Modelling Learning in Plants Using Standard Genetic Algorithm (SGA)

Aparna Gopalakrishnan, Katharina Braeutigam

Department of Mathematical and Computational Sciences and Biology, University of Toronto Mississauga

Abstract

Modelling different phenomena such as the Baldwin effect[1] and Lamarckian learning are useful in evolutionary algorithms to solve a wide array of optimization problems. The purpose of this project was to model the Baldwin effect in plants. Through this project, we were able to confirm results regarding the effect of mutation rate and selection methods on efficiency of convergence in a small population.

Introduction

The interaction between evolution and learning has been studied from a molecular perspective[2] as well as from a computational one to understand the processes, benefits, and costs involved. This interaction is relevant in the field of artificial intelligence as it improves our understanding of efficiency in adaptation in dynamic environments. Classical models implement genetic changes, however, behaviour such as learning can potentially influence fitness, survival, and thus, evolution.

Objective

The aim of this project was to model the Baldwin effect in plants, as well as to explore different plant-specific parameters of reproduction like self-reproduction and the lack of decision and classification problem for plants.

Methods

- Articles published by Ellefsen[3] and Suzuki and Arita[4] provided as starting material:
  - Ellefsen modelled the Baldwin effect using Reinforcement Learning (RL) and used Downing's Script-Based Approach to Evolving Neural Networks (SEVANN) system[5] to implement a neural network.
  - Suzuki and Arita used the Iterated Prisoner's Dilemma (IPD) and the Meta-Pavlov learning method to compare strategies adopted in globally-interacting and locally-interacting agents to understand the impact of the scale of interaction in the emergence of the Baldwin Effect.
- The main challenge in the project was converting a classification problem experiment designed for individuals resembling animals to a non-classification, non-decision problem experiment for individuals resembling plants, due to the lack of availability of such clear-cut classification and decision problems in the context of learning in plants.

Methods

- Options discussed:
  - Defence-priming in plants i.e. memory (plasticity) of plants after, say, an insect attack over a lifetime versus over multiple generations
  - Impact on photosynthetic rate and/or transpiration rate in response to a change in heat and moisture[6]

  Both these options required modelling of the Baldwin Effect using a specific mathematical model for the respective plant processes (if available), resulting in a loss of abstraction.

Abstraction of learning through the following models was explored:

- Own implementation of a Standard Genetic Algorithm (SGA)
- NEAT-Python (NeuroEvolution of Augmenting Topologies)[7]
  (Was not chosen due to the incompatibility of the nature of the problem and the parameters offered by NEAT-Python)
- DEAP (Distributed Evolutionary Algorithms in Python)[8]

Description of Model

- Individual phenotype –
  - 4, 8, and 16 bits (real values in [-1, 1])
- Population size – 20, 30, 50, 100, and 150 individuals
- Fitness function (graphs given below) –
  - Version 1 – Sigmoid (Fig. 1) for benefit and Rectified Linear Unit (ReLu) (Fig. 2) for cost
  - Version 2 – Sigmoid

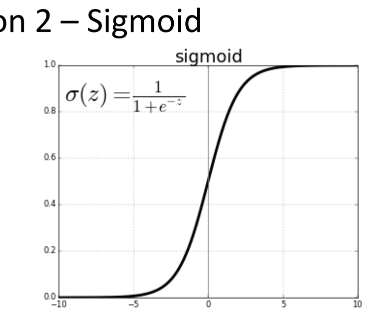

Fig. 1: Graph of sigmoid function

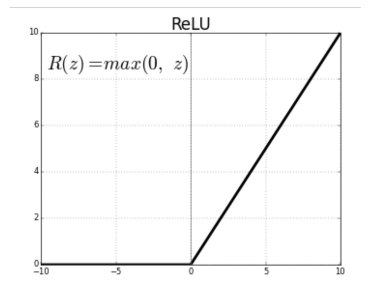

Fig. 2: Graph of ReLu function

- Parent population size (as % of population size) and tournament size (as % of parent population size) – 10%, 20%, 40%, 50%, 60%, 80%, 100%
- Mutation rate – 0.1%, 1%, and 5%
- Selection methods[9, 10] –
  - Linear Rank Selection (LRS)
  - Exponential Rank Selection (ERS)
  - Tournament Selection (TOS)
- Number of generations – 50, 100, 300, 1000, 3000, 10000
- Number of iterations – 50
- Convergence criterion – threshold (average fitness)

