Improving Convergence in Cartesian Genetic Programming Using Adaptive Crossover, Mutation and Selection

Roman Kalkreuth
TU Dortmund University
Department of Computer Science
44221 Dortmund, Germany
Email: roman.kalkreuth@tu-dortmund.de

Günter Rudolph
TU Dortmund University
Department of Computer Science
44221 Dortmund, Germany
Email: guenter.rudolph@tu-dortmund.de

Jörg Krone
South Westphalia University
of Applied Science
58644 Iserlohn, Germany
Email: krone.joerg@fh-swf.de

***Abstract*—Genetic programming (GP) can be defined as an evolutionary algorithm-based methodology which opens the automatic derivation of programs for problem solving. GP as popularized by Koza uses tree representation. The application of GP takes place on several types of complex problems and became very important for Symbolic Regression. Miller and Thomson introduced a new directed graph representation called Cartesian Genetic Programming (CGP). We use this representation for very complex problems. CGP enables a new application on classification and image processing problems. Previous research showed that CGP has a low convergence rate on complex problems. Like in other approaches of evolutionary computation, premature convergence is also a common issue. Modern GP systems produce population statistics in every iteration. In this paper we introduce a new adaptive strategy which uses population statistics to improve the convergence of CGP. A new metric for CGP is introduced to classify the healthy population diversity. Our strategy maintains population diversity by adapting the probabilities of the genetic operators and selection pressure. We demonstrate our strategy on several regression problems and compare it to the traditional algorithm of CGP. We conclude this paper by giving advice about parameterization of the adaptive strategy.**

I. Introduction

Genetic Programming (GP) was first introduced by Cramer [2] in 1985. GP as popularized by Koza [8], [7], [9] in 1990 uses LISP programs represented as parse trees. GP is an automated method for creating computer programs by defining a high-level problem statement. GP is often used for Symbolic Regression, since the parse trees can represent mathematical expressions, fitting a given dataset. In 1999 Miller and Thomson [15], [17] introduced Cartesian Genetic Programming (CGP) which uses a directed graph representation. CGP is often used for classification and image processing problems [14]. CGP originally uses only mutation as genetic operator and is based on an integer genome representation. In 2007, Clegg et al. [1] introduced a new representation of CGP called Real-Value-CGP which enables the use of crossover. The new representation is based on floating points. Although, on complex problem, CGP shows a low convergence rate and suffers from a common issue in evolutionary computation called premature convergence. The algorithm traps into local optima and the population is getting too homogeneous to proceed towards global optima. Slany [20] improved the convergence of integer-based CGP with the Age Layered Population Structure (ALPS) [6] on image operator design problems. But for more complex problems the convergence gain decreased and became less superior. Miller and Smith investigated the correlation between efficiency and node count and achieved a better performance of integer-based CGP [16]. Clegg et al. outlined the problem of low convergence in Real-Value-CGP on a regression problem and introduced a variable crossover. Meier et al. [13] investigated the problem in Real-Value-CGP by introducing a new genetic operator called *Forking* which uses population statistics to fork homogeneous areas in the search space. Another approach to achieve performance acceleration in genetic programming was introduced by Langdon [10] which includes a multithreaded CUDA interpreter. For our research, we tend more to handle the problem by improving the CGP algorithm itself as through increasing computational performance.

In the field of genetic algorithms [5],previous research showed that adaptive genetic operators and selection are helpful to maintain population diversity and to prevent premature convergence. Many adaptive strategies benefit from adapting the probabilities of crossover and mutation [3], [19], [21]. Later McGinley et al. [12] proposed an adaptive genetic algorithm which also uses an adaptive selection method by measuring the healthy population diversity. Clegg et al. introduced a variable crossover operator for CGP, though it merely performs a change to mutation only CGP by gradually decreasing the crossover probability. The variable crossover is based on their observation on regression problems and does not solve the problem itself.

This paper introduces an adaptive strategy for Real-Value-CGP which is based on the population statistics of modern GP Systems. The paper shows in what way these statistics can be used to maintain population diversity. The new strategy has been tested on several regression problems. Our strategy benefits from locating suitable points in the evolutionary process to adapt the probabilities of the genetic operators. As a result, population diversity is increased which may lead to better convergence. On harder problems the new strategy also benefits from adapting the selection pressure. To locate opportunities for adaption, a new metric for CGP is introduced which measures the healthy population diversity.

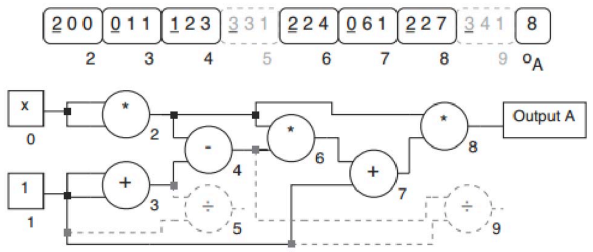

Fig. 1. CGP genotype (sequence of numbers) and phenotype for an individual. The phenotype is determined by decoding the genotype through backward search. Source: Miller et al. [14]

In the following section we describe the background of CGP, CGP population statistics and adaption of genetic operators. In Section 3 the new adaptive strategy for CGP is introduced. Section 4 shows the results of different regression problems. In the last section we discuss conclusions and future work. One purpose of our new strategy in the longterm is the improvement of existing approaches which will be discussed in this section.

II. Background

In this section, we explain CGP in detail, the creation of population statistics in CGP and the adaption of genetic operators.

