

Fitness Importance for Online Evolution

Philip Valencia *†
philip.valencia@csiro.au

Raja Jurdak *†
raja.jurdak@csiro.au

Peter Lindsay †
p.lindsay@uq.edu.au

*CSIRO ICT Centre. Brisbane, Queensland. Australia

†The University of Queensland, School of ITEE. Brisbane, Queensland. Australia

ABSTRACT

To complement standard fitness functions, we propose “Fitness Importance” (FI) as a novel meta-heuristic for online learning systems. We define FI and show how it can be used to dynamically manipulate the population composition in order to vary the instantaneous system performance at a tradeoff to learning capability. The effect of FI is demonstrated on a simple light-sensing and light-actuating optimisation problem on a collection of physical wireless sensor network devices. We also describe the In situ Distributed Genetic Programming (IDGP) framework which has been developed for online evolution of logic on resource-constrained computing devices and demonstrate how FI can be used with the framework in order to achieve a dynamic balance of learning and performing.

Categories and Subject Descriptors

I.2.2 [Computing Methodologies]: Artificial Intelligence—*Automatic Programming* [Program synthesis]

General Terms

Algorithms, Design

Keywords

Late Breaking Abstract, Fitness, Objective, Adaptive, Online, Evolution, Genetic Program, Wireless Sensor Network

1. FITNESS IMPORTANCE

1.1 Motivation

There are many examples where it is much easier to describe the desired system behaviour or outcome rather than how the system achieves this behaviour. Distributed systems comprised of many interacting entities, systems in changing and unpredictable environments, and non-determinism from sensing-actuating feedback loops are examples of complexities that often make acceptable human-designed logic

difficult to realise. Even when human-devised solutions are achieved, it is subject to their biases such as experiences and domain knowledge and designed with explicit and implicit idealisations about the real world. Consequently, expected behaviour is often limited to specific and sometimes unrealistic operating conditions. This is commonly referred to as “brittle” logic [1] and is largely due to the inability of humans to accurately predict the system behaviour in complex dynamic environments. Automated logic design methods, such as genetic programming (GP), can provide a means of devising novel logic to yield an acceptable solution. However performed offline, it too can be susceptible to “brittleness” when used in a dynamic, complex environment due to simplifications and incorrect assumptions about the real environment.

GP can also be performed “online” to allow the system to learn and adapt to unanticipated changes in the fitness landscape. There are far less examples of these systems as compared to systems that evolve logic “offline”. Within the “online” evolution examples, research has largely focussed on optimising learning (coverage and convergence) within a finite time and/or finite resources available. Implicitly, there is a notion that once the “best” solution has been found or at some arbitrary time, the system will “switchover” to run only the best solution forevermore (or until the system is no longer performing acceptably). Another implicit assumption is that system behaviour prior to the “switchover” is unimportant while performance after this time is important.

For a system which undergoes life-long learning (online evolution) however, one can argue that performance is important all of the time or at least there are times when the importance of performance is higher and lower. Learning implicitly means the system is not performing optimally and to an external party, online GP would potentially appear poorly-performing as it goes about trialling and testing offspring, mutated programs etc. There would appear moments of good performance as elite and highly ranked programs are evaluated, but on average it could look quite poorly. One could argue that the system performance as seen by an external observer is in fact the average population (pool) fitness. Using this assumption, the population distribution directly effects both the externally observed performance as well as the learning rate of the system which is largely represented by the improvement in elite fitness.

Somewhat of a paradox emerges from this since in order to perform better (longer term) the system must perform worse than what it could achieve (in the immediate term) in order to facilitate learning capability which ultimately

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

GECCO '10 Portland, Oregon USA

Copyright 2010 ACM X-XXXXX-XX-X/XX/XX ...\$10.00.

leads to better performance. For example, a system could just execute the best solution found thus far, however little knowledge is gained compared to a system which is optimally searching and could discover a much better solution. However the optimal searching system will be performing worse until it finds a better solution which means one needs to decide the importance of performing over the importance of learning.

To address this need for balancing learning and performing of online learning systems, we introduce a control parameter termed ‘‘Fitness Importance’’ (FI). FI can be extended to a function of time and in a sense describes the importance of achieving a high pool fitness. The corollary is that it can also be used to describe when performing (high pool fitness) is unimportant and hence can be used opportunistically to bias the system towards learning.

