

Meta-Learning Genetic Programming

Ryan J. Meuth
University of Advancing Technology
2625 W Baseline Rd.
Tempe, AZ, 85283
1-(636)-578-4171
rmeuth@ieee.org

ABSTRACT

In computational intelligence, the term ‘memetic algorithm’ has come to be associated with the algorithmic pairing of a global search method with a local search method. In a sociological context, a ‘meme’ has been loosely defined as a unit of cultural information, the social analog of genes for individuals. Both of these definitions are inadequate, as ‘memetic algorithm’ is too specific, and ultimately a misnomer, as much as a ‘meme’ is defined too generally to be of scientific use. In this paper, we extend the notion of memes from a computational viewpoint and explore the purpose, definitions, design guidelines and architecture for effective memetic computing. Utilizing two genetic programming test-beds (the even-parity problem and the Pac-Man video game), we demonstrate the power of high-order meme-based learning, known as meta-learning. With applications ranging from cognitive science to machine learning, meta-learning has the potential to provide much-needed stimulation to the field of computational intelligence by providing a framework for higher order learning.

Categories and Subject Descriptors

I.2.2 [Artificial Intelligence]: Automatic Programming – *program synthesis, program verification.*

General Terms

Algorithms, Design, Theory.

Keywords

Genetic Programming, even parity, pac-man, meta-learning, memetic algorithms, late breaking abstract.

1. INTRODUCTION

One of the major drawbacks of evolutionary algorithms and computational intelligence methods in general is that the solvers employed usually start from zero information, or utilize random initial states, independent of how similar the problem instance is to other instances the method has been applied to in the past. In effect, the optimization methods typically do not incorporate any mechanisms to establish inter-instance memory.

In effect, the optimization methods typically do not incorporate any mechanisms to establish inter-instance memory. Parameter recommendations and user-seeded known-good initial evolutionary algorithm populations provide some inter-instance information, though this knowledge is provided by the operator, and gained through human experience. This random initialization property is useful for comparing different computational intelligence methods and in some cases, particularly when computation time is not an issue, is desirable as it allows the search to be more focused, thus leading to solutions that would not otherwise have been found efficiently. It is also worth noting that many real-world problem domains are composed of sub-problems that can be solved individually, and combined (often in a non-trivial way) to provide a solution for the larger problem [1, 2].

In some problem instances, such as large instances of the even parity problem, it is nearly impossible to stochastically arrive at a complete solution without utilizing generalized solutions for small instances of the problem [3]. It is simple to evolve a function that performs even parity on 2 bits using only the logical functions AND, OR and NOT as primitives, but extremely difficult to evolve a 10-bit even parity function without any *a priori* information as the space of all possible solutions is immensely larger, and even the best known solution is complex. By simply defining the general 2-bit XOR function (the even parity computation for 2 bits), the optimization method has a higher probability of combining instances of XOR to arrive at an n -bit even-parity function, greatly accelerating the optimization process.

Both Darwinian evolution and memetics have been sources of inspiration for classes of algorithms for problem-solving techniques with memetic algorithms being the most prominent and direct manifestation of the inspiration. In recent years, there has been a marked increase in research interests and activities in the field of Memetic Algorithms (MA). The first generation of MA refers to hybrid algorithms, the combination of population-based global search (often in the form of an evolutionary algorithm) with a cultural evolutionary stage. The first generation of MA, though it encompasses characteristics of cultural evolution (in the form of local refinement) in the search cycle, may not qualify as a true evolving system according to Universal Darwinism, since all the core principles of inheritance/memetic transmission, variation and selection are missing [4, 5]. This suggests why the term MA stirred up criticisms and controversies among researchers when first introduced [6].

2. META-LEARNING GENETIC PROGRAMMING

A meta-learning system should be composed of four primary components – an optimizer, a memory, a selection mechanism, and a generalization mechanism, shown in Figure 1. The selection mechanism takes the features of a given problem as input, and performs a mapping to solutions in the memory that have an expected high quality. The memory stores previous or generalized solutions encountered by the system, and passes selected solution(s) on to the optimizer. The optimizer performs specialization and modification of solutions to optimize a given specific problem instance, while the generalization mechanism compares the resultant solution with existing solutions in memory, and either adds a new solution or modifies an existing solution. In memetic computation terms, the optimizer generates schema or modifies memes into schema, and then the generalization mechanism converts the schema back into memes for storage in memory. The selection mechanism provides a mapping on memes, providing recognition from a problem specification to a likely useful general solution, effectively utilizing internally represented meta-memes.