A. Cartesian Genetic Programming

Cartesian Genetic Programming is a form of Genetic Programming which is often used for evolving digital circuits. In contrast to conventional GP, CGP represents a program via genotype-phenotype-mapping as an indexed, acyclic and directed graph as shown in Figure 1. CGP is similar to another technique called Parallel Distributed GP, which was introduced by Poli [18]. Originally the structure of the graphs was a multi-row representation, but later work focused on a representation with at least one row. Figure 1 shows an CGP genotype (sequence of numbers) and its corresponding phenotype for an quadratic function. Each group in the genotype refers to a node of the graph except the last one. The group consists of two types of numbers which index the number in the function set (underlined) and the inputs (non-underlined) for the node where the number depends on the arity of the function. The last group represents the index of the node which leads to the output. A recursive backward search through the graph is used to evaluate the program as shown in Figure 2. The backward search starts from the program output and processes all nodes which are linked in the genotype. In this way only active nodes are processed during evaluation. The integer-based representation of the CGP phenotypes restricts CGP to use mutation as genetic operator. The importance of crossover in genetic programming has also been underlined in the past [8]. The real value representation of the CGP genotype as shown in Figure 3 allows the crossover of two genotypes by a weighted average crossover where the genes vary in the interval $\in [0, 1]$. Merely swapping parts of the genotype slowed down the efficiency on different problems [1]. By only using mutation as genetic operator, integer-based and real-valued CGP show similar convergence behaviour. Clegg et al. showed that the

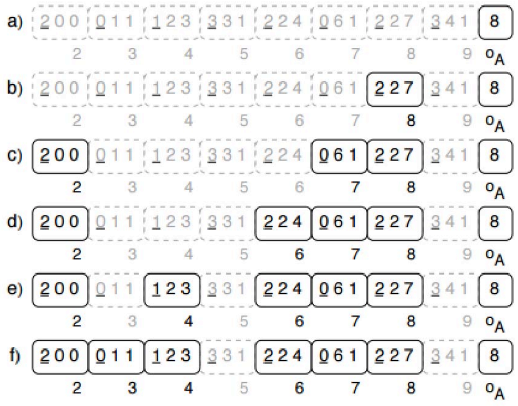

Fig. 2. The decoding procedure of a CGP genotype. a) Output A (oA) connects to the output of node 8. b) Node 8 is connected with the nodes 2 and 7, move to nodes 2 and 7. c) Nodes 2 and 7 are connected with the output of node 6 and the two inputs 0 and 1, move to 6. d) Node 6 is connected with the output of the nodes 2 and 4, move only to 4 because 2 has already been decoded. e) Node 4 is connected with the output of nodes 2 and 3, move to 3 f) Node 3 is connected with the program input 1. Source: Miller et al. [14]

new representation in combination with crossover improves the convergence behaviour for the first generations. For the latter generations, Clegg et al. outlined that the use of crossover in Real-Value-CGP influences convergence negatively.

B. Population Statistics in CGP

In CGP two types of population statistics can be used. Information about the population can be received by exploring the fitness landscape and the phenotype space. On the problem of premature convergence many adaptive approaches react on an homogeneous fitness landscape by increasing the influence of the mutation operator to increase population diversity. In addition to the classical approach of using population statistics by exploring the fitness landscape, Meier et al. explored the phenotype space for the forking operator [13]. The phenotype of every individual can be expressed as a string representation called fingerprint. In CGP different genotypes can refer to the same phenotype. This property is an example of neutrality in CGP and was outlined by Miller and Thomson [17]. The fingerprint of a phenotype is used to determine its frequency in the population. The number of unique phenotypes can be used to classify the diversity of the population. On this way homogeneous areas in the solution space can be detected. This can be more effective than merely exploring the fitness landscape.

```
o0 = * (- 1.0 (* 1.0 (* x x)))
        (- (* 1.0 (* x x)) (* (* x x) (* x x)))
```

Listing 1. Textual representation of a phenotype of fitting the function $x^6 - 2x^4 + x^2$

C. Adaption of genetic operators

Evolutionary algorithms handle with two main abilities. Exploitation of the genetic material in the population and the exploration of the search space. In CGP, exploitation is mostly done by crossover where mutation is used for exploration. Exploitation and exploration can be controlled by the probabilities

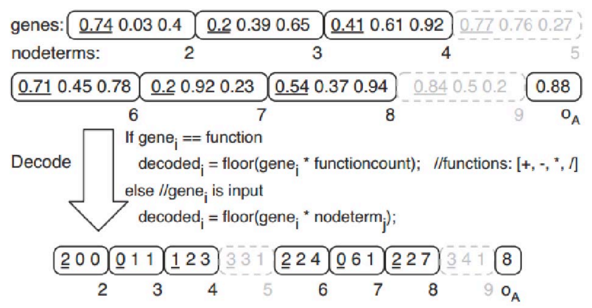

Fig. 3. Decoding from real-value to integer based genotype Source: Meier et al. [13]

of crossover and mutation. Let crossover probability be the probability that the crossover occurs when two parents have been selected, otherwise one of the parents is passed through by random. Let mutation probability be the probability that a gene of the genotype is mutated (replaced by a random number). Finding the right probabilities of the genetic operators is the key for an efficient search. Running the algorithm with inappropriate probabilities can lead to premature convergence or to excessive diversity. During the evolutionary process, the conditions are changing among generation to the next so that the probabilities can be adapted to the current conditions. Previous research showed that increasing crossover probability at high population diversity and increasing mutation probability at low diversity has a beneficial effect [12]. The key of success for an adaptive strategy is the estimation of the right moments for converting the population or exploring the solution space. Beside to adaption of the probabilities of the genetic operators, controlling selection pressure has also shown a beneficial effect [12].

