Spatial Co-Evolution - Quicker, Fitter and Less Bloated

Robin Harper
University of Sydney, Sydney, Australia
rhar9581@uni.sydney.edu.au

ABSTRACT

Operator equalisation is a methodology inspired by the crossover bias theory that attempts to limit bloat in genetic programming (GP). This paper examines a bivariate regression problem and demonstrates that operator equalisation suffers from bloat like behaviour when attempting to solve this problem. This is in contrast to a spatial co-evolutionary mechanism (SCALP) that appears to avoid bloat, without any need for express bloat control mechanisms. A previously analysed real world problem (human oral bioavailability prediction) is examined. The behaviour of SCALP on this problem is quite different from that of standard GP and operator equalisation leading to short, general candidate solutions.

Categories and Subject Descriptors
I.2.2 [Artificial Intelligence]: Automatic Programming—program synthesis

General Terms
Algorithms, Design, Experimentation

Keywords
bloat, GP, operator equalisation, spatial co-evolution (SCALP)

1. INTRODUCTION

Bloat is a problem plaguing evolutionary systems that utilise a variable length genome. Genetic Programming (GP) is no exception. A feature of bloat is that as the run progresses there is a tendency for the average size of the code to increase rapidly without any corresponding effective increase in fitness. Not only does this consume resources but it has also been linked to early convergence of the run [11]. Not surprisingly the phenomenon of bloat has been much studied. It has been linked to a number of theories including the ‘defense against crossover’ theory [1], the ‘fitness causes bloat’ theory [10] and the ‘crossover bias’ theory [2].

Recent work, inspired by the crossover bias theory, introduced an operator equalisation bloat control method. Section 2 explains the way this operator works in more detail, but essentially it seeks to control the sizes of newly generated individuals entering the population maintained by the GP system. To date a number of articles have shown how this operator, which has several flavours, is able to allow a GP system to stave off bloat, evolving fit and, importantly, short individuals. This is at the cost of requiring a large number of additional evaluations (sometimes by an order of magnitude). [17] indicates that further research is being carried out with the aim of reducing the overhead introduced by operator equalisation.

This paper examines a bivariate regression problem (XY regression) that GP has great difficulty solving, normally because of bloat. It is found that with XY regression even the operator equalisation methodology is often unable to constrain program sizes for long enough to allow a solution to be found. The increase in program size that occurs together with the large number of evaluations required by operator equalisation means that runs are frequently brought to a standstill. This is contrasted with a co-evolutionary system (SCALP, described in section 4) that solves XY regression with comparative ease, fewer evaluations (and wall-clock time) and in a way that seems to produce comparatively short solutions.

The performance of operator equalisation over a real world test problem has been analysed in several previous reports (e.g. [18, 16]). The comparison between it and both standard GP and SCALP is looked at in a slightly different light from the previous reports, with SCALP appearing to be able to create general solutions whilst avoiding bloat even when confronted with noisy data.

2. OPERATOR EQUALISATION

For the purposes of this paper standard GP means a genetic programming methodology similar to that exemplified by Koza [9]. In standard GP a generational system is typically used. In such a system there is a population of individuals and each generation a number of new individuals are created (typically by crossover). These new individuals together with a percentage of the old individuals form the population of the new generation. This process is repeated until a solution is found, the search process stagnates or one has exhausted the number of generations for which the search is conducted. Operator equalisation is a recent technique that has undergone a number of iterations [3, 18]. Unlike standard GP it vets the newly created individuals be-
fore allowing them to form part of the new generation. The aim is to prevent individuals that are too short to be useful or longer than needed from entering the target population. It uses the concept of a target distribution. If one imagines a histogram of the target distribution then each bar of the histogram is a bin that contains a number of individuals of a certain length. The width and position of the bar (or bin) is the range of individual lengths that are allocated to that particular bin and the capacity of the bin is the number of individuals that fit into it. A target distribution (see later) determines the capacity of each of the different bins.

