
An Experimental Analysis of Schema Creation, Propagation and Disruption in Genetic Programming

Riccardo Poli* and W. B. Langdon*

School of Computer Science
The University of Birmingham
Birmingham B15 2TT, UK

Abstract

In this paper we first review the main results in the theory of schemata in Genetic Programming (GP) and summarise a new GP schema theory which is based on a new definition of schema. Then we study the creation, propagation and disruption of this new form of schemata in real runs, for standard crossover, one-point crossover and selection only. Finally, we discuss these results in the light of our GP schema theorem.

1 Introduction

Genetic Programming (GP) has been applied successfully to a large number of difficult problems [Koza, 1992, K. E. Kinnear, Jr., 1994, Angeline and Kinnear, Jr., 1996]. 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 [Holland, 1992] and [Goldberg, 1989]), schemata are often used to explain why GAs work (although their usefulness has been recently criticised, e.g. in [Altenberg, 1995]). 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 alternative definitions have been proposed [Koza, 1992, O'Reilly and Oppacher, 1995, Whigham, 1995]. These define schemata as composed of one or multiple fragments of a tree. These definitions allow a schema to 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 considerable complications in the computation of schema-disruption probabilities. Nonetheless two of these definitions have been used in schema theorems for GP, which we will review in Section 2.

In recent work [Poli and Langdon, 1997a] 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, which we call one-point crossover because of its similarity with the corresponding GA operator. As summarised in Section 3, this allowed us to derive a simple and natural schema theorem for GP.

Although theoretical results on schemata are very important, it is well known that schema theorems only model the disruptive effects of crossover. Indeed, the actual importance of the constructive effects of crossover is basically unknown as, to date, no experimental study on GP schemata has been reported.

In this paper the creation, propagation and disruption of GP schemata in real (although small) populations are experimentally studied for standard crossover, one-point crossover and selection only. We present the results of these experiments in Section 4, we discuss them in Section 5 and we draw some conclusions in Section 6.

2 Previous Work on GP Schemata

The first attempt to produce a schema theory for GP was made by Koza [Koza, 1992, pages 116–119], who produced an informal argument showing that Holland's schema theorem would apply to GP as well. The argument was based on the idea of defining a schema as the subspace of all trees which contain, as subtrees, a predefined set of *complete* subtrees. According to Koza's definition, a schema H is represented as a set of S -expressions, e.g.

*E-mail: {R.Poli, W.B.Langdon}@cs.bham.ac.uk

$H=\{(+ \ 1 \ x), (* \ x \ y)\}$ represents all the programs including at least one occurrence of $(+ \ 1 \ x)$ and one of $(* \ x \ y)$.

Koza's work was later formalised and refined into a schema theorem for GP in [O'Reilly and Oppacher, 1995]. A schema was defined as an unordered collection (a multiset) of subtrees and tree fragments. Tree fragments are trees with at least one leaf that is a "don't care" symbol ($\#$) which can be matched by any subtree. For example the schema $H=\{(+ \ \# \ x), (* \ x \ y), (* \ x \ y)\}$ represents all the programs including at least one occurrence of the tree fragment $(+ \ \# \ x)$ and at least *two* occurrences of $(* \ x \ y)$.

This definition of schema allowed the introduction of the concept of order and defining length for GP schemata. The order of a schema is the number of non- $\#$ nodes in the expressions or fragments contained in the schema; the defining length is the number of links included in the expressions and tree fragments in the schema plus the links which connect them together. Unfortunately, the defining length of a schema is not constant but varies with the programs sampling it and the probability of disruption $P_d(H, h, t)$ of a schema H due to crossover depends on the shape and size of the tree h matching the schema. O'Reilly and Oppacher overcame this problem by considering the maximum of such a probability, $P_d(H, t) = \max_h P_d(H, h, t)$ which could lead to severely underestimating of the number of occurrences of the given schema in the next generation. The resulting schema theorem is

$$E[m(H, t + 1)] \geq m(H, t) \frac{f(H, t)}{\bar{f}(t)} (1 - p_c P_d(H, t)), \quad (1)$$

where $m(H, t)$ is the number instances of the schema H in the population at generation t , $f(H, t)$ is the mean fitness of the instances of H , $\bar{f}(t)$ is the mean fitness of the programs in the population, p_c is the crossover probability and $E[\cdot]$ is the expected-value operator. O'Reilly and Oppacher discussed the usefulness of this result and argued that, given the intrinsic variability of $P_d(H, t)$ from generation to generation, no hypothesis can be made on the real propagation and use of building blocks in GP and, therefore, experimental studies are necessary to corroborate theoretical results on schemata.