D. Related Work

In this paper we utilize the computational effort (CE) as a measure introduced by Koza in [7]. The CE statistic is used to report the amount of computational effort to solve a problem with 99% probability by a GP System. We use the minimum of the computational effort as shown in Equation (5). This methodology has been used by Koza to describe experiments on several problems. For this methodology we first have to define a cumulative probability of success $P(N,l)$ (Equation (1)) which represents the number of runs which have been successful after l generations $(N_s(l))$ in relation to the number of total runs N_{total}. by using N individuals in each run.

$$P(N,l) := \frac{N_s(l)}{N_{total}} \tag{1}$$

Since we hope that our GP System can solve the problem with 99% probability we have to determine the number of runs which are required to find the solution. Because of premature convergence a 99% probability of success in every run may never occur. If R runs are independent the odds of failure for all runs can be calculated by

$$P_{\text{all fail}} := (1 - P(N,l))^R \tag{2}$$

and with odds of failure we can compute the number of independent runs $R(z)$ which are required to get a solution with a confidence interval of z where z is often chosen as a probability of 99%.

$$R(z) := \left\lceil \frac{\log(1-z)}{\log(1-P(N,l))} \right\rceil \tag{3}$$

The CE statistic $I(N,z,l)$ describes the number of evaluations which have to be performed to solve a problem to a proportion of z. This is done by multiplying the total number of individuals processed at the end of generation number l to $R(z)$.

$$I(N,z,l) := (N \cdot l \cdot R(z)) \tag{4}$$

Koza defined the statistic over all generation numbers l to find the minimum computational effort $I_{\text{min}}(N,z)$ to solve a given problem. For the determination of $I_{\text{min}}(N,z)$ we take the minimum of all sampled $I(N,z,l)$ as shown in Equation (5).

$$I_{\text{min}}(N,z) := \arg\min_l (N \cdot l \cdot R(z)) \tag{5}$$

III. Introducing the new strategy

A. Measuring Phenotype Space Diversity

Measuring the diversity of the phenotype space is the core of our adaptive strategy. Where other strategies measure the diversity only in genotype space, our strategy benefits also from measuring the diversity in phenotype space. Let S be the phenotype space. Individual $s \in S$ with $s = \Sigma^*$ over the alphabet Σ is the textual representation of a certain phenotype. An example of a textual representation of s is shown in Listing 1. Let N be the number of individuals and $S^* \subset S$ the subset of phenotypes described by all individuals. Phenotype space diversity θ can be split into two terms, standard phenotype diversity θ^{s} and healthy phenotype diversity θ^{h}. We receive information about the diversity of the whole phenotype space from θ^{s}, where θ^{h} refers to the ratio of healthy phenotypes in the population. A healthy phenotype is of high fitness and unique in the phenotype space which is of high value for the evolutionary process. To determine θ^{s} we define a dictionary $M : S \to \mathbb{N}_0$ which shares all fingerprints of the phenotype space. As value of M the frequency of a phenotype in the population is stored. The value of θ^{s} is calculated by the size of the dictionary in ratio to the number of individuals as shown in Equation (9). When every individual refers to one phenotype the dictionary size is equal to the population size which is the best standard diversity in phenotype space. For θ^{h} we first have to determine the phenotype health of every individual in the population. For calculating the phenotype health, the fitness rate F^{Rate} (Equation (7)) of an individual and the frequency of the phenotype in relation to the diversity of the population f^{Rate} (Equation (6)) have to be determined. In Equation (7) $F(i)$ stands for the fitness value of i-th individual in the population. Furthermore, F^{Worst} and F^{Best} represent the best and worst fitness value in the population. The frequency rate f^{Rate} is multiplied with an amplifier α as shown in Equation (8) to amplify lower frequencies and is limited to a maximum with a value of 1. In the discussion section we give advice for the parametrization of the amplifier which is based on our experiments. Finally phenotype health can be determined as the product of the fitness rate and negated frequency rate, since lower frequency rates are better. The sum of the phenotype health values over the population can be used to describe θ^{h} and is normalized by N, consider Equation (10). To calculate θ, we average θ^{s} and θ^{h} as shown in Equation (11).

$$f^{\text{Rate}}(j) := \frac{M(j)}{|M|}, \quad \text{for all } j \leq |S^*| \tag{6}$$

$$F^{\text{Rate}}(i) := \frac{F(i) - F^{\text{Worst}}}{F^{\text{Best}} - F^{\text{Worst}}}, \quad \text{for all } i \leq N \tag{7}$$

$$w(j) := \min(\alpha \cdot f^{\text{Rate}}(j), 1), \quad \text{for all } j \leq |S^*| \tag{8}$$

$$\theta^{\text{s}} := \frac{|M|}{N} \tag{9}$$

$$\theta^{\text{h}} := \frac{1}{N}\sum_{i=1}^{N}(1 - w(i)) \cdot F^{\text{Rate}}(i), \quad \text{for all } i \leq N \tag{10}$$

$$\theta := \frac{\theta^{\text{s}} + \theta^{\text{h}}}{2} \tag{11}$$

B. Adapting crossover probability

With information about the phenotype space diversity, the crossover probability can be adapted to the current conditions. When θ is high, the population consists of healthy phenotypes and less homogeneous areas which is a good point to increase crossover probability to proceed the population towards the global optima. Otherwise when θ is low, we have to handle with more homogeneous areas where a high crossover probability can convert the population to a local optimum. In this case crossover probability is reduced to increase diversity through mutation. The crossover probability is limited by a lower and upper limit. This prevents the crossover probability from getting exceed boundaries. Equation (12) shows the adaption of the crossover probability with the limits C_L and C_H. We define the limits C_L and C_H empirically. In CGP, the crossover probability often vary between the value 0.5 and 1.