2.1 Basic Equalisation Algorithm

The system works with the generational model of genetic programming. However, instead of just generating $x$ individuals each generation, it validates the entry of each individual into the new generation as follows:

- If its length puts it in a bin which has free space, then it is accepted.
- If its length puts it into a bin which is already full, it is evaluated and if its fitness is better than any other individual in that bin it is accepted;
- If its length is such that it is longer than any bin that has been created, then it is evaluated and if it is better than the best individual found in that run so far it is accepted and bins (of capacity one) are opened up between the previous maximum bin size and this new bin size;

Otherwise it is rejected and does not enter the new generation. This process is repeated until the requisite number of individuals has been accepted into the new generation.

2.2 Target Distribution

Most recent papers (e.g. [3, 16]) favour a target distribution algorithm that allocates bin sizes based on the average fitness of the individuals allocated to each bin in the previous generation (Dynamic Operator Equalisation). This creates a moving distribution that biases the search space for the new population towards the size of programs that currently have the best fitness, avoiding the creation of short unfit individuals (as well as longer individuals which do not improve the fitness of the population). In addition there are a number of equalisation techniques that use different distribution targets, such as flat distribution (where bin sizes are allocated equally up to the maximum bin that has opened so far). These have also been found to be effective [16]. In this paper the Dynamic Operator Equalisation method as outlined in [18] is used.

2.3 Increased Effort in Operator Equalisation

One of the features of the operator equalisation technique is that each generation every individual generated (whether or not they are accepted into the population) needs to be evaluated by the fitness function. This means that every generation there are a larger number of fitness evaluations than with standard GP. By way of example, with a population of 5,000, it was found that in a typical XY regression run 25,000 individuals might be evaluated in the first few generations of the run, dropping to about 15,000 individuals per generation after a few generations (once a larger variety of bin sizes have opened up). This can be compared to a standard GP run where there are only 5,000 evaluations per generation. This represents a considerable increase in the work required per generation with operator equalisation. This has been noted in previous papers and attempts have made to quantify the additional work in [17]. There are, however, many circumstances in which the trade off between work done and fighting bloat is a reasonable one. If the fitness function is relatively quick and the time taken to evaluate an individual is proportionate to the length of that individual (or the number of nodes in that individual) then the wall clock time of operator equalisation might not be larger (and may be smaller) than standard GP. This is because although there are more evaluations, each evaluation will be quicker than evaluations with a bloated population. There are also circumstances in which the aim is not only to find a solution but to find a solution which is short (and, hopefully, therefore more readily analysed). In such a case the amount of work done to find a solution is only of secondary importance to the type of solution found. Finally (as is the case with the XY regression problem discussed in section 4) it may be that the bloat in the standard GP runs prevents solutions from being found. If it constrains bloat, the additional work per generation required by operator equalisation may be worthwhile because it might allow better solutions to be found in circumstances where standard GP struggles.

On the other hand there are situations where the evaluation function requires a substantial amount of time, such as problems that require a simulation to be run, competitions to be held or even software to be run in an embedded environment. In such situations the length of the individual being evaluated is unlikely to be deterministic of the time taken to evaluate the individual. In these circumstances the additional burden of operator equalisation may well be unacceptable, as the time to move from one generation to the next will be greatly increased. Such problems require techniques that try to minimise the number of evaluations each generation.

3. SCALP

Spatial Co-Evolution in Age Layered Planes (SCALP) has been previously used to solve the XY regression problem [5] (detailed later). In SCALP the population is laid out in a number of planes. Each plane is a $M \times N$ grid which wraps at the boundaries. It is a combination of the co-evolutionary systems described, inter alia, in [7] and [19] and the age layered system (ALPS) described in [8].

SCALP works as follows: At the beginning of the run there is only one co-evolutionary plane. To start with each node within the plane is connected to the eight surrounding nodes. In a regression style problem each node contains a host (which is an individual or candidate solution) and a parasite (which is one of the test cases of the problem). Each generation the following takes place:

- **Fitness calculation:** The host in each node is evaluated against the nine parasites (test cases) in its local neighbourhood (being the one in its node and the eight in the surrounding nodes, a tile). A host’s score is the combined error from each of these nine test cases (the nine parasites). Typically for a regression style problem a low score is better. A parasite (a test case) is scored on the error it caused in the hosts that are evalu-
measured against it. For a parasite the bigger the error it causes the host to suffer, the better it has performed. A parasite is given a score equal to its least impressive performance against these nine hosts.