1.2 Definition

We define ϕ_t as the desired Fitness Importance at time t as described by the Fitness Importance function $\Phi(t)$. For convenience we will describe t as discrete units in terms of generations. Hence $t = 1$ represents the period corresponding to the first generation of programs that were evaluated. The values of ϕ range from 0 to 1 inclusive. A value of 0 corresponds to no importance being placed on the current fitness of the system, but it provides optimal convergence (i.e. minimum convergence time for an optimal solution). It is worth noting however that $\phi = 0$ does not mean that the fitness of the system is also 0. In fact, $\phi = 0$ represents the conventional GP approach for maximizing exploration. A value of 1 for ϕ however corresponds to a maximum importance of fitness, meaning that the system should perform as best as possible with the current knowledge of programs and their fitnesses. To achieve this, the program with the highest expected fitness (i.e. E_1) should be exploited. This is similar to offline evolutionary approaches where the best solution evolved offline is placed into the online environment and then not altered. However, no exploration (learning) occurs while $\phi = 1$ because of the lack of diversity in the program population.

With the limits of the Fitness Importance set, we now define the values between these limits as a linear scale of desired expected performance between the desired expected performance when $\phi = 0$ and when $\phi = 1$. i.e.

$$E(F_j)|_{\phi_{i+1}} = E(F_{i+1})|_{\phi=0} + (E(F_{i+1})|_{\phi=1} - E(F_{i+1})|_{\phi=0})\phi_{i+1} \quad (1)$$

or in terms of ϕ_j ,

$$\phi_j = \frac{E(F_j)|_{\phi_j} - E(F_j)|_{\phi=0}}{E(F_j)|_{\phi=1} - E(F_j)|_{\phi=0}} \quad (2)$$

Hence, when $\phi_j = 0.5$, it indicates a desired expected pool fitness to lie exactly half way between the expected average fitness of a standard GA, $E(F_j)|_{\phi=0}$ and the expected pool fitness of using only the current elite solution $E(F_j)|_{\phi=1}$.

1.3 Results

We demonstrate the effects of FI on a simple optimisation problem running on a physical system of wireless sensor network devices. The devices use an online genetic programming framework known as IDGP to achieve adaptation to unexpected environmental changes and FI can be used within this framework, see Figure 1, to effectively balance

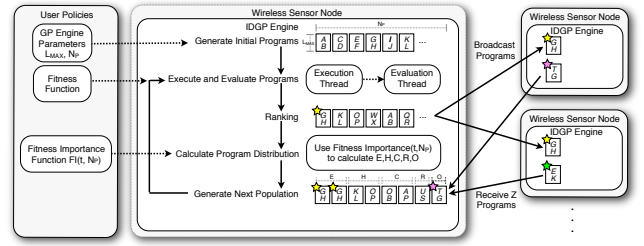


Figure 1: An overview of the In situ Distributed Genetic Programming (IDGP) Framework.

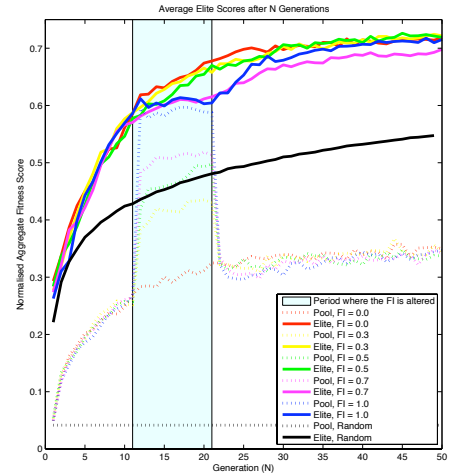


Figure 2: FI effects on system performance (pool fitnesses) and learning capability (elite fitnesses)

performance and learning during the life of the nodes. A number of experiments have been performed, such as Figure 2, to investigate various aspects and effects of the FI parameter which have also highlighted a number of open questions for online learning systems.

1.4 Conclusion

The ‘‘Fitness Importance’’ function was proposed as an addition to typical fitness functions. It was shown how this function can be used to bias the distribution of programs of a generated population in order to achieve higher system performance on demand. A tradeoff was evident where the reduction in diversity of the population to obtain higher expected fitness reduced the learning potential of the system. Mitigating this effect will be the subject of future research. In summary, the IDGP framework provides a method for continuous online learning in distributed systems while FI provides a practical means for dynamic control of the balance between learning and performing of the system.

2. REFERENCES

- [1] J. H. Holland. Escaping brittleness: The possibilities of general-purpose learning algorithms applied to parallel rule-based systems. In T. M. M. Ryszard S. Michalski, Jaime G. Carbonell, editor, *Machine learning: An artificial intelligence approach*, volume II, pages 593–623. Morgan Kaufmann, 1986.