Observations

- Combinations of low (0-10%), moderate (20-50%), and high (60-100%) parent population size and tournament size were tested to compare the effect of each combination on the loss of diversity in population, and consequently, the rate of convergence of population.
- With the combination of higher parent population size and tournament size percentages (60-100%), quick convergence is due to the best individual having a higher probability of being chosen (seen by an initial sharp increase in average fitness) for crossover leading to quick loss in diversity in the population and convergence towards local optimum.
- The observations have been summarized in the table below:

Table 1: Observations on convergence, averaged over 50 iterations

Parameter choices			Observations (on convergence)	Number of generations to converge
PH	P, T (%)	M (%)		
4, 8, 16	10, 10	0.1, 1, 5	• No or inefficient convergence • Decrease in average fitness • Observed in Fig. 3	Not applicable
4, 8	20-50, 20-50	0.1	• Convergence to local optima due to low mutation rate	200-2000
4, 8	60-100, 60-100	0.1	Quicker convergence in population of 4-bit individuals	250-600
4, 8	20-50, 20-50	1	• Quick convergence • Observed in Fig. 4 (PH=4), Fig. 5 (PH=5)	30-250
4, 8	60-100, 60-100	1	Quicker convergence in population of 4-bit individuals	10-50
4, 8	20-50, 20-50	5	• Quicker convergence in population of 4-bit individuals • Best individual outperforms population in many generations	10-200
4, 8	60-100, 60-100	5	• Very quick convergence • Sharp increase in average fitness in the first 10-15 generations	10-20
16	20-50, 20-50	0.1	Slower rate of increase in average fitness	1500-3000
16	60-100, 60-100	0.1	Quicker convergence	600-2000
16	20-50, 20-50	1	• Slower rate of increase in average fitness for lower tournament size • Observed in Fig. 6	100-600
16	60-100, 60-100	1	Quicker convergence	100-200
16	20-50, 20-50	5	Quicker convergence	20-200
16	60-100, 60-100	5	• Sharp increase in average fitness within the first 15-20 generations • Very quick convergence	20-100

PH = Phenotype length (in bit-length), P = Parent population size percentage, T = Tournament size percentage, M = Mutation rate

- Moderate mutation rate, parent population, and tournament size percentages (20-40%) allow for slower convergence, but avoid premature convergence toward the local optima.
- Extreme mutation rates are not suggested, as very low mutation rates allow for minimal exploration of the search space, and very high mutation rates result in inefficient convergence.

