Examining Forms of Inductive Bias Towards 'Simplicity' in Genetic Algorithms to Enhance Evolvability of Boolean Functions

Hetvi Jethwani
Indian Institute of Technology Delhi
New Delhi, India
mt1180754@iitd.ac.in

Sumeet Agarwal
Indian Institute of Technology Delhi
New Delhi, India
sumeet@iitd.ac.in

ABSTRACT

Recent work has looked at the evolvability of biological networks by representing them as graphs or functions, and studying the conditions under which certain structures are feasibly reachable. One key factor in enabling evolvability appears to be some form of prior constraint or *inductive bias* on the evolutionary landscape, which typically represents selectivity for low-complexity structures. Here we examine two different approaches for incorporating this kind of inductive bias in genetic algorithms, and propose a simulation framework allowing us to compare them and to relate the notion of a function class to that of a function's algorithmic complexity.

CCS CONCEPTS

• **Applied computing** → **Life and medical sciences**; **Systems biology**;

KEYWORDS

Evolution, Evolvability, Learning Theory, Algorithmic Complexity, Block Decomposition Method (BDM), Boolean Functions

ACM Reference Format:
Hetvi Jethwani and Sumeet Agarwal. 2021. Examining Forms of Inductive Bias Towards 'Simplicity' in Genetic Algorithms to Enhance Evolvability of Boolean Functions. In *2021 Genetic and Evolutionary Computation Conference Companion (GECCO '21 Companion), July 10–14, 2021, Lille, France.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3449726.3459429

1 INTRODUCTION

A key question in thinking about biological evolution algorithmically is that of computational complexity: what sorts of evolutionary targets are actually evolvable given feasible time and resources? Valiant [3] sought to use learning theory to characterise the evolvability of targets from certain Boolean function classes (representing combinatorial regulation of gene expression). Restricting the search space to a chosen class (such as conjunctions) here served as a form of *inductive bias*, in the learning-theoretic sense. Hernández-Orozco et al. [2], on the other hand, deployed the notion of algorithmic (Kolmogorov) complexity to favour the evolution of 'simple' network structures (represented as matrices). In different settings, both studies showed that biasing evolutionary search towards simplicity of some kind can speed up convergence to relevant targets. Here we propose a common simulation framework to examine how these two forms of inductive bias might relate to each other in terms of their facilitation of evolution, and whether they might be unified.

GECCO '21 Companion, July 10–14, 2021, Lille, France

ACM ISBN 978-1-4503-8351-6/21/07.
https://doi.org/10.1145/3449726.3459429

2 BACKGROUND

Valiant [3] suggests that Darwinian evolution can be formulated as a learning problem, where learning is only guided by aggregate fitness. Individuals/populations are modelled as many-argument Boolean functions, where the input variables may represent gene expression levels or environmental factors, and the output the response of a particular downstream gene. Within a genetic algorithm setup, restricting the search space to a particular function class allows him to obtain results on which function classes are evolvable in polynomial (in the number of input variables) time and resources. Simpler classes like conjunctions are shown to be more evolvable than less simple classes like parity functions.

Chaitin [1] introduced a model based on algorithmic information theory (AIT) where evolution is seen as a random walk in the space of all valid programs. The framework proposed by Hernández-Orozco et al. [2] explores this setting experimentally. They investigate how altering the probability distribution over mutations affects the rate of convergence of evolution, showing that when mutations with low algorithmic complexity are favoured, convergence to target matrices with some algorithmic structure can be significantly faster, relative to uniform mutations. This algorithmic structure is described using BDM (Block Decomposition Method) values, which are an approximation to the Kolmogorov complexity. Matrices with low BDM values are said to be more structured.

3 METHODOLOGY

We examine how the Valiant [3] notion of inductive bias via choice of function class might translate to the simulation framework of Hernández-Orozco et al. [2]. The distribution favouring low-complexity mutations in the AIT setting, called the *universal distribution*, could be seen as a 'soft' constraint on the mutation space, in contrast to the 'hard' constraint of explicitly restricting mutations to a chosen function class. We propose a framework which represents Boolean functions as matrices, to study their evolution in the AIT setting; and in addition to the soft constraint, also restricts the mutation space in a hard way by using the algorithmic structure.

Boolean functions are represented via *minterms*, *i.e.*, for a function of n variables it maps to a 2^n-bit string or $2^n \times 1$ matrix. The hard constraint is implemented via a BDM-value threshold where the

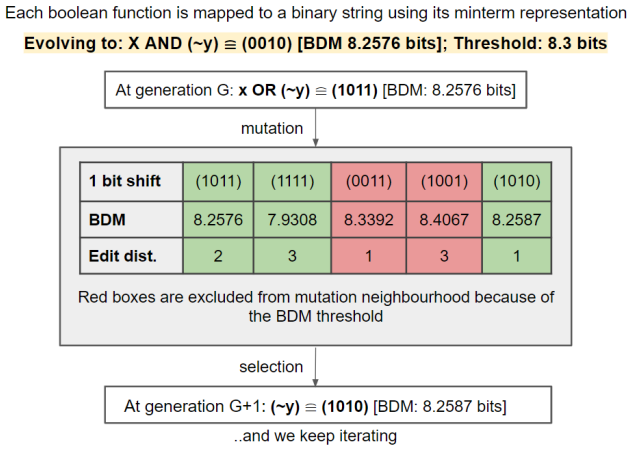

Figure 1: An example iteration with hard BDM threshold.

mutation space is $M(r) = \{x : Hamming(r, x) \leq 1 \ \& \ BDM(x) \leq threshold\}$ (Figure 1). In the 'soft' case, the mutations are biased towards structures with low BDM; in the 'hard' case, every mutation lies below a BDM threshold. The latter restricts the space to a subset of all Boolean functions which are meant to be simple in terms of algorithmic structure, and we explore how such subsets might relate to particular function classes like conjunctions.