In the framework of his GP system based on context free grammars (CFG-GP) Whigham produced a very general concept of schema for context-free grammars and the related schema theorem [Whigham, 1995]. In CFG-GP programs are the result of applying a set of rewrite rules to a starting symbol. The process of creation of a program can be represented with a derivation tree whose internal nodes are rewrite rules and whose terminals are the functions and terminals used in the program. In CFG-GP the individuals

in the population are derivation trees and the search proceeds using crossover and mutation operators specialised to produce valid derivation trees.

Whigham defines a schema as a partial derivation tree rooted in some non-terminal node, i.e. as a collection of rewrite rules organised into a single derivation tree. The terminals of a schema can be both terminal and non-terminal symbols of a grammar. Therefore a schema can actually represent all the programs that can be obtained by adding other rules to its leaves until only terminal symbols are present and by using it as a component for other derivation trees. This definition of schema leads to simple equations for the probability of disruption of schemata under crossover and mutation. Unfortunately, again these probabilities vary with the size of the complete derivation tree (i.e. of the program) containing the schema and a schema can occur multiple times in the same derivation tree. Whigham overcame these problems by considering the average disruption probability and the average fitness of each schema.

3 New GP Schema Theory

As highlighted in [Radcliffe, 1991], a schema is a subspace of the space of possible solutions, usually represented using some concise notation (rather than enumerating all the solutions it contains). Schemata are mathematical tools to describe which areas of the search space are sampled by a population. For schemata to be useful in explaining how GP searches their definition must make the effects of selection, crossover and mutation comprehensible and relatively easy calculate. The problem with earlier definitions of schema for GP is that they make the effects on schemata of the genetic operators used in GP too difficult to evaluate mathematically.

Recently we have proposed a simpler definition of schema for GP and a form of crossover which have allowed us to derive a new GP schema theorem [Poli and Langdon, 1997a]. We define a *schema* as a tree composed of functions from the set $\mathcal{F} \cup \{=\}$ and terminals from the set $\mathcal{T} \cup \{=\}$, where \mathcal{F} and \mathcal{T} are the function set and the terminal set used in a GP run. The symbol $=$ is a "don't care" symbol which stands for a *single* terminal or function. In line with the original definition of schema for GAs, a schema H represents programs having the same shape as H and the same labels for the non- $=$ nodes. For example, if $\mathcal{F}=\{+, -\}$ and $\mathcal{T}=\{x, y\}$ the schema $H=(+ (- = y) =)$ would represent the four programs $(+ (- x y) x)$, $(+ (- x y) y)$, $(+ (- y y) x)$ and $(+ (- y y) y)$. Our definition of schema is in some sense lower-level than those adopted by others, as a smaller number of trees can be represented by schemata with the same number of "don't care"

symbols and it is possible to represent one of their schemata by using a collection of ours.

The number of non- $=$ symbols is called the *order* $\mathcal{O}(H)$ of a schema H , while the total number of nodes in the schema is called the *length* $N(H)$ of the schema. The number of links in the minimum subtree including all the non- $=$ symbols within a schema H is called the *defining length* $\mathcal{L}(H)$ of the schema. For example the schema $(+ (- = =) \times)$ has order 3 and defining length 2. These definitions are independent of the shape and size of the programs in the actual population.

In order to derive a GP schema theorem for our schemata we used very simple forms of mutation and crossover, namely point mutation and one-point crossover. *Point mutation* is the substitution of a function in the tree with another function with the same arity or the substitution of a terminal with another terminal. *One-point crossover* works by selecting a common crossover point in the parent programs and then swapping the corresponding subtrees like standard crossover. In order to account for the possible structural diversity of the two parents, one-point crossover starts analysing the two trees from the root nodes and considering for the selection of the crossover point only the parts of the two trees which have the same topology (i.e. the same arity in the nodes encountered traversing the trees from the root node) [Poli and Langdon, 1997a].