- **Selection:** For each node the host and its surrounding 8 hosts are ranked according to their fitness, from 1 to 9. A host has a probability of being selected for this site equal to 0.5\(^{\text{rank}}\) other than the last one which uses a probability of 0.5\(^{\text{rank}}\), ensuring the probabilities add up to 1. Each selected host is marked as the new host (h'). A similar process is applied to the parasites.

- **Crossover of hosts:** 40% of the nodes are selected for crossover. If crossover is to be performed at a particular site the h' at that site is crossed over with one of the surrounding h' (chosen randomly) and replaced by one of the children formed from the crossover (again selected randomly). With the regression style problem there is no crossover for parasites.

- **Mutation of hosts:** 20% of the nodes are selected and the h' in each of these nodes is mutated.

- **Mutation of parasites:** 10% of the nodes are selected and the p' at each of these nodes is mutated. If a parasite is mutated then it is replaced by a different test case.

Then if the new host (h') is valid it replaces the old host in each of the nodes. The new parasite (p') replaces the old parasite. A SCALP system consists of several stacked layers, where each layer limits the oldest genetic life of the individuals allowed in it. A genetic life is the number of generations an individual has lived through; a child inherits the genetic life of its oldest parent. For each subsequent layer introduced (see section 4.2) the competitive neighbourhood of a node is the eight surrounding nodes and the tile of nine nodes directly underneath it.

4. **THE XY REGRESSION PROBLEM**

This problem is a regression problem with two variables as follows:

\[
f(x, y) = \frac{1}{1 + x^{-4}} + \frac{1}{1 + y^{-4}}
\]

It is evaluated over the range of

\[-5 \leq x \leq 5 \text{ and } -5 \leq y \leq 5 \ (x, y \neq 0)\]

and has a graph of the form shown in figure 1. Typically the training set consists of data points spaced 0.4 apart between the limits, leading to 676 distinct x,y data points over which the regression can be trained (each a test point). As with a typical regression problem the fitness valuation reports an error value the sum of the absolute differences between the value given by the evolved function and value given by the target function for each of the test points (so a low fitness is better). A successful run is one in which every data point has an error of less than 0.01 (each success being a ‘hit’). The XY regression problem has been studied in a number of papers such as [15, 20] and as noted in [15] is easily scalable to any number of dimensions by the inclusion of additional variables. The reason why the problem is used here is the difficulty standard GP has in solving it. Over all the runs reported in the papers discussed above standard GP failed to find a solution and in this paper only, approximately, 12% of the runs were successful; indicating that the problem is very difficult for standard GP to solve. One of the problems standard GP has is that the population becomes bloated very quickly, effectively halting any improvements in fitness. Since this is exactly the type of problem that operator equalisation should be good at (the time taken to evaluate an individual is directly related to the complexity of the individual in question), XY regression is a good hard test of the principles behind operator equalisation.

4.1 Details of the system used: GP and Operator Equalisation

The experiments reported in this paper were implemented using a Grammatical Evolution system [14]. Grammatical Evolution (GE) is an enhancement of Genetic Programming that can be constrained to act in the same way as GP. It uses a Backus Naur Form (BNF) grammar to specify the phenotypes to be evolved by the system. Whilst there are a variety of crossover mechanisms available to a GE system, if the GP style crossover is used, with an appropriate grammar, then the system behaves in a way that is indistinguishable from a standard GP system [6]. For these experiments only the GP style crossover was used. A GE individual consists of a genotype, which is a series of numbers, called codons. These codons are translated by the grammar chosen to form a phenotype. In this case the phenotype is a mathematical expression that can be evaluated on the test data points. In this paper the length of an individual means the number of codons that are actually used to form the phenotype. The number of codons required to translate an individual’s genotype to its phenotype represents the number of choices that must be made when applying the grammar. In a more traditional GP system the phenotype is typically represented by an expression tree (or a lisp s-expression). The number of codons used to form an individual in a GE system corresponds closely to the number of nodes in a standard GP expression tree - rather than, say, the depth of the tree. It is an appropriate choice when trying to measure the size of the individual solutions, specifically when one is analysing the system for signs of bloat.