$$P^{\text{Cross}} = \theta \cdot (C^{\text{H}} - C^{\text{L}}) + C^{\text{L}} \tag{12}$$

C. Adapting mutation probability

In contrast to adapt a global crossover probability, mutation probability is adapted locally. Since mutation has a strong effect on the phenotype in CGP, a global adaptive mutation is not recommendable. By increasing the mutation probability for all phenotypes, good solutions with high fitness would not protected by altering their genotype through mutation. Also the population is getting too homogeneous when the mutation probability is too low. The mutation probability is adapted directly in phenotype space based on the health of the phenotype. Healthy phenotypes are protected by a lower probability where unhealthy phenotype will be mutated with higher probability. As a result diversity is increased in areas where the population is too homogeneous. The need of a higher mutation rate for a phenotype can be determined by its frequency. The mutation probability is calculated as shown in Equation (13) with the limits M^{L} and M^{H}. Since each individual has its own mutation probability the probabilities of two individuals are averaged if crossover occurs. Like the limits of the crossover probability, M^{L} and M^{H} are set empirically. In GGP, the mutation probability often vary between a value of 0.01 and 0.3.

$$P^{\text{Mut}}(i) = w(i) \cdot (M^{\text{H}} - M^{\text{L}}) + M^{\text{L}} \tag{13}$$

D. Adapting selection pressure

On more complex problems we also adapt the selection pressure in relation to θ^{h}. When θ^{h} is high, the selection pressure is increased to select individuals with high fitness. At low θ^{h}, selection pressure is decreased, to give lower fitting individuals a greater chance to get selected. Normally we handle with the principle of survival of the fittest but in situations with a homogeneous population a higher selection pressure is not beneficial. By selecting lower fit individuals which are of higher diversity, θ^{h} is increased. As we use tournament selection, the tournament size is adapted as shown in Equation (14). Like in traditional GP we tend to use higher selection pressure. The limits are set empirically as we show in the next section.

$$T^{\text{Size}} = \theta^{\text{h}}(T^{\text{H}} - T^{\text{L}}) + T^{\text{L}} \tag{14}$$

IV. Results

In this section we compare our new strategy with the traditional Real-Value-CGP [1] on four different regression problems. We perform two different types of experiments. The first experiment is similar to the experiments of Meier et al. with a fixed tournament size to evaluate the use of adaptive crossover and mutation. We intend to use both adaptive operators in further research to develop a combined strategy with the forking operator. For the first experiment we chose the regression problems f_1 and f_2. The second experiment was performed on the more complex regression problems f_3 and f_4. For this experiment we also use adaptive selection pressure. Let $T = \{x_p\}_{p=1}^{\mathcal{P}}, x_p \in [-1, 1]$ be a training dataset of $\mathcal{P}$ random points and $f_{\text{ind}}(x_p)$ the value of an evaluated individual and $f_{\text{ref}}(x_p)$ the true function value. Let

$$C := \sum_{p=1}^{\mathcal{P}} |f_{\text{ind}}(x_p) - f_{\text{ref}}(x_p)|$$

be the cost function. When the difference of all absolute values becomes less then 0.01, the algorithm is classified as converged. To evaluate the results, we use a methodology based on Meier et al. and Clegg et al. which includes the average number of generations until convergence and the computational effort. For the calculation of the computational effort, we take the minimum computational effort (Min. CE) as shown in Equation (5). For each average number, e.g. 89 ± 216, the first number refers to the average value and the second to the standard deviation. Furthermore, we classify every run which exceeds 1000 generations as slow run and summarized the number of slow runs for each problem. We perform 1000 independent runs with different random seeds on every regression problem and use the Mann-Whitney-U-Test [11] to classify the significance of the results. The average number of generations is denoted with a^* if the significance level is $P < 0.05$ or $a^\dagger$ if the significance level is $P < 0.01$. Diagrams which show the detailed convergence behaviour, the

TABLE I. ALGORITHM CONFIGURATION FOR THE FIRST EXPERIMENT

Property	Traditional	Adaptive
Maximum node count	10	same
Function lookup table	+ (0), - (1), * (2), / (3)	same
Population size	50	same
Maximum Generations	20,000	same
Crossover operator	weighted average	same
Crossover probability (P^{Cross})	0.75	0.5 - 1.0 (general)
Mutation operator	Reset gene $\in [0, 1]$	same
Mutation probability (P^{Mut})	0.2	0.2 - 0.3 (general)
Tournament selection size (T^{Size})	20	same
Elitism size	2	same
Amplifier	-	3 (general)

TABLE II. THE AVERAGE NUMBER OF GENERATIONS AND COMPUTATIONAL EFFORT REQUIRED BY CGP FOR THE PROBLEM $f_1(x) = x^6 - 2x^4 + x^2$

Algorithm	Avg. Generations	Min. CE	Slow Runs
Traditional	151 ± 712	22622	17
Adaptive (general)	$101 \pm 169^*$	20380	5
Adaptive (specific)	$89 \pm 128^\dagger$	17828	3

function value of the best solution has been averaged for each generation over all runs.

