Noname manuscript No.
(will be inserted by the editor)

Behavioral Program Synthesis with Genetic Programming: by Krzysztof Krawiec

R. Muhammad Atif Azad

Received: date / Accepted: date

By combining Genetic Programming (GP) with program synthesis, the title invites the GP and program synthesis communities to bridge the gap between them. At the same time, it refocuses effort on the dream behind GP, that is, automatic computer programming. The dream (rather than just predictive modelling) distinguishes GP from a host of Machine Learning (ML) algorithms that are not designed to produce software. And so echoes the essence of discussions on how best to position GP in the wake of the recent successes enjoyed by some competing ML algorithms.

The preface sets a clear agenda. It wants GP to synthesise arbitrarily complex programs. To do this it says GP needs multifaceted fitness feedback. Typically, instead, the only feedback that GP is given is a scalar measure of the disparity between the actual and the ideal program outputs. The book calls this scalar feedback the evaluation bottleneck, as it blocks the kind of extra information that a human evaluator might use to evaluate a piece of code. Although not quite humanised, the multifaceted feedback which it advocates comes from a combination of sophisticated search drivers. Each driver attempts to quantify intricate aspects of the evolved program's behavior. Essentially, the goal is to convert program verification in GP from a black box into a white box by using much more sophisticated multi-objectives fitness measures which include knowledge of the program's internals as well as its external outputs.

Another critical aspect of programming is modularity (reusable subprograms). However, for GP, finding useful subprograms has been a challenge in part because the fitness function is blind to the role of subprograms. As a result, programs with potentially useful subprograms may be lost due to an overall poor rating. Krawiec addresses the challenge effectively with a simple yet novel approach that applies a Machine Learning classifier to the *execution*

R. Muhammad Atif Azad
CSIS Department, University of Limerick, Ireland.
E-mail: atif.azad@ul.ie

record of a GP-evolved program to identify a set of important subprograms. An execution record is a matrix of execution traces. Each row of the matrix corresponds to a set of inputs and contains the execution trace which records the sequence of effects of an executing program on its execution environment. The next row records the effects when the program is given the next set of inputs, and so on. Each column in the execution record corresponds to a particular sub-sequence of instructions, and the ML-classifier treats each column as an *attribute*. The ML-classifier then uses a subset of these attributes to model the ideal output desired from the program. The subset thus identified becomes the set of important subprograms. These subprograms are saved in an archive for later reuse. Chapter 10 shows that this archive further improves performance of several other ideas proposed later in the book. Thus, it potentially advances the state of the art in automatic identification of useful subprograms in GP; however, inexplicably, the preface does not highlight this.

The notion of recording the execution record is central to the book. For instance, Chapter 11 poses the execution record as a treasure chest of information that can be further exploited to derive useful search drivers. The final chapter recommends that the choice of such drivers should influence all aspects of metaheuristic-design, including initialisation, selection and program modification.

The book throws up a surprise in Chapter 10 when it shows that a many-objective search helps program synthesis. This is at odds with conventional wisdom which suggests that search with many objectives may degrade into random search. Nevertheless, Krawiec shows that the best performance comes with multiobjective search with up to five objectives. Although the book's principle recommendation is using multifaceted evaluation, it does not explain why so many objectives work well together.

Chapters 1-3 give the background; chapters 4-8 describe increasingly sophisticated measures to capture the behavior of a given program and how to use them to promote modularity in GP. Finally chapters 9-12 round up the discussion by presenting them as ways to drive the search. After reading the first three chapters, the reader can skip chapters 4 and 5 because the subsequent text does not critically depend on them.

Generally the text is very good. It first establishes the enormity of the challenge that automatic program synthesis poses, and then builds towards potentially ground breaking work that goes some way towards scaling this challenge. Rarely does a passage require re-reading and typing errors are rare. However, although the book addresses a broad audience interested in program synthesis, it inevitably draws on bio-inspired optimisation. Therefore, to appreciate the nuances some background in GP is essential. As such the book is only suitable for postgraduate or advanced undergraduate level.

It is important not to overstate *Behavioral Program Synthesis with Genetic Programming*'s accomplishments. It does not tackle constructs such as loops and conditionals. Also, it only briefly comments on computational expense. Moreover, Chapter 10 should have been reorganised to separate desirable search drivers (as described in section 9.12) from the search drivers in general. This would help the reader to appreciate the critique of previous work in section 9.10. Furthermore, the bibliography might have included work on automatic synthesis of parallel programs with GP. Then there are some follow-up questions. Why do many search objectives work in tandem? Is it because of a loosely monotonic relationship between them? Must that relationship be *designed* by the experimenter? Finally do the results from Chapter 10 apply to Semantic GP described in Chapter 5?

Overall the book advances the state of the art in GP based program synthesis. It is compact (only 147 pages). It offers excellent value and I highly recommend it.