For the XY regression problem a simple no-function regression grammar was used. With the no function grammar
the phenotype consists of the standard +, -, *, and % (protected division) operators, the \( x \) and \( y \) parameters and constants that can be created by the system. Unlike Standard GP these constants can mutate. This simple (no function, no types) grammar was used in the anticipation that the results of the paper will have a wider application to all GP systems and not just GE systems. For each of the runs with standard GP and operator equalisation a population of 5,000 was used. The runs were allowed to continue for 100 generations. Each child generated had a 5% chance of a mutation being applied. If a child was generated that was invalid it was rejected and a new child was generated in its place. An invalid child is one that fails to translate from the codons (genotype) to the phenotype. (This doesn’t occur often but when it does it is typically because an individual has too few codons to complete the translation process.) Tournament selection was used to select individuals for breeding, with a tournament size of 7. With the standard GP and the operator equalisation runs a form of lexicographic parsimony pressure was applied. Basically this means that if two individuals would otherwise have the same fitness, the shorter individual is favoured. Such parsimony control does, however, seem to be slightly less effective where there is a floating point fitness function, such as in the present case.

The populations were initialised using Sean Luke’s probabilistic tree creation method (version 2)[12]. This methodology creates a broader distribution of tree shapes and sizes than, say, Koza’s ramped half and half method [4]. For the operator equalisation method runs were conducted with bin widths of 1 and 7. Save for the number of evaluations required each run (see subsection 4.4) there was no significant difference between the runs so only the runs with bin width 1 are reported.

4.2 Details of the system used: SCALP

The SCALP system was run using a layer size of 30 x 30 (900 individuals), with layer lives of 300, 600, 1200 and 2700. This means that the lowest layer does not allow individuals to remain in it if they are older than 300 generations, the second layer older than 600 generations and so on. The fifth layer has an implied “no life limit”. Layers are only generated when required. So for the first 300 generations there is only one layer. After 300 generations, but before 600 generations there two layers, etc. Finally it should also be remembered that in the SCALP system there are far fewer test case evaluations per individual per generation. In the lowest layer there are only 9 evaluations per individual (being the nine parasites in an individual’s neighbourhood) compared with 676 evaluations per individual per generation with standard GP. Each subsequent layer introduces a further 18 evaluations (for a total of 9 + 18 or 27) per individual per layer, the additional evaluations being the evaluations against the nine parasites in the corresponding tile in the layer immediately below and the comparison evaluation of the host in the layer below with the nine parasites in the upper layer. This means, for instance, that in the first 300 generations there are (30 * 30) population * 300 (generations) * 9 (evaluations) = (approx.) 2.4 million evaluations. This is fewer evaluations than the equivalent of a single generation of a

\footnote{It should also be noted that determining whether an individual is valid or invalid is different from and (for any non-trivial problem) far less time-consuming than evaluating the fitness of the individual.}

5,000 population standard GP system (5000 population * 676 evaluations = (approx.) 3.4 million evaluations). Once the last layer is in place (four layers with limited lives, one with unlimited lives) there will be 4,500 individuals (five layers of 900 individuals) and a total of 105,300 evaluations per generation. Going by evaluations performed; an approximate equivalence of SCALP generations to standard GP generations is 50:1. Consequently in graphs comparing the different systems where generations form one of the axes, 50 SCALP generations are taken as equaling one standard GP generation.

4.3 Preparation of Data

Where graphs are produced showing items such as fitness over time against generations, the data has been compiled by taking the median result from the 25 runs performed. The median is used because it is more robust to some of the outliers that are observed - particularly so in the Bio Availability runs, described in section 5. If a run was successful (and therefore terminated early) the results for the successful generation were replicated if data was needed from that run for later generations. Where a box-whisker chart (such as figure 2) is shown to demonstrate the spread of a particular measure at the end of the run, then each of the runs is represented. As is typical for such charts the median is the middle line, the box represents the quartiles, the whiskers the most extreme runs providing they lie within 1.5 times the inter-quartile spread. Outliers (if shown) are plotted separately.