$$f_1(x) = x^6 - 2x^4 + x^2 \tag{15}$$

$$f_2(x, y) = (x^2 \cdot y^2)/(x + y) \tag{16}$$

$$f_3(x) = x^5 - 2x^3 + x \tag{17}$$

$$f_4(x) = x^4 + x^3 + x^2 + x \tag{18}$$

A. Experiment I

For the first experiment we use the same parameter configuration as Meier et al. for the traditional Real-Value-CGP which has been found to be the best probabilities for mutation and crossover on regression problem. The parameter configuration for the traditional Real-Value-CGP and our adaptive approach is shown in Table (I). Our adaptive strategy uses the same configuration except the mutation and crossover probability. For the adaptive probabilities and the amplifier we define a general configuration for all problems which is marked as general. Furthermore we use a specific configuration for the adaptive strategy which is marked as specific and represents the optimal strategy parametrization for the given problem. For example for Problem 1 we use a crossover probability ranging from 0.7 to 1.0 and α is set to 5. The optimal strategy parametrization has been determined empirically. Since our strategy maintains population diversity, we are able to use higher crossover probabilities with less risk of trapping into local optima. Table II shows the results for the first problem which outlines that the adaptive strategy convergences faster with general and specific settings. Figure 4 shows a comparison

TABLE III. THE AVERAGE NUMBER OF GENERATIONS AND COMPUTATIONAL EFFORT REQUIRED BY CGP FOR THE PROBLEM $f_2(x, y) = (x^2 \cdot y^2)/(x + y)$

Algorithm	Avg. Generations	Min. CE	Slow Runs
Traditional	313 ± 612	60689	71
Adaptive (general)	$222 \pm 444^\dagger$	44559	31
Adaptive (specific)	$214 \pm 422^\dagger$	42915	33

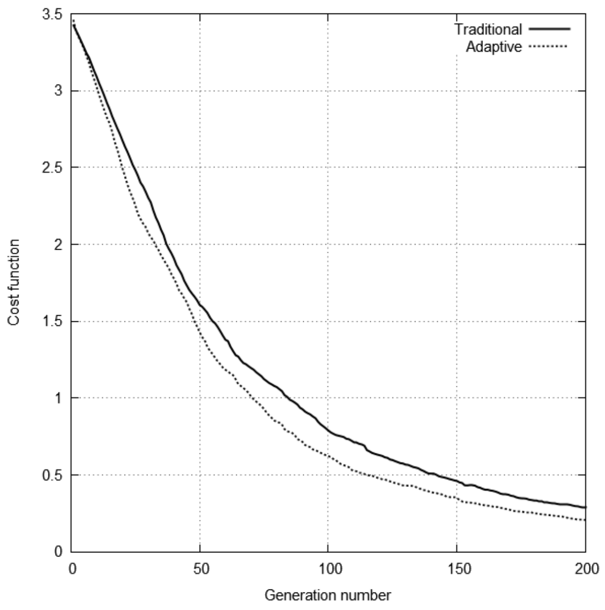

Fig. 4. Average convergence for the first generations covering traditional Real-Value CGP and adaptive Real-Value CGP with specific settings on $f_1(x) = x^6 - 2x^4 + x^2$.

of the average convergence between the traditional Real-Value CGP and our adaptive approach with specific settings for the first generations. Our adaptive approach converges faster and reaches the convergence criteria faster as illustrated in Figure 5. Table III shows the result for the second regression problem. As illustrated, the number of generations until convergence and the computational effort is better in comparison to the traditional Real-Value-CGP. Figure 6 and 7 underline the faster convergence of our adaptive approach.

B. Experiment II

For the second experiment we use the same algorithm configuration as shown in Table I except the tournament size. The configuration of the tournament size is shown in Table IV. For the problems f_3 (Equation 17) and f_4 (Equation 18) we also use the general and specific set of probabilities as we did in our first experiment. For this experiment we use the plot style as shown in Figure 8 because the presented problems use high polynomials, which produce high fitness values. As a result the detailed convergence for the given problems is difficult to plot. Table V shows the result for the third regression problem and as illustrated the average generation number and the computational effort is also better. The adaptive selection pressure is beneficial in maintaining population diversity but also in processing the population towards the global optimum by adjusting high selection pressure at the right time. Table VI shows the result for the last regression problem which shows also a better result for our adaptive approach. Also the number of generations to convergence is better as shown in Figure 9. Our adaptive approach has also slow runs but prevents the occurrence of very slow runs.

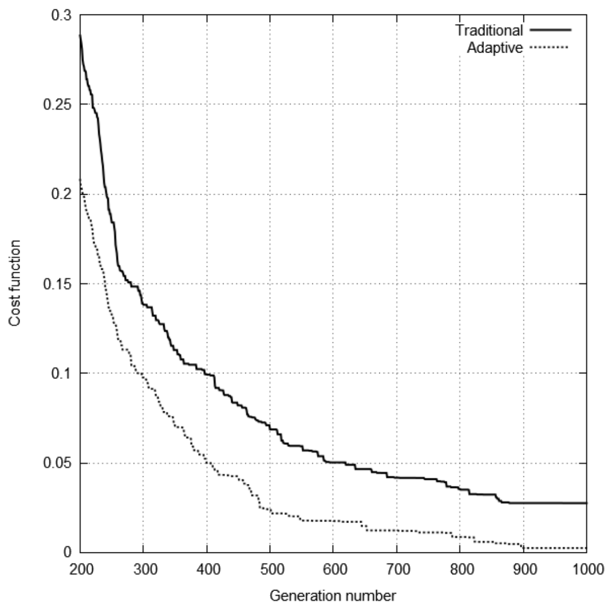

Fig. 5. Average convergence for the latter generations covering traditional Real-Value CGP and adaptive Real-Value CGP with specific settings on $f_1(x) = x^6 - 2x^4 + x^2$.

