutch system

Yu Zhong[1], Bart Wyns[1], Robain De Keyser[1], Gregory Pinte[2], and Julian Stoev[2]

[1] Department of Electrical energy, Systems and Automation, Ghent University, Ghent, Belgium (Email: Yu.Zhong & Bart.Wyns & Robain.DeKeyser@ugent.be)
[2] Flanders' Mechatronics Technology Center, Heverlee, Belgium (Email: Gregory.Pinte & Julian.Stoev@fmtc.be)

Abstract. As a methodology for search and optimization, Genetic Algorithms (GA) has now reached a mature stage. An interesting application of GA is to use it as a rule discovery system inside a Learning Classifier System (LCS), especially in a strength-based system (zero level classifier system, ZCS). An LCS is a good solution for some complex problems. But when facing multiple objectives, the conventional LCS will have difficulties in selecting classifier since the inside rule discovery system (GA) will need a certain population size for evolution diversity. In this paper, a new genetic-based LCS with integrated Non-dominated Sorting Genetic Algorithm-II (NSGA-II) is suggested to solve such problem. The LCS is implemented on a wet clutch system for achieving different engagement performance.
Keywords: Genetic algorithm, learning classifier systems, wet clutch, fuzzy clustering

1. Introduction

A learning classifier system, or LCS, is a rule-based machine learning system with close links to reinforcement learning and genetic algorithms. The first concept was described by John Holland in 1975 [1], and his LCS used a genetic algorithm to alter and select the best rules among a population of binary rules. Classifier systems are a special kind of production systems [2]. Initially the classifiers or rules were binary, but recent research has expanded this representation to include real-valued, neural network, and functional (*S*-expression) conditions.
Every learning classifier system contains an internal population of "condition-action rules", which are called classifiers. Usually the classifiers are of the form *if <condition> then <action>*. When a particular input occurs, the LCS forms a so-called match set (and denoted as [M]) of classifiers whose conditions are satisfied by that input. In general, a classifier's condition will refer to more than one of the input components, usually all of them. In a sense, the match set consists of classifiers in the population that recognize the current input. This sequence of deductions leads to the systems answer to the problem.
An LCS can improve the ability to choose the best action as well as the performance of the classifiers with experience. This adds the property of adaptability to the LCS. There are two components in an LCS to achieve such adaptability. The first one is a reinforcement learning algorithm which is similar to *Q*-Learning [3], and will operate on the action selection process. The second is a rule discovery system implemented as a genetic algorithm [1, 4] that operates on

the classifiers as a population to generate diversity in the classifier set, allowing exploration of the problem space. The overall architecture of an LCS is illustrated in Figure 1 [5].

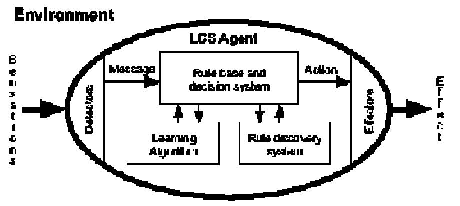


Fig. 1 Illustration of an LCS [5]

Learning classifier systems can be chartered into two types depending on where the genetic algorithm acts. A Pittsburgh-type LCS has a population of separate rule sets, where the genetic algorithm recombines and reproduces the best of these rule sets. In a Michigan-style LCS there is only a single population and the algorithm focuses on selecting the best classifiers within that rule set. Michigan-style LCSs have two main types of fitness definitions, strength-based (ZCS) and accuracy-based (XCS).

The LCS is a very good solution to some complicated systems which are very hard to model. In this paper, the LCS will be applied to a wet clutch system to achieve a good performance which is measured by engagement time and the torque loss during the engagement.

2. Background

2.1 Strength-based system (ZCS)

Among all the types of LCSs, Strength-based LCSs are the simplest ones. In a ZCS, each classifier contains only one evaluation variable which is both its estimation of the accumulated reward brought by its firing and its fitness for the population evolution process. Moreover a ZCS works with a classifier population of fixed size *P*. In forming a ZCS, there are four major processes which are: selection process, reward propagation process, evolution (discovery) process, and covering process.