The new schema theorem provides the following lower bound for the expected number of individuals sampling a schema H at generation $t + 1$ for GP with one-point crossover and point mutation:

$$E[m(H, t + 1)] \geq m(H, t) \frac{f(H, t)}{f(t)} (1 - p_m)^{\mathcal{O}(H)} \times \left\{ 1 - p_c \left[p_{\text{diff}}(t) \left(1 - \frac{m(G(H), t) f(G(H), t)}{M f(t)} \right) + \frac{\mathcal{L}(H)}{(N(H) - 1)} \frac{m(G(H), t) f(G(H), t) - m(H, t) f(H, t)}{M f(t)} \right] \right\} \quad (2)$$

where p_m is the mutation probability (per node), $G(H)$ is the zero-th order schema with the same structure of H where all the defining nodes in H have been replaced with “don’t care” symbols, M is the number of individuals in the population, $p_{\text{diff}}(t)$ is the conditional probability that H is disrupted by crossover when the second parent has a different shape (i.e. does not sample $G(H)$), and the other symbols have the same meaning as in Equation 1 (see [Poli and Langdon, 1997a] for the proof). The zero-order schemata $G(H)$ ’s represent different groups of programs all with the same shape and size. For this reason we call them *hyperspaces* of programs. We denote non-zero-order schemata with the term *hyperplanes*.

Equation 2 is more complicated than the corresponding version for GAs [Goldberg, 1989, Holland, 1992, Whitley, 1993] because in GP the trees undergoing optimi-

sation have variable size and shape. This is accounted for by the presence of the terms $m(G(H), t)$ and $f(G(H), t)$, which summarise the characteristics of the programs belonging to the same hyperspace in which H is a hyperplane.

In [Poli and Langdon, 1997a] we analysed Equation 2 in detail and discussed the likely interactions between hyperspaces and hyperplanes and the expected time-dependence of the probability $p_{\text{diff}}(t)$ during a run. In particular we conjectured that, given the diversity in the initial population, in general p_{diff} would be quite close to 1 and the probability of schema disruption would be quite high at the beginning of a run. Schema disruption and creation by crossover would then heavily counteract the effects of selection, except for schemata with above average fitness and short defining-length whose shape $G(H)$ is also of above average fitness and is shared by an above average number of programs.

We also conjectured that, in the absence of mutation, after a while the population would start converging, like a traditional GA, and the diversity of shapes and sizes would decrease. During this second phase, the competition between schemata belonging to the same hyperspace would become more and more important and the GP schema theorem would asymptotically tend to the standard GA schema theorem. Therefore, during this phase *all* schemata with above average fitness and short defining-length would tend to have a low disruption probability.

In [O’Reilly and Oppacher, 1995] serious doubts have been cast on the correctness of the Building Block Hypothesis (BBH) for standard GP. However, our analysis of Equation 2 led us to suggest that it cannot be ruled out in GP with one-point crossover.

4 Experimental Results

The number of different schemata contained in a single program of length N is 2^N , which is large even for short programs. Therefore, even a small population of GP programs sample such a huge number of schemata that it is not practical to follow in detail their creation, propagation and disruption during a run. However, it is possible to do so if one limits the depth of the programs being evolved to three or four levels. For this reason in the experiments reported in this paper we decided to impose these depth limits and to consider only a very simple problem, the XOR problem, which can be solved by small programs.

In the experiments we used the function set $\mathcal{F} = \{\text{AND, OR, NAND, NOR}\}$ and the terminal set $\mathcal{T} = \{x_1, x_2\}$. With these choices, by limiting the maximum allowed depth to 2 (the root node is at level 0) we were able to follow the evolution of all the schemata in a population of 50 individuals for 50 generations. We were also able to study subsets

of the schemata present in populations of programs with maximum depth 3.

The population was initialised using the “grow” method [Koza, 1992]. The fitness of a solution was the number of entries in the XOR truth-table it correctly predicted. In the following subsections we describe the results obtained with selection only and with different forms of crossover (with $p_c = 0.35$). Mutation was not used.

4.1 Schema Propagation Under Selection Only

In a first series of experiments we studied the effects of selection on the propagation of single schemata and on the number of hyperplanes and hyperspace sampled by the programs in the population in 10 GP runs with different random seeds. In all the experiments we did not stop the runs when 100% correct solutions were found. In each run all schemata in the population were recorded together with: the average fitness of the population in each generation, the average fitness of each schema in each generation, the number of programs sampling the schema in each generation, and the order, length and defining length of the schema.

Figure 1 shows the average population fitness, the diversity and the number of schemata in a typical run. Diversity has been assessed both in terms of different programs and in terms of different hyperspaces of programs ($G(H)$'s). The initial population contained four different hyperspaces, G_1 represented by the schema =, $G_2=(= (= = =) (= = =))$, $G_3=(= (= = =) =)$ and $G_4=(= = (= = =))$, and 46 different programs. Due to the biases of the “grow” method, these hyperspaces included quite different numbers of programs. For example G_1 was sampled by 6 programs, while G_2 included 40 programs. Of the four hyperspaces G_2 was the one with the highest fitness and after 10 generations it was the only one to survive. At that point it still contained four different types of individuals all with the same fitness. So from that point in time fitness proportionate selection was not acting any more. Nonetheless the diversity in the population kept slowly decreasing even after generation 10. This is due to the stochastic nature of the selection process which makes the number of instances of each individual vary with a kind of Brownian motion which can lead to extinction or to take over the whole population (genetic drift) [Ridley, 1993].