4.4 Analysis of Operator Equalisation

Figure 2 shows a box-whisker chart of the fittest individual produced at the end of the XY problem runs; a low fitness being better. The number after the name indicates the number of runs (out of 25) that had a successful solution (less than 0.01 error for each of the 676 test points). The difference in success rate between standard GP and operator equalisation was not significant. As can be seen operator equalisation produced a “tighter” range of fitness than standard GP. There were several standard GP runs which suffered from bloat very early on, as a consequence of which even the best individuals at the end of such runs had a very poor fitness. As discussed in section 2.3 oper-
tor equalisation requires far more evaluations than standard GP. Whilst standard GP evaluates 5,000 individuals per generation, operator equalisation required, on average, approximately 16,000 evaluations per generation with a bin width of 1 and 32,000 evaluations per generation with a bin width of 7. No other bin widths were tried. Figure 3 shows a generation by generation comparison of those two methods. As noted earlier, other than the increased effort, there appeared to be no notable differences between the runs with a bin width of 1 and those with a bin width of 7 and so only the bin width 1 runs are reported.

Figure 4 shows a standard plot of fitness versus generations. For each run the best fitness in each generation was recorded. The plot shows the median of these recorded best fitnesses per generation. Looking at the plots of the standard GP fitness and the operator equalisation fitness it can be seen that operator equalisation appears to find lower (and therefore better) fitness levels more quickly than standard GP. However, when reading this graph it must be kept in mind that operator equalisation is performing approximately three times the number of evaluations over standard GP. As well as the fitness of the population it is also important to look at the average size of the programs in the population. Figure 5 shows the median of the average program size in the population for each of the different methods. As can be seen with operator equalisation and standard GP average program size increases, with very little to distinguish the two methods. The question then becomes whether this is an increase in size that corresponds to an increase in fitness or whether the system is bloating.

The metric used, inter alia, in [18] provides a methodology to answer this question. Instead of comparing fitness with evaluations or generations, it compares fitness with program length. For this type of chart, if a better fitness is lower, a line heading down the chart (south) shows a fitness that improves without an increase in program length, a line which heads south-east indicates a better fitness, at the cost of increasing program length and a line which heads directly east indicates the type of condition which is, by definition, bloat (increase in average program length, no corresponding improvement in fitness).

Figure 6 shows the fitness/length graphs for each of the three different runs conducted. Looking at the standard GP run it can be see the graph runs south east at the start showing an improvement in fitness together with an increase in size followed by a longer east running tail showing an increase in size with very little improvement in fitness. This is typical of a standard GP run on many such problems and indicates bloat taking hold. Looking at the operator equalisation run one can see a “south” running line from about a fitness of 250 to approximately 75 indicating an improvement in fitness without an increase in size, i.e. operator equalisation is constraining bloat. After that, however, the graph quickly levels off running east and showing that operator equalisation is also suffering from bloat like behaviour. (the SCALP run is analysed in the next subsection.) From the above it appears that with both standard GP and operator equalisation as the generations increase the runs in-
increasingly suffer from an increase in program size with very little (if any) improvement in fitness.

### 4.5 Analysis of SCALP

As can be seen from the box-whisker chart in Figure 2, SCALP produced a solution in 24 out of the 25 runs within the 5,000 generations allocated. 16 of the 25 solutions had a fitness of zero (i.e. they had found the actual solution), of the remaining 9, 8 found approximate solutions (i.e. 676 hits) and one had not yet found the solution. The ability of a spatial co-evolution system to solve XY regression is well documented (e.g. [15, 20]). What has not been looked at before is the length of the solutions found.