With these components, the architecture should be capable of exploiting information gained in previous problem sessions towards the solution of problems of increasing complexity. Integrating a cross-instance memory and a selection mechanism with an optimization method allows the recognition of a situation and the selection of previously utilized schema as likely high quality solution candidates. The optimization process then combines and refines these solution candidates to provide a good solution much faster than if the method had only random initial solutions. Once the solution is deployed, the selection method is trained to associate the situation (stimulus) with the solution (behavior) utilizing the fitness (reward) of the solution[7].

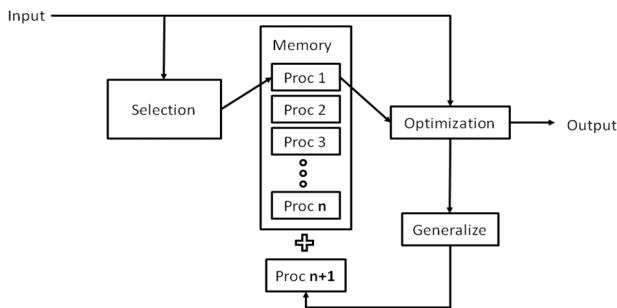


Figure 1. Meta-Learning Architecture

This architecture is implemented using Gram-ART, a new Adaptive Resonance Theory variant, which is capable of clustering variable dimension semantic inputs by creating templates that store a non-parametric distribution over the symbols and structure of a given grammar. This method serves as generalization and memory mechanism for a genetic-programming optimizer.

A GP Meta-Learning system is constructed by augmenting the Automatic Function Definition GP with a neural network method

that is trained to map between a parametric description of a given task and the function-categories created by the Gram-ART method. The output of this mapping is used to probabilistically bias the use of functions in the initial generation of the GP process. By seeding the population with genetic information that has been useful in similar situations in the past, it is expected that the GP will be able to more quickly find a high-quality solution. If a high-quality solution is not found, the exploration/exploitation feedback mechanism will drive the system towards new solutions, which will be incorporated into the function library at the end of training.

To demonstrate the principles and advantages of meta-learning, its application to the even and odd parity problems, standard benchmarks for GP and automatic function definition methods [8] are examined. Additionally the meta-learning architecture is evaluated on the game of Pac-Man. The game of Pac-Man is a standard benchmark for the study of evolution of autonomous agents in changing environments. The Pac-Man scenario allows the demonstration of behaviors such as task-prioritization (eating dots vs. avoiding ghosts), adaptability, and robustness[9].

The results show that the addition of memory, and the training and integration of separately learned skills can significantly increase the fitness of evolved individuals for even-parity function approximation, and playing the game of PAC-MAN.

3. REFERENCES

- [1] D. Shahaf and E. Amir, "Towards a Theory of AI Completeness," *8th International Symposium on Logic Formalizations of Commonsense Reasoning*, 2007.
- [2] D. Lenat and R. V. Guha, *Building Large Knowledge-Based Systems*: Addison-Wesley, 1989.
- [3] J. R. Koza, "Hierarchical genetic algorithms operating on populations of computer programs," in *International Joint Conference on Artificial Intelligence*, 1989, pp. 768-774.
- [4] D. Dennett, *Darwin's Dangerous Idea*. New York: Touchstone Press, 2005.
- [5] Y. S. Ong and A. J. Keane, "Meta-Lamarckian learning in memetic algorithms," *IEEE Transactions on Evolutionary Computation*, vol. 8, pp. 99--110, 2004.
- [6] P. Moscato, "On evolution, search, optimization, genetic algorithms and martial arts: Towards memetic algorithms," Caltech Concurrent Computation Program, C3P Report, 826. 1989.
- [7] R. J. Meuth, M.-H. Lim, Y.-S. Ong, and D. C. Wunsch, "A Proposition on Memes and Meta-Memes in Computing for Higher-Order Learning," *Journal of Memetic Computing*, vol. 1, 2009.
- [8] J. R. Koza, "Hierarchical Automatic Function Definition in Genetic Programming," in *Foundations of Genetic Algorithms 2*: Morgan Kaufmann, 1992, pp. 297-318.
- [9] J. R. Koza, "Evolution and co-evolution of computer programs to control independent-acting agents," in *From Animals to Animats: Proceedings of the First International Conference on Simulation of Adaptive Behavior*, 1991.