In the run shown in Figure 1 there were initially 2,892 schemata: an average of 62.9 per program. This relatively small number is the effect of the depth limit we imposed, which prevents any program from having more than 7 nodes and, therefore, more than 128 schemata. Before generation 10 the number of schemata decreases nearly exponentially as does the number of different individuals.

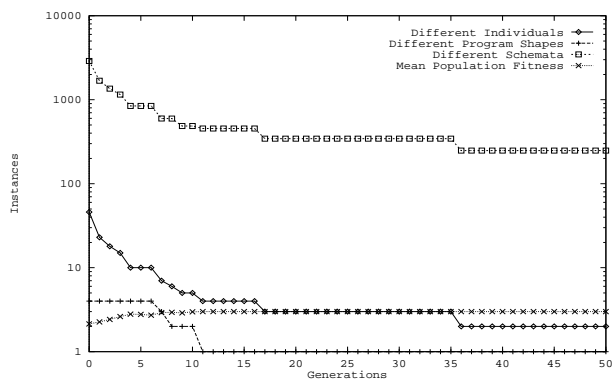


Figure 1: Average population fitness, diversity and number of schemata in a typical run of the XOR problem with selection only (maximum program depth 2).

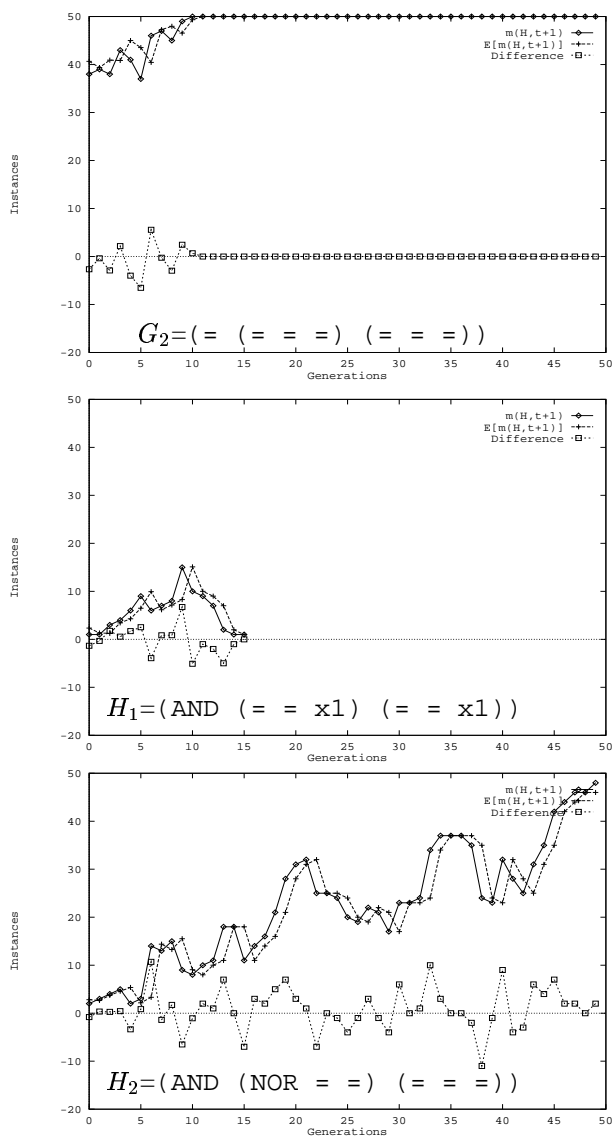


Figure 2: Effect of selection only on the programs sampling four different schemata in a typical run (maximum program depth 2).

According to the schema theorem, when selection only is acting, schemata with below average fitness should tend to disappear from the population while above-average schemata should be sampled more frequently. This effect was indeed observed in the run mentioned above at the level of both hyperplanes and hyperspaces. Figure 2 shows the expected value of $m(H, t + 1)$ predicted by the schema theorem and its actual value for the hyperspace G_2 , and for two of its hyperplanes $H_1 = (\text{AND } (= = x1) (= = x1))$ and $H_2 = (\text{AND } (\text{NOR } = =) (= = =))$.¹ In this run the competition between the few hyperspaces present in the population seem to end before genetic drift becomes the only driving force. This does not happen to relatively higher-order schemata. For example, both H_1 and H_2 , which are approximately of the same above-average quality, tend to spread in the population during the first 9 generations. However, they are initially sampled by a small number of programs. Therefore, when the selective pressure stops working at generation 9, neither of them has taken over the population ($m(H_1, 9) = 15$ and $m(H_2, 9) = 9$) and H_1 is driven to extinction by genetic drift very quickly.