In order to analyse whether SCALP was controlling bloat it was necessary to gather additional metrics from the system. To this end every 50 generations each individual was evaluated as for each of the 676 data points in order to measure how the fitness of the population was improving. This measurement was for data gathering purposes only and did not affect the evolutionary algorithm. In addition the average size of the individuals in the oldest layer was measured. This data was incorporated into Figures 4 to 6. Figure 5 shows the average length of programs in the top layer as against time (measured by way of generations). The interesting point about this chart is that the length appears not only to increase, but to decrease as well. As previously mentioned, evolution was halted once a solution was found and the long flat length versus generations line for SCALP must be treated with a bit of caution as a result of this.

After about generation 3000 (generation 60 on the graph) well over 50% of the SCALP runs had solved so the median would be unchanged even if the unsolved runs bloat.

Figure 7 shows the average of the average size (rather than the median of the average size shown in Figure 6). This confirms that the SCALP program size is not increasing for those runs that have yet to find a solution. With a typical standard GP or even operator equalisation run the length trends upwards. With SCALP, whilst longer individuals appear in the population, it appears that the dynamics of the methodology (e.g. the constantly changing test cases) serves to penalise larger individuals, allowing smaller individuals to dominate the population. Unlike the standard GP and operator equalisation methodologies there was no form of length control or direct parsimony pressure. This would seem to imply that smaller individuals are able to use the evolutionary mechanisms (crossover and mutation) to adapt to the change in parasites more quickly. Perhaps the ‘defence against crossover’ provided by long individuals serves to inhibit adaptation to the changing environment. The plots in Figures 6 and 7 demonstrate the difference between the behaviour of SCALP and standard GP/operator equalisation. As can be seen, towards the end of runs, fitness improvements are correlated with a shrinking of program lengths (a west moving line in the figure) - the exact opposite of bloat.

## 5. REAL WORLD PROBLEM - HUMAN ORAL BIOAVAILABILITY

This is a real world problem that has been the subject of various operator equalisation papers (e.g. [18]). It is the prediction of human oral bioavailability of a set of drug compounds, based on their molecular structure. This paper uses the same dataset as [18], which is available online. The dataset consists of 262 items (representing molecules) with 241 parameters and a target value. Each parameter is a molecular descriptor, the target value is the known bioavailability value. In previous papers the datasets were prepared by randomly splitting them at the start of each run so that 70% of the molecules (randomly selected with uniform probability) formed the training set, whilst the remaining 30% formed the validation set. The phenotype was based on a simple regression problem with non-terminals *+, - and % (protected division). For tree creation, where a choice is to be made between choosing a non-terminal or a terminal, an equal probability was assigned with respect to whether a non-terminal would be chosen or a terminal. A similar methodology was adopted for the tests in this paper.

Some of the interesting things about this dataset is that it may well be noisy, the data included may be insufficient to give an accurate prediction of bioavailability and/or the representation chosen may not be complex enough to allow accurate predictions. How such a dataset impacts on the co-evolution methodology is interesting.

[18, 17] were concerned with whether operator equalisation might prevent over-fitting of the data on such datasets. Not surprisingly they concluded that over-fitting and bloat were not positively correlated and may in fact be counter-indicative. It is arguable that any system which is capable of learning an arbitrarily complex relationship will over-fit when presented with noisy/partial data. Within any set of noisy and/or partial data there will exist relationships that are purely coincidental and limited to that sample of the data. A system that strives to fit the data perfectly is likely to take advantage of these coincidental rules, which will limit the generality of the solution. The trick with decision trees and, it is argued, GP is to know when to stop building/evolving [13]. One method of doing this is to use the validation data set to work out when to stop learning. One can either halt the learning process when the validation score starts to worsen or when the combination of the training data and the validation score starts to worsen. The problem with using validation data for this purpose is that if the data is used to determine which individual to return it cannot be used to work out how successful the run was - the validation data becomes part of the overall data used to generate the evolved individual. Consequently, a third set of data is required to work out how successful the train-
5.1 Changes necessary for SCALP

One of the issues with SCALP is that every generation each individual is only assessed against a small number of the training cases. It therefore becomes difficult to assess when the run has been completed, especially if, as with this problem, there is no perfect solution. In order to allow a comparison to be made with the two other methodologies each SCALP run was terminated after a set number of generations (5000). The individuals in the oldest layer were tested every 50 generations as against the training data and validation data do not influence the course of the run performed. It should be noted that for this problem whilst the evaluations every 50 generations on the combined training data and validation data do not influence the course of the evolutionary run they are used in determining which evolved individual is returned at the end of the run. Accordingly these additional evaluations constitute additional overhead for this methodology.