4 RESULTS

We first examine how BDM values are distributed within Boolean function classes (Figure 2). Conjunctions are spread over a much smaller range than parity functions, and both are clearly separated from randomly sampled Boolean functions. This validates our key idea: a hard threshold on the BDM value can approximate the selection of mutations from a chosen function class, and intuitively 'simpler' classes appear to have lower BDM values.

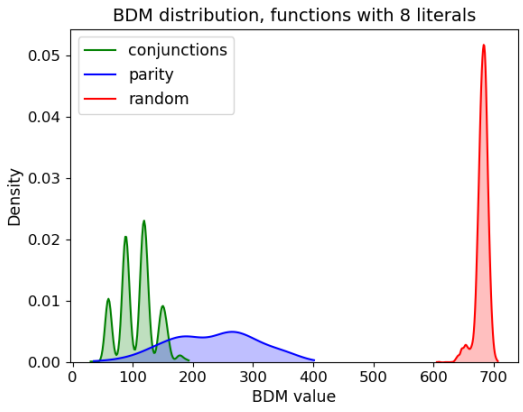

Figure 2: BDM value distribution by function type.

We then experiment with evolving target functions ranging over all conjunctions and parity functions. We do 50 runs for each target and average over all runs. The baseline 'no threshold' case uniformly allows all 1-bit shift mutations. The soft constraint setting converges fastest (Table 1) and our hard threshold case is faster than the baseline. As expected, mutations as per the full universal distribution are the strongest form of inductive bias, and lead to selected mutations with the lowest BDM values (Figure 3), while the hard threshold also sees some lowering compared to the baseline.

BDM Thr.	None	126.26 bits	145 bits	174.96 bits	185 bits	Full BDM distr.
Targets: 6-variable conjunctions						
Gens.	9.37 ± 5.12	8.74 ± 4.82	8.87 ± 4.92	9.01 ± 5.05	9.03 ± 5.04	5.04 ± 3.08
Exts.	0 ± 0.0	1.84 ± 5.18	0.81 ± 4.29	0 ± 0.0	0 ± 0.0	0 ± 0.0
Targets: 6-variable odd parity functions						
Gens.	32.76 ± 7.45	29.67 ± 9.37	31.49 ± 8.13	32.04 ± 7.11	31.96 ± 7.3	17.15 ± 9.78
Exts.	0 ± 0.0	41.8 ± 5.89	34.54 ± 10.67	0.02 ± 0.13	0 ± 0.0	0 ± 0.0

Table 1: Generations to converge and no. of extinctions for different inductive biases. Pink/blue: BDM thresholded at max. value for conjunctions and parity fns. respectively.

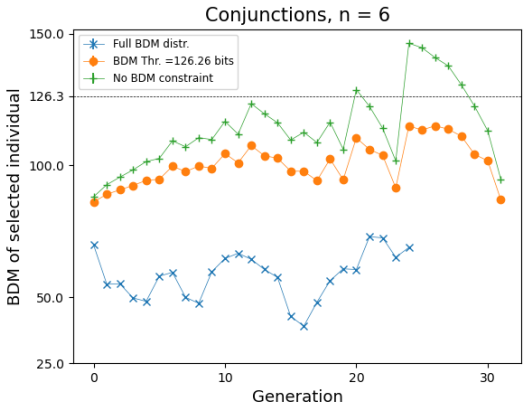

Figure 3: Generation-wise BDM values by bias setting.

5 DISCUSSION AND CONCLUSIONS

We show that bounding algorithmic complexity when evolving Boolean functions approximates the restriction of the search space to 'simple' function classes: a starting point for relating the 'hard' [3] and 'soft' [2] forms of inductive bias previously examined. Such a bound is a weaker form of bias than the full universal distribution, and leads to a smaller but still notable speed-up in the evolution of functions with algorithmic structure. Biologically, a restriction on realisable functional mechanisms (such as conjunction) may be a more plausible constraint on the search space than algorithmic complexity values of functions being directly encoded in the system. But the facilitation of evolution in such restricted search spaces might in turn be seen as the result of an implicit favouring of low-complexity mutations. These observations motivate further study of the relations between learning-theoretic and information-theoretic perspectives on the computational complexity of evolution.

REFERENCES

[1] Gregory J. Chaitin. 2009. Evolution of Mutating Software. *Bull. EATCS* 97 (2009), 157–164.
[2] Santiago Hernández-Orozco, Narsis A. Kiani, and Hector Zenil. 2018. Algorithmically probable mutations reproduce aspects of evolution, such as convergence rate, genetic memory and modularity. *Royal Society Open Science* 5, 8 (Aug. 2018), 180399.
[3] Leslie G. Valiant. 2009. Evolvability. *Journal of the ACM* 56, 1 (Jan. 2009), 1–21.