Figure 3 shows the population fitness, the diversity and the number of schemata averaged over 10 different runs. These results suggest that selection is less effective on hyperspaces than on programs. Indeed, the rate at which 0-order schemata (hyperspaces) disappear is lower than for maximum-order schemata (programs). This can be explained by considering that the average deviation of the fitness of high-order schemata (e.g. programs) from the population fitness is in general bigger than for low-order schemata. Therefore, (positive or negative) selection will be stronger on average for high-order schemata and programs.

In this respect the situation does not change significantly if we change the depth limit to 3. Figure 4 shows the population fitness as well as the hyperspace and program diversity averaged over 10 different runs. In this case, the average number of different initial hyperspaces is 15, only approximately 3 times smaller than the number of different programs. Therefore, the average deviation of the fitness of hyperspaces from the population fitness is in general larger and selection is consequently stronger than in the previous case.

In summary, these experiments suggest that there is a good agreement between the predictions of the schema theorem and the observed number of instances of any schemata. However, we have not observed the expo-

¹The differences between the observed and measured values of $m(H, t + 1)$ are due to the stochastic nature of the roulette wheel algorithm implementing fitness proportionate selection. The apparent shift between the plots of $E[m(H_2, t + 1)]$ and $m(H_2, t + 1)$ is due to the fact that $f(H_2, t) \approx f(t)$ for $t > 4$.

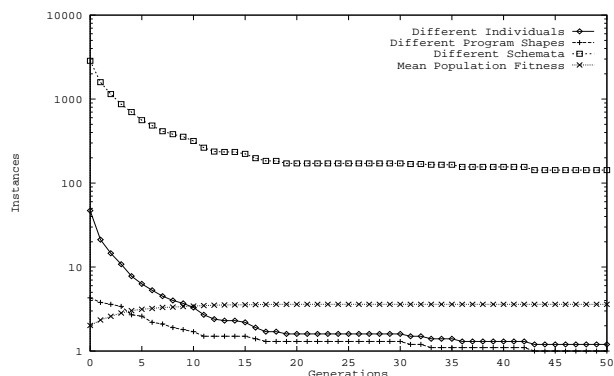


Figure 3: Average population fitness, diversity and number of schemata in 10 GP runs of the XOR problem with selection only (maximum program depth 2).

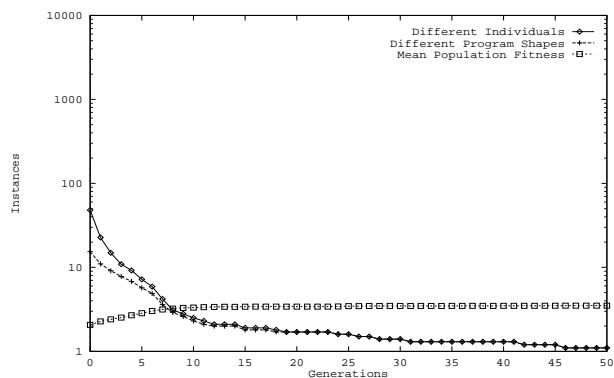


Figure 4: Average population fitness and diversity in 10 GP runs of the XOR problem with selection only (maximum program depth 3).

ponential schema growth/decay often claimed to happen in GAs [Goldberg, 1989]. The experiments also show that when small populations are used genetic drift plays an important role at all stages. This becomes prominent when the selective pressure decreases, i.e. when nearly all the programs in the population have the same fitness. Finally the experiments suggest that the diversity of more specific schemata (e.g. programs) decreases more quickly than the diversity of low-order ones (e.g. hyperspaces). This happens because high-order schemata have a wider range of fitnesses. This effect is not explicit in the schema theorem.

4.2 Effect of Standard Crossover

In a second set of experiments we considered the effect of normal crossover on the creation, propagation and disruption of schemata. Again we repeated 10 different runs for each program depth limit, using the same initial populations as in the runs studied in the previous section. In these runs, the offspring violating the depth limit were rejected and crossover was repeated until the correct number of offspring had been created.