The selection process for a chosen action in ZCS is based on a roulette wheel [4] mechanism after a "match-set" ([M]) is determined. The reward propagation mechanism in ZCS is close to the original Bucket Brigade algorithm [6]. The primary role of the covering mechanism is to ensure that there is at least one classifier in population that can handle the current input [7]. The covering mechanism can be implemented differently by modifying the frequency at which wild cards are added to the new rule [8-10], altering how a new rule's parameters are calculated, and expending the instances.

The most important process in ZCS, the evolution of the classifier population, is driven by a GA. At each time step, there is a probability *p* of running the GA. The GA works on the Darwinian principle of natural selection. In GA, it uses a code to present the genome which is used in the natural evolution [11, 12].Then the genetic operators (mutation operator and crossover operator) are used to generate the new individuals. Depending on the fitness function, the "better" rules will be selected more often to reproduce the off-spring. Initially, the GA needs to be started with a population of rules which can be generated randomly to broadly cover the range of possible solutions (the search space).

The main drawback of ZCS is that overly general classifiers can be sustained in the population and can result in the system taking suboptimal actions [11]. Moreover, the creation and deletion of the classifiers in ZCS are based on a global population; if this initial global population is random or not pre-optimized, then it will take very long time before the final convergence. Especially in multi-objective engineering problems, the ZCS may require a well defined weight function to transfer the multi-objective problem into a single objective problem. But most of the time, it is very difficult to properly define such weight factors.

2.2 Improved Genetic based LCS

Evolutionary algorithm is a popular approach to solve multi-objective optimization problems. Nowadays, most evolutionary optimizers apply Pareto-based ranking schemes. Genetic algorithms such as the Non-dominated Sorting Genetic Algorithm-II (NSGA-II) and Strength Pareto Evolutionary Approach 2 (SPEA-2) have become standard approaches, although some schemes based on particle swarm optimization and simulated annealing [13] are also worth of mentioning.

The improved genetic-based LCS is especially suitable for a multi-objective problem. It introduces the NSGA-II as the rule discovering mechanism, so that the optimized classifier will cover the Pareto front for a multi-objective problem. Moreover, since the required population size for a multi-objective problem cannot be very small, the number of classifiers will also be large corresponding to the population size. Meanwhile, because of the difficulty of selecting the niching parameters [4], it is very hard to make sure that the individuals will be evenly distributed along the Pareto front. This will lead to the situation that a group of classifiers will offer similar performance since they are too close to each other. Thus, after the Pareto front is formed, first the fuzzy clustering method (fuzzy C-mean method [14]) will be used to cluster the whole front into desired zones/pools, then the reduced population size genetic algorithm will generate respective individuals for each zone/pool. This will significantly reduce the number of classifiers in the system without losing the information of the Pareto front.

In the improved genetic-based LCS, the evolution of the classifiers is driven by a GA operator. For a multi-objective engineering problem, the steps to create the classifiers are:

1. To form the initial global population, the GA is employed to discover the policy space with the non-dominated sorting technique. The exploration terminates when a certain percentage of the individuals lays on the first front.
2. The approximated Pareto front is formed by ranking all the individuals involved in the first step.

3. Use the fuzzy clustering method to divide the approximated Pareto front into desired zones (each zone will be reduced into a single classifier in the next step).
4. To generate a single classifier from the individuals belonging to same zone into one classifier, the reduced population size genetic algorithm is applied until the number of the individuals in one zone decreases to a set value.

The purpose for reducing the population size with the time is to generate a small number of super performance off-springs from the pre-optimized parents group. The “pre-optimized parents group” here means the individuals located in the approximated Pareto front generated by NSGA-II, while the termination condition for NSGA-II here is not defined by the convergence or the number of generations but by the percentage of individuals within one generation laying on the first front.