Observations

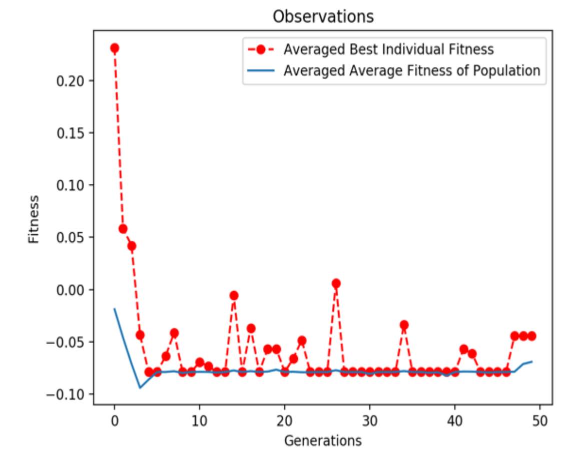

Fig. 3: Observations for PH=8, P=10%, T=10%, M=0.1%, population size=50

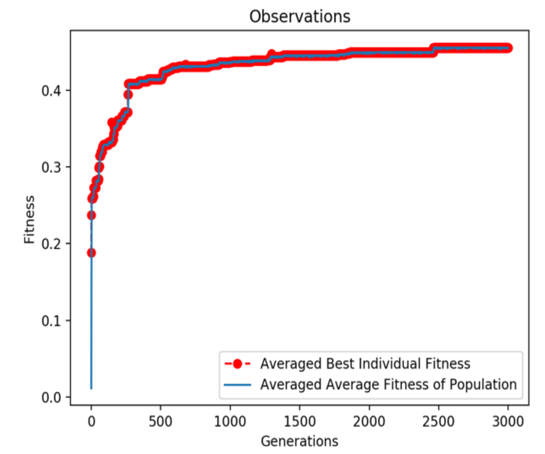

Fig. 4: Observations for PH=4, P=40%, T=40%, M=1%, population size=50

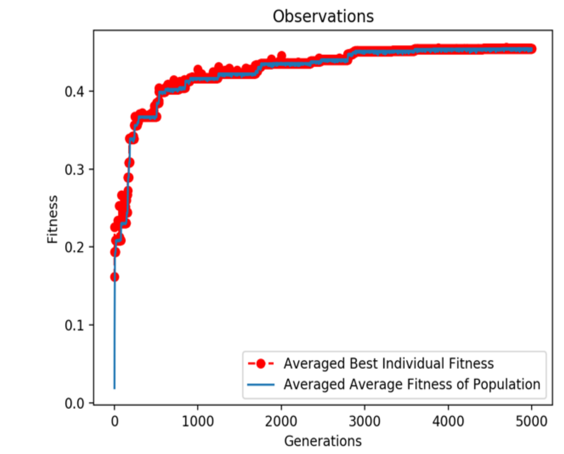

Fig. 5: Observations for PH=8, P=20%, T=20%, M=1%, population size=50

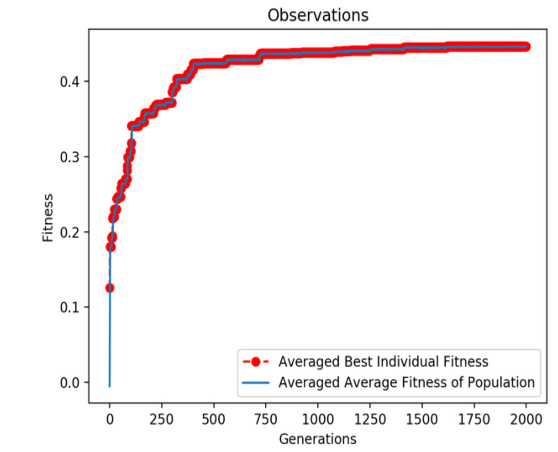

Fig. 6: Observations for PH=16, P=50%, T=40%, M=1%, population size=50

Future Steps

- Model with self-reproduction (plant-specific parameter)
- Introduction of elitism, generational mixing
- Comparison of globally and locally interacting agents for reproduction
- Alternative crossover procedures and rates
- Alternate machine learning tools and neural network architectures

Acknowledgements

Special thanks to Professor Kai Olav Ellefsen and Professor Keith L. Downing for their interest, insight, and suggestions.

References

[1] Baldwin, M. (1896). A New Factor in Evolution. *The American Naturalist,* (30):536-553.
[2] Bräutigam, K. and Cronk, Q. (2018) DNA Methylation and the Evolution of Developmental Complexity in Plants. 9, 1447
[3] Ellefsen, K. O. (2013) Balacing Costs and Benefits of Learning Ability. Conference Proceedings, *12th European Conference on Artifical Live, At Taorima, Italy*
[4] Suzuki, R. and Arita, T. (2004) Interactions between learning and evolution. The outstanding strategy generated by the Baldwin effect. BioSystems 77:57-71
[5] Downing, K. L. (2010). A Script-Based Approach to Evolving Neural Networks. In *Proceedings of the second Norwegian Artificial Intelligence Symposium*, pages 29-36.
[6] Stanley, K. O. and Miikkulainen, R. (2002). Efficient Evolution of Neural Network Topologies. In *Proceedings of the 2002 Congress on Evolutionary Computation (CEC '02).*
[7] https://github.com/deap/deap
[8] Zee, F. and Bubenheim, D. (1997). Plant Growth Model Using Artificial Neural Networks.
[9] Blickle, T. and Lothar, T. (1995). A Comparison of Selection Schemes used in Genetic Algorithms.
[10] Jebari, K. and Madiafi, M. (2013). Selection Methods for Genetic Algorithms