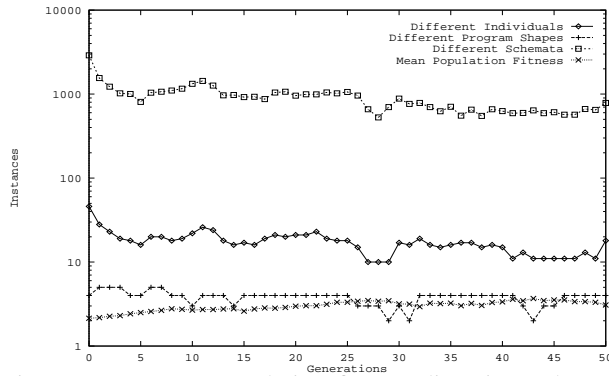


Figure 5: Average population fitness, diversity and number of schemata in a typical run of the XOR problem with normal crossover (maximum program depth 2).

Figure 5 shows the average population fitness, the diversity and the number of schemata in a typical run. The initial population in this run was the same as for the run studied in the previous section. Obviously the situation depicted in this figure is quite different from the one illustrated in Figure 1. In a short initial phase lasting 4 or 5 generations the standard deviation of the fitness of programs is large. Therefore, programs are subject to a relatively strong selection pressure. As in the selection-only case, hyperspaces are not as strongly effected. As soon as the population fitness starts saturating the effects of crossover become prevalent. These effects are the constant creation of new individuals and the destruction of old ones. They prevent the population from converging and maintain a high number of different schemata in the population. Towards the end of the run, there are approximately six times as many schemata as in the selection-only case. Standard crossover prevents the competition between hyperspaces from finishing as hyperspaces are constantly repopulated.

The situation does not change significantly if we look at the averages, shown in Figure 6, of population fitness, diversity and number of schemata in 10 different runs. The same is true for populations of programs with maximum depth 3, as shown in Figure 7. The only difference in this case is that the number of hyperspaces is larger and so each is sampled by fewer members of the population. Therefore, hyperspaces are subject to a stronger initial selective pressure.

4.3 Effect of One-point Crossover

One-point crossover behaves quite differently from standard crossover. Figure 8 shows the results obtained with the same initial population as before. Like in the case of selection only the competition between schemata ends and the population converges. The competition between low-order schemata finishes relatively more quickly than the compe-

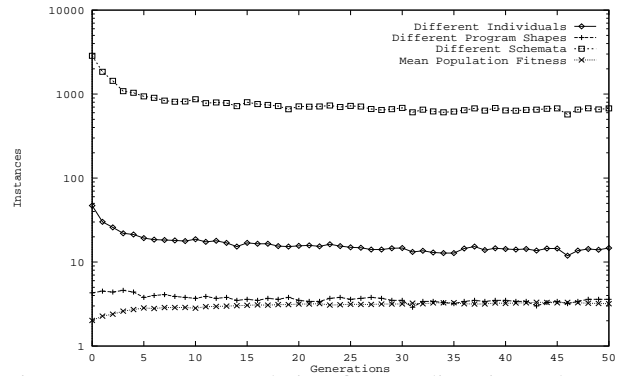


Figure 6: Average population fitness, diversity and number of schemata in 10 GP runs of the XOR problem with normal crossover (maximum program depth 2).

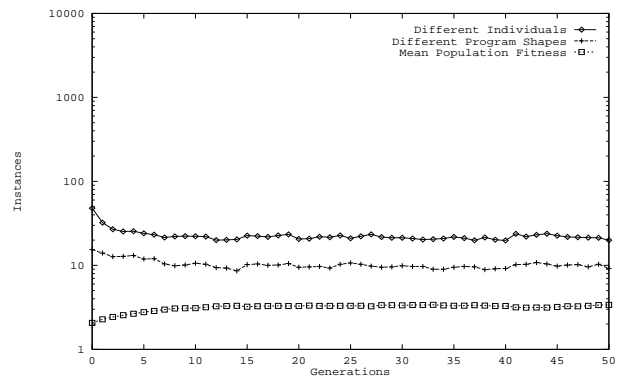


Figure 7: Average population fitness and diversity in 10 GP runs of the XOR problem with normal crossover (maximum program depth 3).

tion between higher order ones, e.g. programs. For example, when a final shape is selected in generation 5, there are still 10 different kinds of programs. After generation 19 the population has converged and no new individual can be created (without mutation).

This behaviour of one-point crossover is confirmed by the plots of the population fitness, diversity and number of schemata averaged over 10 different runs (Figure 9). These plots show that the average number of different individuals per hyperspace tends to vary very slowly during a run. This implies that when the competition between hyperspaces is over (not later than generation 19), the one between the programs sampling them is still active. This may take quite a few more generations to finish, but it always does it and the population always converges.