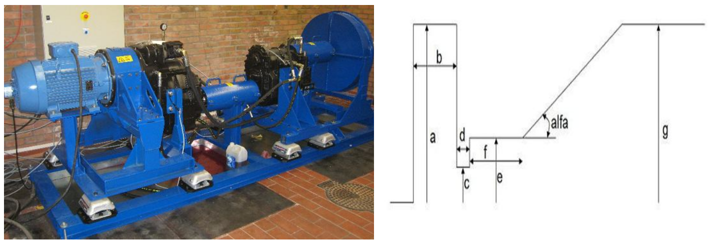
1. Assign a dummy fitness value σ to all the individuals in initial generation (the initial generation comes from the approximated Pareto front).
2. Produce n_i individuals from the current population by using a roulette wheel selection mechanism based on the individual fitness value, n_i is the number of individuals in the current generation and i is the number of generations.
3. Compare the off-spring with their parents; if both off-spring dominate its parents, delete parents; if only one of the off-spring dominate its parents, keep the dominating off-spring only; if the off-spring and their parents are non-dominant to each other, randomly delete either the parents or off-spring individual; and if the parents dominate their off-spring, delete the off-spring..
4. Do Pareto ranking, and assign the fitness value $\varepsilon^{(k-j)}\sigma$ to the individual, where k is the total number of fronts, j is the current number of front and ε is the selection pressure with $\varepsilon>1$.
5. Return to step 2 if the number of individuals left is larger than the desired value.

3. Experiments and results

3.1 Experiment Setup

The test bench is a wet clutch system driven by an electromotor. The electromotor drives a flywheel via two mechanical transmissions: one transmission is controlled in this project; the other transmission is used to vary the load and to adjust the braking torque (Fig. 2(a)).

Nowadays, wet clutches in industrial transmissions are filled using a feed forward controller of the current to the electro-hydraulic valve, which regulates the oil pressure and hence the piston position in the clutch. Fig. 3 shows a typical parameterized, feed forward current signal, which is sent in the filling phase to the valve [15]. Although nowadays more advanced feed forward signals with more tunable parameters are sometimes applied [16], the above mentioned parameterization perfectly illustrates the underlying idea behind the actual industrial control design. First, a current pulse is sent to the valve in order to generate a high pressure level in the clutch. This way, the piston will overcome the preloaded return spring and start to accelerate towards the friction plates. After this pulse, a lower constant current is sent out in order to decelerate the piston and position the piston near the friction plates. Finally a growing ramp current signal is sent to the valve such that the pressure in the clutch gradually increases and the clutch smoothly engages. The duration of the current pulse and the constant current level afterwards are critical to achieve a good filling and a smooth start of the engagement process [17]. On the one hand, a very long current pulse leads to an overfilling of the clutch such that the piston suddenly makes brutal contact with the friction plates resulting in undesired high peaks in the transmitted torque. On the other hand, a very short current pulse or a very low constant current level after the pulse leads to an under filling of the clutch, resulting in a very slow engagement. To avoid over- and under filling, in many industrial vehicles long calibration procedures are applied to find the optimal parameters of the feed forward current signal (i.e. the optimal combination of the pulse duration and constant current level) for a smooth clutch engagement. Furthermore, since the controlled system is time-varying, as described above, regular recalibrations of these parameters are inevitable.



(a) (b)

Fig. 2 (a) Wet clutch system; (b) parameterized profile

For safety reasons, experiments are performed under low external load and low breaking inertial condition in a laboratory environment. Before sending the individual profiles to test, all the individuals are first tested under no external load and no breaking inertial condition for safety check. If the reading of the torque loss under such working condition is greater than 200 *N*m* for any individual, then this one is considered as "unsafe", and this one will not be further tested. The fitness values for such individual will then be set at a very large dummy value as for rejection. If the individual is checked as "safe", it will then be taken to the low external load and low breaking inertial condition for testing, the readings for the two objectives will be treated as the fitness value for further operations. All the testing is done under a working temperature of 40°C, with a controlled range of $\pm$ 1°C.

3.2 Experiment Results

It will be very difficult to form the classifiers without comprehensive knowledge of the system like models or physical understandings. Unlike the conventional model based control techniques, the genetic based learning classifier system can start with randomized population/classifiers (Fig. 3(a)), and use evolutionary programming to explore the solution space. Thus after accumulating individuals in regions of high fitness value, the population will converge to the (or one of the) global maxim of this fitness landscape and form the optimized classifiers. For the confidential reason, the values of the results are presented in scale measurements.

