Design of Self-organizing Systems Using Evolutionary Methods

István Fehérvári and Wilfried Elmenreich (Faculty Mentor)
Institute for Networked and Embedded Systems / Mobile Systems Group
University of Klagenfurt
Klagenfurt, Austria
Email: {ifeherva,wilfried.elmenreich}@uni-klu.ac.at

Abstract — *Having many advantages, self-organizing systems could mean a solution for intelligent group behavior. Until now there is no general methodolgy how to design and control these systems. In this paper we examine a genetic approach for generating group behavior. Related research proved that it is possible to create sensor based robot control program using genetically evolved ANNs. Extending this concept, we propose a method for designing self-organizing systems by extending this concept to co-operative behavior among the individuals.*

I. Introduction

The power of evolutionary methods was shown in many applications where the implementation of artificial intelligence was necessary. As an example genetic algorithms (and so genetic programming) were used in simple and more complex games as proposed by Hauptman and Sipper [1]. Another approach is using neural networks to build programs capable of learning. Following this idea adaptive robot control can be developed at the hardware level for practical motion [2] or taking one step further by evolving neural networks in a genetic manner to get a complete control program of a single robot driven by sensor data. According to [3] the evolved network was significant better than the handcrafted solution.

For self-organizing systems to get the highest satisfaction σ_{sys} for the system a cooperative behavior should be developed. This is mostly done by manually adjusting parameters which in case of systems with high complexity could be very difficult. In these systems, a small change in a parameter could arise unpredicted behavior due to the missing direct function between element and system. Thus, a lot of trials are necessary to find a satisfying solution. Evolutionary methods provide a means to automatize the testing and optimizing of parameters in an intelligent way.

In this paper we present a method where the team strategy is not pre-programmed, but using the idea proposed by Elmenreich and Klinger [3], is generated by genetically evolving an Artifical Neural Network (ANN). The intention is to minimize the required human interference in creating the model by making it partly or fully generated by itself. The rest of the paper is structurized as follows: Section 2 describes self-organizing systems while Section 3 explains the genetic approach and discusses the method for the given problem. Current work is discussed in Section 4.

II. Self-organizing Systems

The term "self-organizing" was first introduced in 1947 by W. Ross Ashby. It refers to pattern-formation processes within a system caused by the behavior of its individual entities. Their basic feature is the way how they acquire their order and structure. Among the several definitions the most suitable could be the following:

"*In self-organizing systems, pattern formation occurs through interactions internal to the system, without intervention by external directing influences.*"[4, p.7]

A good example is a team of workers acting on their own following a mutual understanding. If there would be common blueprints or a boss controlling them, it would mean an external influence resulting in no self-organization. Nature also has many examples like a school of fish where each individual has knowledge only about its neighbors and so there is no leader among them. The key part is the communication between the individuals satisfying their own goals and by doing so a pattern emerges. Through these features it is expected that the overall system shows many advantages like robustness, adaptability and scalability which makes self-organizing systems (SOS) an interesting option for controlling complex systems. However the above mentioned properties and the decentralized control makes an SOS difficult to design and control. Although, Gershenson [5] provides some ideas for the design of SOS for technical applications, there is no general methodology yet to explain how these should be done.

III. The Genetic Approach

The main idea behind genetic methods is the Darwinian selection process. The algorithm operates on a pool of possible solutions. It evaluates them based on their fitness function which measures their performance when applied to the given problem. While the best ones are kept, those with bad performance will be replaced by offsprings or mutations of the pool. For more complex problems where the solutions cannot be represented by variables but individual programs it is hard to define how to mutate or recombine them while keeping the syntactically correct structure. In these cases normal programming languages like C or Java do not qualify for this kind of programming, however the usage of ANN could solve the problem.

A. Neural network model

We choose a fully connected, time-discrete, recurrent ANN where each neuron has a connection to every other neuron and also to itself via several input connectors. Based on the neurons activation value, the neurons output is forwarded via different connections to all other neurons. Each connection is assigned a weight and each neuron is assigned a bias value. There are special input and output neurons for the controlling interface: the input neurons produce the sensor data on their output independtly of their input values, the output neurons act like normal neurons in the network, but their output data is also forwarded to the actuators. Other nodes which are not characterized as input or output nodes are the so-called hidden neurons. Figure 1 shows a schematic structure.

In more complex systems containing many individuals like self-organizing systems there might be a need for dif-

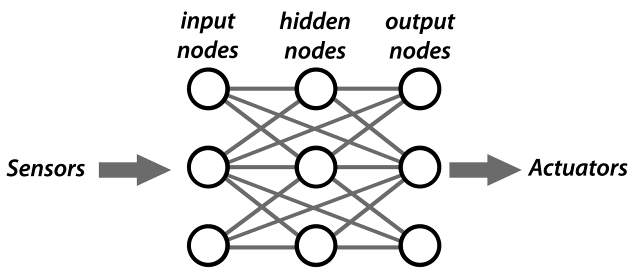

Figure 1: Neural network model example

ferently arranged sensor subsystems connected to the same network. Robot soccer would be a good example where one group of sensors watch for the ball while other groups for teammates or opposing partymembers. Their separate inputs will be connected to the common network at the input nodes so the output would depend on several external factors resulting in more compound behavior.

