A Review of Theoretical and Experimental Results on Schemata in Genetic Programming

Riccardo Poli and W. B. Langdon

School of Computer Science
The University of Birmingham
Birmingham B15 2TT, UK
E-mail: {R.Poli,W.B.Langdon}@cs.bham.ac.uk

Abstract. Schemata and the schema theorem, although criticised, are often used to explain why genetic algorithms (GAs) work. A considerable research effort has been produced recently to extend the GA schema theory to Genetic Programming (GP). In this paper we review the main results available to date in the theory of schemata for GP and some recent experimental work on schemata.

1 Introduction

Genetic Programming (GP) has been applied successfully to a large number of difficult problems [8,7,2]. However a relatively small number of theoretical results are available to try and explain why and how it works.

Since John Holland’s seminal work in the mid seventies and his well known schema theorem (see [6] and [3]), schemata are often used to explain why GAs work (although their usefulness has been recently criticised, e.g. in [4] and [1]). In particular it is believed that GAs solve problems by hierarchically composing relatively fit, short schemata to form complete solutions (building block hypothesis). So the obvious way of creating a theory for GP is to define a concept of schema for parse trees and to extend Holland’s schema theorem.

One of the difficulties in obtaining theoretical results using the idea of schema is that the definition of schema for GP is much less straightforward than for GAs and a few alternative definitions have been proposed in the literature. All of them define schemata as composed of one or multiple trees or fragments of trees. In some definitions [8,10,11,18,19] schema components are non-rooted and, therefore, a schema can be present multiple times within the same program. This, together with the variability of the size and shape of the programs matching the same schema, leads to some complications in the computation of the schema-disruption probabilities necessary to formulate schema theorems for GP. In more recent definitions [14,17] schemata are represented by rooted trees or tree fragments. These definitions make schema theorem calculations easier. In particular, in our work [14] we have proposed a new simpler definition of schema for GP which is much closer to the original concept of schema in GAs. This concept of schema suggested a simpler form of crossover for GP, called one-point crossover,
which allowed us to derive a simple and natural schema theorem for GP with one-point crossover. We will critically review the main results obtained to date in the theory of schemata for GP in Sect. 3 after briefly recalling and reformulating Holland's schema theory for binary GAs in Sect. 2.

Although theoretical results on schemata are very important, it is well known that schema theorems only model the disruptive effects of crossover and represent only short-term predictions. At the time of writing this paper, only one empirical study on GP schemata had been carried out [12]. This analysed the effects of standard crossover, one-point crossover and selection only on the propagation of schemata in real, although small, populations. We describe the main results of this study in Sect. 4 and we draw some conclusions in Sect. 5.

2 Background

As highlighted in [16], a schema is a subspace of the space of possible solutions. Usually schemata are written in some language using a concise notation rather than as ordinary sets, which would require listing all the solutions they contain: an infeasible task even for relatively small search spaces.

In the context of binary representations, a schema (or similarity template) is a string of symbols taken from the alphabet \{0,1,#\}. The character # is interpreted as a "don't care" symbol, so that a schema can represent several bit strings. For example the schema #10#1 represents four strings: 01001, 01011, 11001 and 11011.\(^1\) The number of non-# symbols is called the order \(O(H)\) of a schema \(H\). The distance between the furthest two non-# symbols is called the defining length \(s(H)\) of the schema. Holland obtained a result (the schema theorem) which predicts how the number of strings in a population matching a schema varies from one generation to the next [6]. The theorem\(^2\) is as follows:

\[
E[m(H, t + 1)] \geq m(H, t) \cdot \frac{f(H, t)}{f(t)} \cdot (1 - p_m)^{O(H)} \cdot \frac{P_a(H, t)}{1 - p_c} \cdot \frac{L(H)}{N - 1} \cdot \left(1 - \frac{m(H, t)f(H, t)}{M f(t)}\right)
\]  

(1)

where \(m(H, t)\) is the number of strings matching the schema \(H\) at generation \(t\), \(f(H, t)\) is the mean fitness of the strings in the population matching \(H\), \(f(t)\) is the mean fitness of the strings in the population, \(p_m\) is the probability of mutation per bit, \(p_c\) is the probability of one-point crossover per individual, \(N\)

\(^1\) As pointed out by one of the reviewers, both schemata and strings could be seen as logic formulae, and the problem of matching a string \(h\) against a schema \(H\) could be formalised using standard substitution techniques used in Logic Programming.

\(^2\) This is a slightly different version of Holland's original theorem (see [3, 21]).