A careful comparison of the plots in Figures 9 and 3 reveals that, on average, the rate at which the diversity of high-order schemata changes is reduced by one-point crossover. This is due to the continuous creation of new schemata.

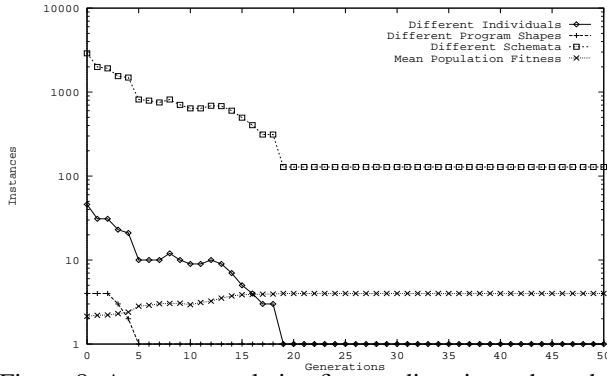


Figure 8: Average population fitness, diversity and number of schemata in a typical run of the XOR problem with one-point crossover (maximum program depth 2).

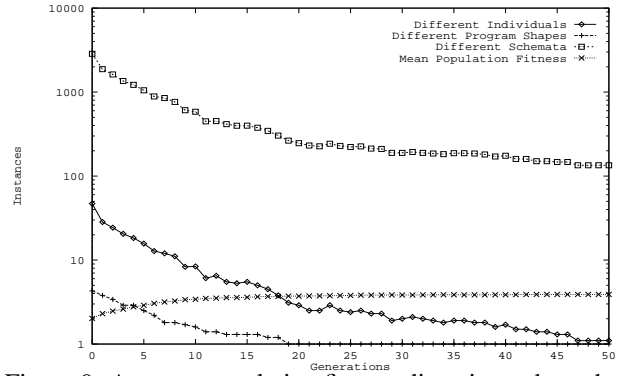


Figure 9: Average population fitness, diversity and number of schemata in 10 GP runs of the XOR problem with one-point crossover (maximum program depth 2).

As shown in Figure 10, the diversity of programs does not change significantly when the maximum allowed depth for programs is set to 3. However, like for the selection only case, the plot of the diversity of hyperspaces with depth 3 lies between the plots of the diversity of hyperspaces with depth 2 and the diversity of programs.

5 Discussion

When the size of the populations used is small genetic drift can be a major component in schema propagation and extinction. This certainly happened in our experiments as soon as the selective pressure decreased, both when selection only was present and when one-point crossover was used. Genetic drift could not become a major driving force in runs with standard crossover, because of the relatively large schema disruption/creation probability associated with it.

Although in the first few generations of runs with standard crossover selection was a primary driving force, the disruption and innovation power of standard crossover remained very strong throughout the entire runs and the population never converged. This contributed to maintaining a certain selective pressure even in late generations, as the average population fitness never reached its maximum.

The relatively large random oscillations of the total number of schemata observed in all runs with standard crossover (e.g. in Figure 5) seem to suggest that, however large, the probability of schema disruption $P_d(H, t)$ in Equation 1 is not constant but may vary significantly from one generation to the next and should be considered a stochastic variable as suggested in [O'Reilly and Oppacher, 1995]. Our experiments also suggest that the probabilities of disruption $P_d(H, t)$'s for schemata of different order are correlated.

One-point crossover allows the convergence of the popula-

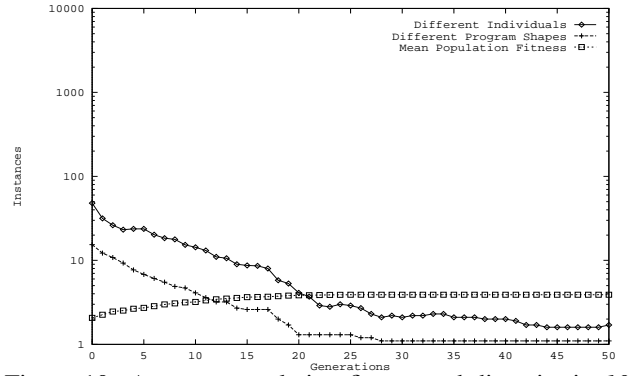


Figure 10: Average population fitness and diversity in 10 GP runs of the XOR problem with one crossover (maximum program depth 3).

tion. So, on average we must assume that its schema disruption probability is smaller than for standard crossover. However, if we compare the total number of schemata in the first few generations of the runs with one-point crossover and the runs with standard crossover, we can see that *initially one-point crossover is as disruptive as standard crossover*. This corroborates our conjecture that schema disruption would be quite high at the beginning of runs with one-point crossover.