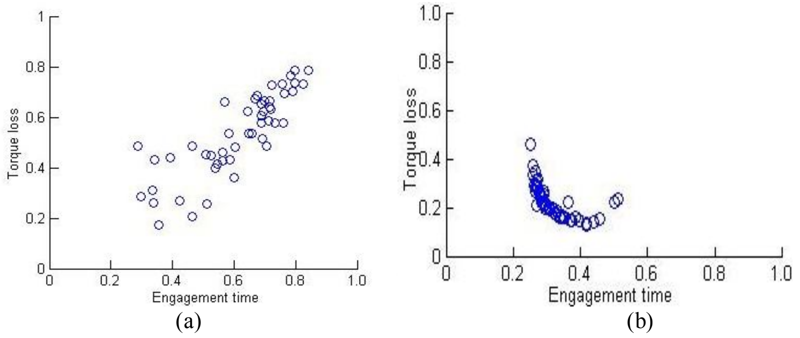


Fig. 3 (a) Initial population; (b) Final generation

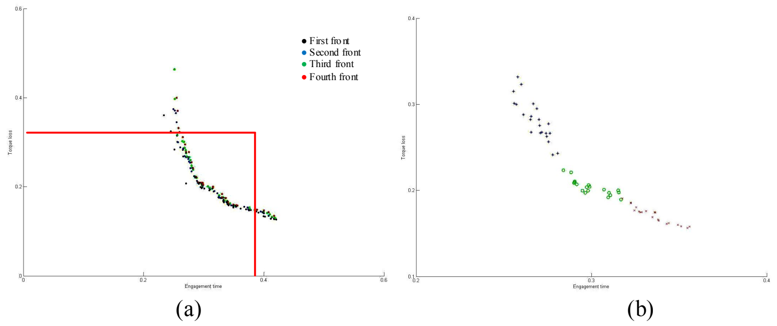
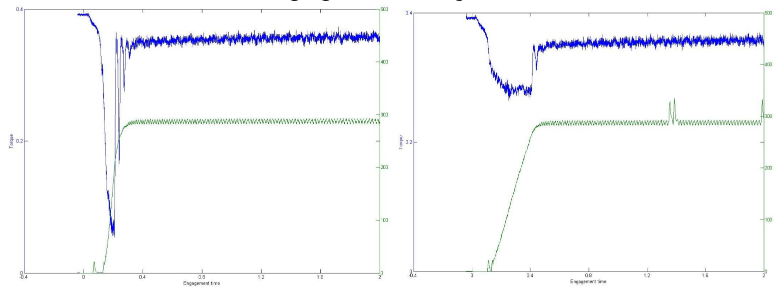


Fig. 4 (a) approximated Pareto front, (b) Pareto front after clustering

Fig. 3(b) clearly shows that most of the individuals converge to one curve, which is called "Pareto Front" for the multi-objective optimization problem. In order to figure out the Pareto front, all the individuals evaluated during the GA process are gathered together for a further sorting. The first four fronts are kept since even some identical profiles were given to the setup, the measurement can vary a little bit for every test run. Considering the test environment is under low external load and low breaking inertia, the torque loss and the engagement time should not be allowed to exceed certain limits. In case for the real working condition (i.e. heavy external load and large breaking inertia) the performance of current front can be worse. So we trim off the individuals who give a torque loss larger than 30% or an engagement time larger than 36% (outside the red line in Fig. 4(a)). Technically, the individuals remaining in the pool can be thought of as different classifiers, but for practical use, these should be further generalized.

A fuzzy clustering method (fuzzy C-means clustering) is used to identify the natural groupings of data from a large data set to produce a concise representation

of a system's behavior. In this particular setup, it is reasonable to generate three characteristic classifiers, the first one representing a fast engagement with large torque loss (in stars), the second one stands for a median engagement time with median torque loss (in cycles), and the last one is slow engagement with small torque loss (in crosses). The result from fuzzy clustering is shown in Fig. 4(b).
The reduced population size genetic algorithm mentioned before is then used to generate the divided zones/pools into a single classifier. The parameters used in this stage are $\varepsilon = 1.2$ and $\sigma = 1$. The algorithm will stop when the population size of each zone/pool is reduced to one. The results from zones 1 and 3 are shown in Fig. 5. The green lines stand for the output speed, while the blue lines present the torque loss during the engagement. We can notice that the generated classifiers inherited the properties of the zones they come from but dominate the original individuals/classifiers belonging to the zone/pool.



(a) (b)

Fig. 5 Performance of representative individual for (a) zone 1, (b) zone 3.