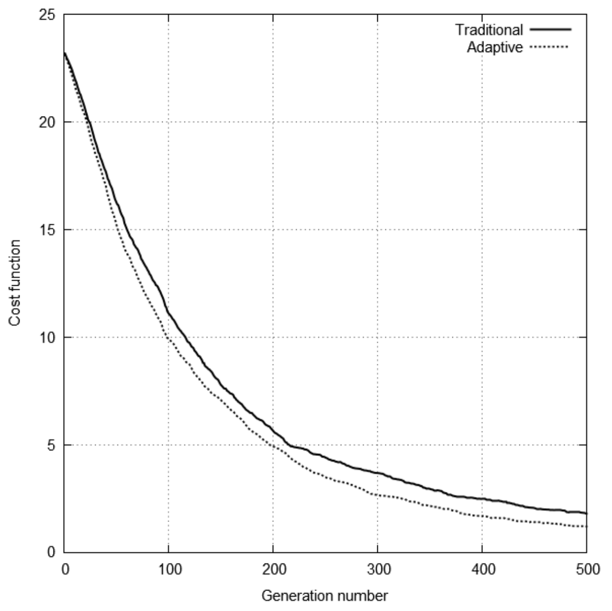

Fig. 6. Average convergence for the first generations covering traditional Real-Value CGP and adaptive Real-Value CGP with specific settings on $f_2(x, y) = (x^2 \cdot y^2)/(x + y)$.

TABLE IV. TOURNAMENT SIZE CONFIGURATION FOR THE SECOND EXPERIMENT

Algorithm	Tournament Size (T^{Size})
Traditional	8
Adaptive	4-10

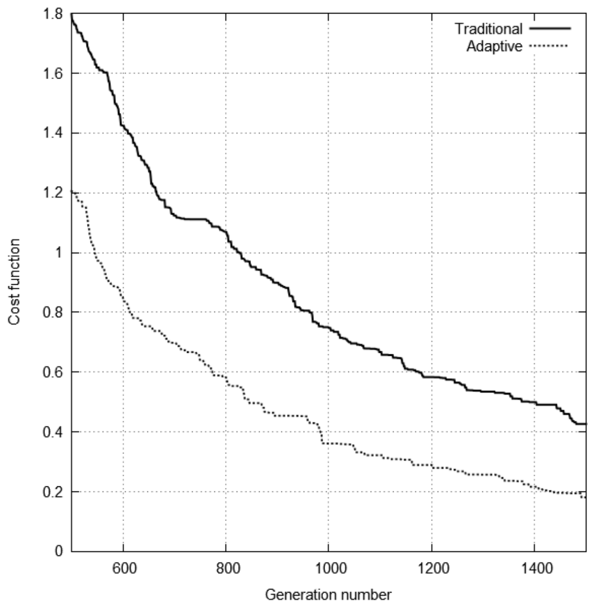

Fig. 7. Average convergence for the latter generations covering traditional Real-Value CGP and adaptive Real-Value CGP with specific settings on $f_2(x, y) = (x^2 \cdot y^2)/(x + y)$.

TABLE V. THE AVERAGE NUMBER OF GENERATIONS AND COMPUTATIONAL EFFORT REQUIRED BY CGP FOR THE PROBLEM $f_3(x) = x^5 - 2x^3 + x$

Algorithm	Avg. Generations	Min. CE	Slow Runs
Traditional	524 ± 1005	104965	126
Adaptive (general)	$351 \pm 564^{\dagger}$	70337	70
Adaptive (specific)	$349 \pm 551^{\dagger}$	69892	67

C. *Diversity Comparison*

Since $f_4(x)$ is our most complex problem we use for the results, we choose this problem for a diversity comparison in phenotype space. We average the diversity value θ (Equation 11) of 100 runs which exceed 500 generations for each generation. Based on our second Experiment, we use adaptive genetic operators and selection pressure for the adaptive strategy. Since the diversity value θ is normalized and ranges between 0 and 1, a number of 1 represents a phenotype space where every phenotype is unique. The diversity comparison is shown in Figure (10) and as visible the average diversity value of the adaptive algorithm is higher.

V. DISCUSSION

Our experiments on the four regression problems shows that using adaptive genetic operators and selection could be

TABLE VI. THE AVERAGE NUMBER OF GENERATIONS AND COMPUTATIONAL EFFORT REQUIRED BY CGP FOR THE PROBLEM $f_4(x) = x^4 + x^3 + x^2 + x$

Algorithm	Avg. Generations	Min. CE	Slow Runs
Traditional	1009 ± 1425	252378	328
Adaptive (general)	$713 \pm 909^{*}$	178296	265
Adaptive (specific)	$620 \pm 820^{\dagger}$	155246	217

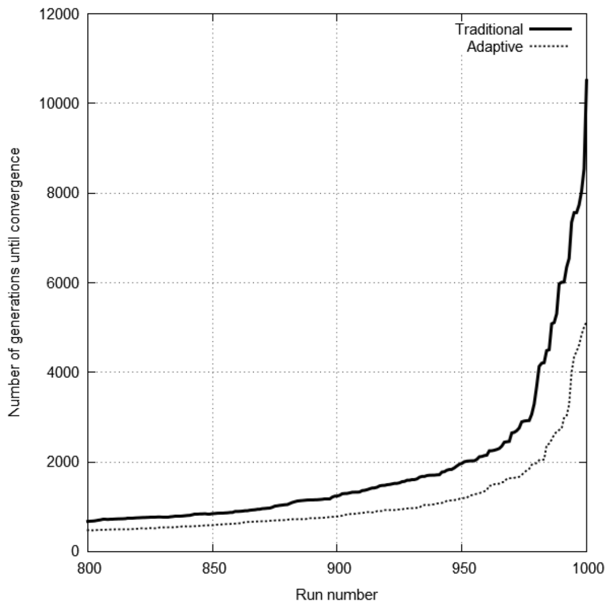

Fig. 8. The number of generations to convergence over the slowest 200 runs for traditional Real-Value CGP and adaptive Real-Value CGP with specific settings on $f_3(x) = x^5 - 2x^3 + x$.