The experiments also corroborate our conjecture that with one-point crossover, after this first highly-disruptive phase in which different hyperspaces compete, the competition between schemata belonging to the same hyperspace becomes the most important effect.

In the experiments with maximum depth 2, after generation 19 only one hyperspace was present in all runs. This means that the different programs in such hyperspace had all the same size and shape and that GP with one-point crossover was actually working like a GA with a non-standard crossover. This corroborates our conjecture that the GP schema theorem asymptotically tends to the GA

schema theorem.

This behaviour suggests that, while the BBH seems really in trouble when standard crossover is used, it is as relevant to GP with one-point crossover as it is for traditional GAs.

6 Conclusions

In this paper we have first reviewed the main results available to date on the theory of schemata for GP and summarised a new schema theory we have recently developed [Poli and Langdon, 1997a]. The theory is based on a new simpler definition of the concept of schema for GP which is very close to the original concept of schema in GAs. On the basis of this theory, we made several conjectures on the behaviours of GP with one-point crossover.

The motivation for the experiments described in this paper was to verify empirically the predictions made by us and by others. The experiments, although limited by the combinatorics of processing and storing schemata, have shown the substantial correctness of our and other theorists' conjectures. Standard crossover is very innovative and disruptive, and strongly counteracts selection. This prevents the population from converging. One-point crossover is very disruptive at the beginning of a run but becomes less and less so generation after generation. This suggests that mutation could be a very important operator in runs with one-point crossover, as recently observed in [Poli and Langdon, 1997b].

More theoretical and experimental work will be needed to investigate which form of crossover is better and in which cases.

Acknowledgements

The authors wish to thank the members of the EEBIC (Evolutionary and Emergent Behaviour Intelligence and Computation) group for useful discussions and comments. This research is partially supported by a grant under the British Council-MURST/CRUI agreement and by a contract with the Defence Research Agency in Malvern.

References

[Altenberg, 1995] Altenberg, L. The Schema Theorem and Price's Theorem. In Whitley, L. D. and Vose, M. D., editors, *Foundations of Genetic Algorithms 3*, pages 23–49, Estes Park, Colorado, USA. Morgan Kaufmann.

[Angeline and Kinnear, Jr., 1996] Angeline, P. J. and Kinnear, Jr., K. E., editors. *Advances in Genetic Programming 2*. MIT Press, Cambridge, MA, USA.

[Goldberg, 1989] Goldberg, D. E. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley, Reading, Massachusetts.

[Holland, 1992] Holland, J. *Adaptation in Natural and Artificial Systems*. MIT Press, Cambridge, Massachusetts, second edition.

[K. E. Kinnear, Jr., 1994] K. E. Kinnear, Jr., editor. *Advances in Genetic Programming*. MIT Press.

[Koza, 1992] Koza, J. R. *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. MIT Press.

[O'Reilly and Oppacher, 1995] O'Reilly, U.-M. and Oppacher, F. The troubling aspects of a building block hypothesis for genetic programming. In Whitley, L. D. and Vose, M. D., editors, *Foundations of Genetic Algorithms 3*, pages 73–88, Estes Park, Colorado, USA. Morgan Kaufmann.

[Poli and Langdon, 1997a] Poli, R. and Langdon, W. B. A new schema theory for genetic programming with one-point crossover and point mutation. *Proceedings of Genetic Programming '97*, Stanford, July 1997. Morgan Kaufmann.

[Poli and Langdon, 1997b] Poli, R. and Langdon, W. B. Genetic programming with one-point crossover and point mutation. Technical Report CSRP-97-13, School of Computer Science, The University of Birmingham.

[Radcliffe, 1991] Radcliffe, N. J. Forma analysis and random respectful recombination. In *Proceedings of the Fourth International Conference on Genetic Algorithms*. Morgan Kaufmann.

[Ridley, 1993] Ridley, M. *Evolution*. Blackwell Scientific Publications, Boston.

[Whigham, 1995] Whigham, P. A. A schema theorem for context-free grammars. In *1995 IEEE Conference on Evolutionary Computation*, volume 1, pages 178–181, Perth, Australia. IEEE Press.

[Whitley, 1993] Whitley, D. A genetic algorithm tutorial. Technical Report CS-93-103, Department of Computer Science, Colorado State University.