5.2 Analysis

Figures 9 and 10 show the plots of fitness against generations and fitness against program length for this problem. The plots for operator equalisation and standard GP are much as expected. Both show improving (decreasing) fitness as the evolution proceeds. Figure 10 shows the “south” headed line for operator equalisation which indicates that fitness is improving without an increase in size. However as the fitness continues to decrease the average length of the programs increases - although not as much as they do with standard GP\(^3\). The behaviour of the SCALP run is quite different. Instead of a continual improvement in fitness as one might expect, figure 9 demonstrates an oscillating fitness. The reasons for this are complex and linked to the way the parasites (test cases) self select. Whilst figure 9 is showing the score against all the test data, at any one time the top layer is only evolving against a subset of this. Since this oral bioavailability problem does not appear to have a perfect solution (for this representation at least) SCALP appears to be cycling between different solutions which score well against different parasites. Unlike both operator equalisation and standard GP, SCALP seems incapable of finding a hypothesis that scores better (less) than, approximately, 29 when expected. Both show improving (decreasing) fitness against program length for this problem. As can be seen SCALP is criss-crossing between different solutions which score well against different parasites. Unlike both operator equalisation and standard GP, SCALP seems incapable of finding a hypothesis that scores better (less) than, approximately, 29 when tested against all the training data. Figure 10 shows the same effect, although this time it is plotting program length against fitness. As can be seen SCALP is criss-crossing between certain lengths of population size. Unlike the XY regression problem there is no perfect solution to be found and so it keeps cycling through imperfect solutions. What is remarkable is that the population does not appear to bloat, even to try and avoid over-fitting by returning the individual that scores best on the training data and the validation data. The unseen test data is used to work out how well the run performed. It should be noted that for this problem the unseen test data is used to try and avoid over-fitting by returning the individual that scores best on the training data and the validation data.

3The difference between the OpEq results shown here and in [18] appear to be mainly attributable to the different initialisation methodology and the increased population size.

---

Figure 8: Bioavailability data. The first three plots show the scores of the individual that had the lowest (best) score on the combined training and validation (T+V) data. Operator equalisation found the best fit for this data. The second three plots show the score on the test data, this is the data not seen by the evolutionary system. SCALP returns the tightest and lowest scores here suggesting it is finding better general solutions.

Figure 9: Bioavailability Data, showing the fitness against the training data as it changes during the course of the run. Both standard GP and operator equalisation show continued improvement as they fit the data, but SCALP is constantly oscillating apparently unable to fit the data.

---

3Since there were relatively few items of validation data (about 54) using only the validation data to try and determine if over-fitting had occurred led to large variations between runs.
presumably for similar reasons as discussed with the XY regression problem, that is if the solution becomes too long it is not responsive to the change in the parasites.

Figure 8 contains the last items to note. Whilst the best individual returned by SCALP does not score as well on the training (plus validation) data as, say, operator equalisation, there is a very tight range returned over all 25 runs. The performance of SCALP on the unseen test data is the best of the methods, returning a tight range of low scores. This would seem to indicate that SCALP is not over-fitting the training data unlike the other two methods (despite their use of validation data to try and prevent over-fitting).

6. SUMMARY AND CONCLUSIONS.

SCALP appears to be remarkably resilient to bloat even although it has no express bloat control measures. Its behaviour in the oral bioavailability problem is both concerning and exciting. It is concerning because it appears that it is incapable of fully fitting (over-fitting) the data, which indicates that there will be problems it struggles with. Exciting because it appears to be better at finding a general solution than either of the two other methods examined in this paper. Its search of the program space is quite different from more traditional methods. Of course it is not possible to draw any conclusions on one set of test data, but the results are definitely interesting and warrant further investigation.

7. REFERENCES