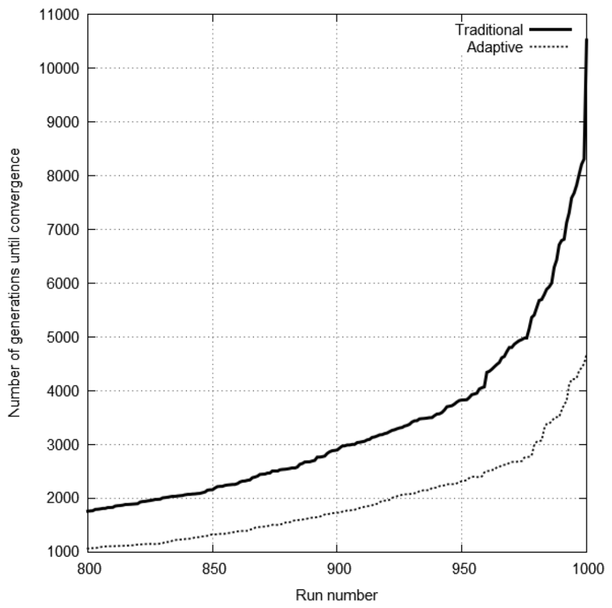

Fig. 9. The number of generations to convergence over the slowest 200 runs for traditional Real-Value CGP and adaptive Real-Value CGP with specific settings on $f_4(x) = x^4 + x^3 + x^2 + x$.

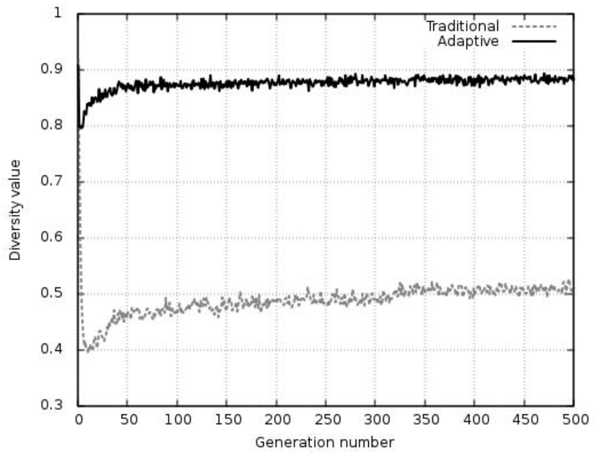

Fig. 10. Phenotype space diversity comparison for traditional Real-Value CGP and adaptive Real-Value CGP with specific settings on $f_4(x) = x^4 + x^3 + x^2 + x$.

beneficial for CGP. By maintaining population diversity, the occurrence of premature convergence could be prevented. As shown in our first experiment, our strategy improves the convergence in the former and latter generations by maintaining population diversity. In the first generations our strategy benefits from the ability of using higher probabilities of crossover. Using high crossover probabilities with traditional CGP increases the risk of the occurrence of premature convergence. The population converges to local optima and the effect of mutation for the exploration of the search space is too low. In this case our new strategy reacts with increasing the effect of mutation which prevents the populations getting too homogeneous. The use of a general set of parameters showed the flexibility of our new strategy when the optimal configuration is not set. Our adaptive strategy uses one new parameter which depends on the complexity of the problem and the size of the population. For the choice of the amplifier value we have found no general approach yet. Since this value was chosen empirically, we observed that an higher value for the harder problems of our tested regression problems was beneficial. Since modern evolutionary computation systems produce population statistics which can be used for our strategy, the determination of the phenotype space diversity becomes easier. Since our strategy handles with two types of population statistics, the use of our strategy with an GP system which has no build-in statistic module is difficult. But our analysis of the most used evolutionary computation systems showed that nearly every modern system produces statistics. This includes also fingerprints for genetic programming. Since the most genetic programming systems produce fingerprints for tree representation, the use of our new strategy in traditional GP is feasible. For the selection we only used the fitness value as selection criterion. In our experiments we showed that the crossover of unique and high fitted individuals is beneficial for CGP. For a better selection in our strategy, an improved selection method is necessary which works with two selection criteria, fitness and the frequency of the phenotype in the population. Using the textual representation as fingerprint is a simple form to map the phenotype space but has the disadvantage that textual different phenotypes are similar in their function. These behavior of neutrality in phenotype space has to be investigated in detail to develop more precise mapping techniques. In our experiments we only used four possible functions and ten nodes per CGP program but Miller and Smith showed that increasing node count improves convergence, too [16]. Further research is needed to investigate how our strategy can be applied on this behavior. Since our strategy is able to measure opportunities for high crossover probabilities, a combined use of the forking operator and our strategy is feasible.