4. Conclusions

In this paper, Non-dominated Sorting Genetic Algorithm-II (NSGA-II) is integrated with a Strength-based learning classifier system (ZCS). This integration will extend the application of the conventional ZCS to the multi-objective problem implementation. In order to further simplify the system, a new process, called generalization process, is added to the system. In this process, the optimized individuals/classifiers are first clustered into zones/pools, and then reduced population size genetic algorithm is used to simplify the information into a small number of individuals/classifiers. This can guarantee that the useful information, i.e. the Pareto front; in the multi-objective problem will not be affected.
The proposed algorithm is then verified on a wet clutch system. In this wet clutch system, the engagement speed and the torque loss occurring during the engagement are two objectives to be minimized. It is very hard to define a proper weight factor to reduce this system into a single objective problem. It is then essential to form a classifier system to let the user determine the desired performance as the system input. The proposed algorithm works well on the wet clutch system; it first constructs the whole Pareto front; then generates three representative classifiers which corresponding to three different performances.

Acknowledgements

This work has been carried out within the framework of the LeCoPro project (grant nr. 80032) of the Institute for the Promotion of Innovation through Science and Technology in Flanders (IWT-Vlaanderen).

References

[1] J.H., Holland, "Adaptation in Natural and Artificial Systems". *University Press of Michigan*, Ann Arbor, 1975.

[2] R. Davis and J. King, "An Overview of Production Systems". *Machine Intelligence*, volume 8, pp. 300-332. New York, 1976.

[3] L. Kaelbling, M. Littman, and A. Moore, "Reinforcement Learning: A Survey". *Journal of Artificial Intelligence Research*, volume 4, pp. 237-285, May 1996.

[4] D. E. Goldberg, "Genetic Algorithms in Search, Optimization, and Machine Learning". *Addison-Wesley Publishing Company, Inc.*, Reading, MA, January 1989.

[5] A. Robert, "EMuds: Adaptation in Text-based Virtual World". *PhD Thesis*, Faculty of Science, University of Fribourg, Switzerland, 2000.

[6] J.H. Holland, "Escaping brittleness: The possibilities of general-purpose learning algorithms applied to parallel rule-based systems". *Machine Learning*, *An Artificial Intelligence Approach* (*volume II*). Morgan Kaufmann, 1986.

[7] S. W. Wilson, "Knowledge growth in an artificial animal," *Proceedings of the 1st International Conference on Genetic Algorithms and Their Application*, pp. 16–23, 1985.

[8] S. W. Wilson, "ZCS: a zeroth level classifier system," *Evolutionary Computation*, vol. 2, no. 1, pp. 1–18, 1994.

[9] S. Smith, "A learning system based on genetic adaptive algorithms", *Ph.D. thesis*, University of Pittsburgh, Pittsburgh, Pa, USA, 1980.

[10] S. W. Wilson, "Classifier systems and the animat problem," *Machine Learning*, volume 2(3), pp. 199–228, 1987.

[11] O. Sigaud and S. Wilson, "Learning classifier systems: a survey," *Soft Computing*, volume 11(11), pp. 1065–1078, 2007

[12] J. Holmes, "Learning classifier systems applied to knowledge discovery in clinical research databases," *Learning Classifier Systems, from Foundations to Applications*, pp. 243–262, 2000

[13] B. Suman and P. Kumar, "A survey of simulated annealing as a tool for single and multiobjective optimization". *Journal of the Operational Research Society*, volume 57(10), pp.1143-1160, 2006.

[14] J.C. Bezdek, "Pattern Recognition with Fuzzy Objective Function Algorithms", ISBN 0306406713, 1981.

[15] K.V. Hebbale, C.-K. Kao, D.E. McCulloch, "Adaptive electronic control of power-on upshifting in an autamatic transmission". *US Patent No. 5,282,401*, 1994.

[16] J.R. Hillman, D.P. Simon, "Method for determining the fill time of a transmission clutch". *US Patent No. 6,216,074*, 2001.

[17] Z. Sun, K. Hebbale, "Challenges and Opportunities in Automotive Transmission Control". *Proceedings of the 2005 American Control Conference*, pp. 3284-3289, Portland, OR, USA, 2005.