B. Evolution method

Evolution method means a genetic algorithm that looks for solutions with high fitness values to the given problem. During the process many solutions are created and evaluated in parallel while the best ones will be kept for the next generation. First, the genomes have to be defined for each entity. In our case they are represented by the weight matrix and the biases of each neuron. The next step is the definition of the fitness function which is a proper formulation of what we expect from the system. As proved in [3] genetic selection processes supporting crossover, mutation and elite selection have higher learning speed than the ones without crossover [6]. To have support of all this features we will use the versatile framework given by Pfandler [7].

IV. Testbed

There are several ways to test the proposed method; we chose the RoboCup Soccer Simulator[1] as an evaluation tool. This open-source program is capable of simulating a soccer game between two teams played in real-time. See Figure 2 for a screenshot of the simulator. As a result of each genetic procedure a population of ANNs will be created. These will be turned into robot soccer team control programs, where we want to rank them for further processes based on their in-game performance. In the soccer environment there is no effective way to assign fitness values to the entities by evaluating them one by one, so we will use a Swiss system tournament [8] organized among them. A standard simulated robot soccer game takes 12 minutes to finish if played in real-time. Calculating with a population of 100 it would take 84 minutes to finish with only one iteration. To overcome this problem and shorten the required time for the test the simulator will be modified to run as fast as possible using an asynchronic coupling between server and team clients. Further speedup can be achieved using a parallel evaluation on a multi-computer system.

V. Conclusion and Outlook

The success of genetically evolved ANNs used in [3] for autonomous mobile robot control shows a good potential

[1]http://sserver.sourceforge.net

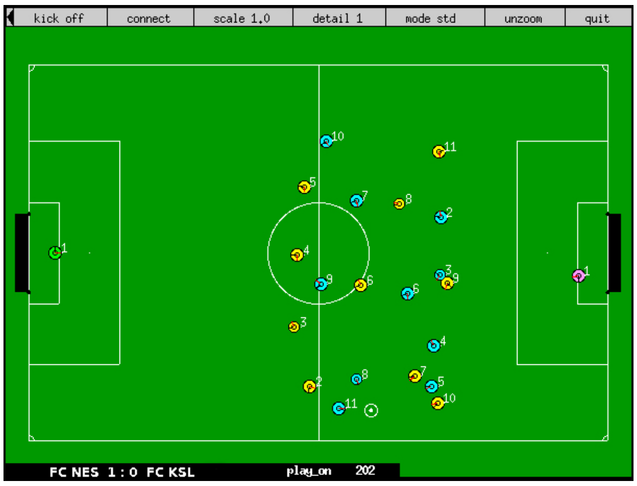

Figure 2: Team evaluation using soccer simulator

for applying to self-organizing systems. The goal of the current project is to create a simulation environment for self-organizing systems evolved by following the proposed method. Once implemented, experiments will be performed to optimize and to compare this solution with hand coded ones in robot soccer environments.

Acknowledgments

This work was supported by the Austrian FWF project TTCAR under contract No. P18060-N04.

References

[1] M. Sipper, Y. Azaria, A. Hauptman, and Y. Shichel. Designing an evolutionary strategizing machine for game playing and beyond. *IEEE Trans. on Systems, Man and Cybernetics, Part C*, 37(4):583–593, July 2007.

[2] M. W. Han and P. Kopacek. Neural networks for the control of soccer robots. In *Proceedings of the 2000 IEEE International Symposium on Industrial Electronics*, pages 571–575, Cholula, Mexico, August 2000.

[3] W. Elmenreich and G. Klingler. Genetic evolution of a neural network for the autonomous control of a four-wheeled robot. In *Sixth Mexican International Conference on Artificial Intelligence (MICAI'07)*, Aguascalientes, Mexico, November 2007.

[4] S. Camazine, J. Deneubourg, N. R. Franks, J. Sneyd, G. Theraulaz, and E. Bonabeau. *Self-Organization in Biological Systems*, volume 1. Princeton University Press, Princeton, NJ, USA, 2nd edition, January 2001.

[5] C. Gershenson. Design and control of self-organizing systems. PhD Dissertation, Vrije Universiteit Brussel, Brussel, Belgium, 2007.

[6] L. A. Meeden. An incremental approach to developing intelligent neural network controllers for robots. *IEEE Trans. on Systems, Man, and Cybernetics, Part B: Cybernetics*, 26(3):474–485, June 1996.

[7] A. Pfandler. Design and implementation of a generic framework for genetic optimization of neural networks. Bachelor's thesis, Vienna University of Technology, Vienna, Austria, 2007.

[8] J. A. Bergstra and L. M. G. Feijs, editors. *Algebraic Methods*, volume 2. Springer-Verlag, New York, LLC, 1st edition, May 1991.