VI. Conclusions and Future Work

A first adaptive strategy for CGP has been proposed. Our strategy maintains population diversity by adapting the probabilities of the genetic operators and selection pressure based on measurements in phenotype space. Also our new metric for measuring the ratio of healthy phenotypes helps to determine opportunities for the adaption. Our strategy works similar to the adaptive strategy of McGinley et al. [12] by adapting crossover probability globally and mutation probability locally. Since some adaptive shemes handle with a global mutation probability this approach showed no beneficial effect in our experiments. Also increasing mutation and crossover probability simultaneously if the population diversity becomes to less as proposed by Srinivas et al. [21], this behaviour influenced the convergence in our experiments negatively. Increasing mutation probability in homogeneous areas of the search space and reducing crossover probability if the population diversity is low as proposed by McGinley et al. [12] was the most successful approach to adapt the genetic operators. It has been shown that mapping phenotype space is beneficial for CGP on several symbolic regression problems so that our future work will focus on the improvement of these measurement techniques. Choosing the textual representation as fingerprint of a CGP program is the simplest form of mapping phenotype space. To improve the measurement of phenotype space, more detailed methods have to be found which are able to handle with phenotype neutrality which means that phenotypes which are different in their textual representation are semantically equal. Since we only benchmarked symbolic regression problems we will test our strategy on several image processing problems. Another focus of our further work lies on the development of a multi-objective selection method for our strategy which may lead to better convergence. Since the parametrization of our strategy has been determined empirically, we also intend to investigate methods for an automated determination. At this time, several effective approaches for CGP have been found [13], [4], [16]. For the future of CGP it will be important to investigate in which way different approaches are working together. As a first step we intend to perform experiments which include a combined use of the forking operator [13] and our adaptive strategy. Inspired by the forking operator which samples new individuals in the near of a solution, an adaption of the principle for our strategy is feasible. Further research is needed to investigate the use of an adaptive mutation operator with adaptive step size control.

VII. Acknowledgement

We thank Dominik Kopczynski from the Department of Computer Science of the TU Dortmund University who supported our work by giving advice for the improvement of the layout of this paper.

References

[1] J. Clegg, J. A. Walker, and J. F. Miller, "A new crossover technique for cartesian genetic programming," in *Proceedings of the 9th annual conference on Genetic and evolutionary computation*. ACM, 2007, pp. 1580–1587.

[2] N. L. Cramer, "A representation for the adaptive generation of simple sequential programs," in *Proceedings of the First International Conference on Genetic Algorithms*, 1985, pp. 183–187.

[3] L. Davis, "Adapting operator probabilities in genetic algorithms," in *International Conference on Genetic Algorithms\'89*, 1989, pp. 61–69.

[4] S. L. Harding, J. F. Miller, and W. Banzhaf, "Self-modifying cartesian genetic programming," in *Cartesian Genetic Programming*. Springer, 2011, pp. 101–124.

[5] J. H. Holland, "Genetic algorithms," *Scientific american*, vol. 267, no. 1, pp. 66–72, 1992.

[6] G. S. Hornby, "Alps: the age-layered population structure for reducing the problem of premature convergence," in *Proceedings of the 8th annual conference on Genetic and evolutionary computation*. ACM, 2006, pp. 815–822.

[7] J. Koza, *Genetic programming: on the programming of computers by means of natural selection*. MIT press Cambridge, 1992, vol. 1.

[8] J. R. Koza, *Genetic programming: A paradigm for genetically breeding populations of computer programs to solve problems*. Stanford University, Department of Computer Science, 1990.

[9] J. R. Koza and J. P. Rice, *Genetic programming II: automatic discovery of reusable programs*. MIT press Cambridge, 1994, vol. 40.

[10] W. B. Langdon, "A many threaded cuda interpreter for genetic programming," in *Genetic Programming*. Springer, 2010, pp. 146–158.

[11] H. B. Mann and D. R. Whitney, "On a test of whether one of two random variables is stochastically larger than the other," *The annals of mathematical statistics*, pp. 50–60, 1947.

[12] B. McGinley, J. Maher, C. O'Riordan, and F. Morgan, "Maintaining healthy population diversity using adaptive crossover, mutation, and selection," *Evolutionary Computation, IEEE Transactions on*, vol. 15, no. 5, pp. 692–714, 2011.

[13] A. Meier, M. Gonter, and R. Kruse, "Accelerating convergence in cartesian genetic programming by using a new genetic operator," in *Proceedings of the 15th annual conference on Genetic and evolutionary computation*. ACM, 2013, pp. 981–988.

[14] J. Miller, *Cartesian genetic programming*. Springer, 2011.

[15] J. F. Miller, "An empirical study of the efficiency of learning boolean functions using a cartesian genetic programming approach," in *Proceedings of the Genetic and Evolutionary Computation Conference*, vol. 2, 1999, pp. 1135–1142.

[16] J. F. Miller and S. L. Smith, "Redundancy and computational efficiency in cartesian genetic programming," *Evolutionary Computation, IEEE Transactions on*, vol. 10, no. 2, pp. 167–174, 2006.

[17] J. F. Miller and P. Thomson, "Cartesian genetic programming," in *Genetic Programming*. Springer, 2000, pp. 121–132.

[18] R. Poli, "Parallel distributed genetic programming," School of Computer Science, University of Birmingham, Tech. Rep., 1999.

[19] J. D. Schaffer and A. Morishima, "An adaptive crossover distribution mechanism for genetic algorithms," in *Genetic Algorithms and their Applications: Proceedings of the Second International Conference on Genetic Algorithms*. Hillsdale, NJ: Lawrence Erlbaum Associates, Inc, 1987, pp. 36–40.

[20] K. Slanỳ, "Comparison of cgp and age-layered cgp performance in image operator evolution," in *Genetic Programming*. Springer, 2009, pp. 351–361.

[21] M. Srinivas and L. M. Patnaik, "Adaptive probabilities of crossover and mutation in genetic algorithms," *Systems, Man and Cybernetics, IEEE Transactions on*, vol. 24, no. 4, pp. 656–667, 1994.