Systems Identification, Genetic Programming, Genetic Diversity

Stephan WINKLER[*], Michael AFFENZELLER[**], Stefan WAGNER[**]

VARIABLES DIVERSITY IN SYSTEMS IDENTIFICATION BASED ON EXTENDED GENETIC PROGRAMMING

There are several methods that are frequently used for solving data based system identification problems; Genetic Programming (GP) has already been used successfully for solving data mining problems in the context of several scientific domains. Extended functional bases, additional optimization phases and further developed selection mechanisms essentially contribute to the method's ability to generate high quality results for various kinds of data based identification scenarios. Even though there has already been a lot of investigation regarding the optimization of the method and its parameter settings, there is still rather little systematic analysis of internal processes regarding genetic dynamics and the progress of genetic diversity during the execution of Genetic Programming based identification using these algorithmic extensions.
In this paper we report on results of investigations regarding exactly these aspects: We have developed methods and statistical features that are able to describe genetic diversity and dynamics of GP-based structure identification algorithms; in this paper we introduce statistic analysis of genetic diversity regarding variables and time offset settings within GP populations. Genetic diversity is (amongst other aspects) characterized by the occurrence of variables for the models in which they are used; statistical methods for estimating respective impact features are also presented here. Data sets representing various kinds of systems (complex mechatronical systems as well as medical benchmark data) have been used for empirical tests; furthermore, standard implementations of Genetic Programming are compared to extended techniques including Offspring Selection as well as sliding window techniques.

1. DATA BASED SYSTEMS IDENTIFICATION USING GENETIC PROGRAMMING

Genetic Programming based techniques have been used successfully for solving data based structure identification problems in various different areas of data mining. Within the last years we have set up a further developed, fully automated and problem domain independent GP based structure identification framework that has been successfully used in the context of various different kinds of identification problems. For example, models describing the NO_x emissions of diesel engines have been evolved and tested successfully (as described in [10] and [3], e.g.). Classification problems have also been attacked yielding very satisfying results for medical benchmark data sets [11] as well as data-based estimators for the quality of the results of steel production processes [12].

[*] Upper Austrian University of Applied Sciences, Campus Hagenberg, Research Center;
Softwarepark 11 / 21, 4232 Hagenberg, Austria; stephan@heuristiclab.com
[**] Upper Austrian University of Applied Sciences, Campus Hagenberg, Department of Software Engineering;
Softwarepark 11 / 21, 4232 Hagenberg, Austria; {michael,stefan}@heuristiclab.com

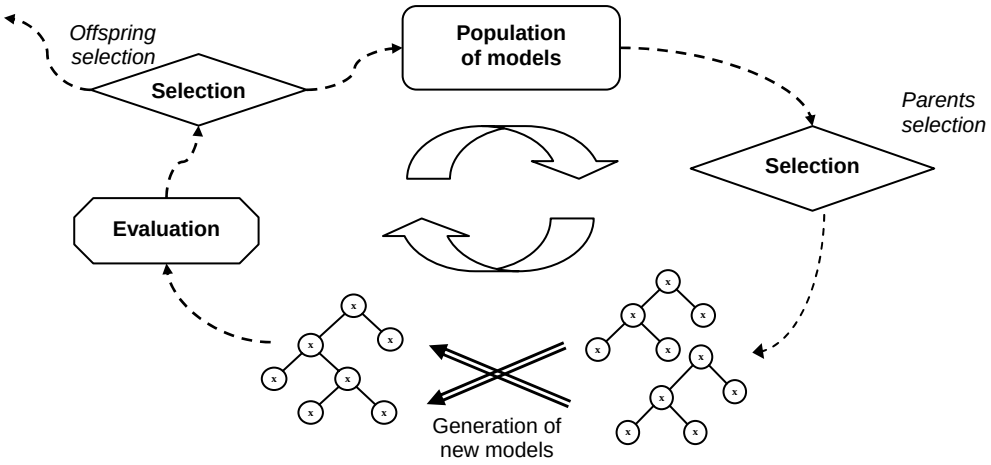

Fig. 1. The extended GP cycle

The fundamental principles of the Genetic Algorithm (which provides major parts of the theoretical basis of GP) were first presented by Holland [6], overviews about GAs and their implementation in various fields are given for instance in [5]. A GA works with a set of candidate solutions (also known as individuals) called population. During the execution of the algorithm each individual has to be evaluated, and new individuals are created on the one hand by combining the genetic make-up of two parent solution candidates (this procedure is called “crossover”), and on the other hand by mutating some individuals, i.e. changing randomly chosen parts of genetic information (which is normally applied on a minor ratio of the population). The third decisive aspect of Genetic Algorithms is selection: Usually, the individual’s probability to propagate its genetic information to the next generation is proportional to its fitness; the better a solution candidate's fitness value, the higher the probability, that its genetic information will be included in the next generation's population. This procedure of crossover, mutation and selection is repeated over many generations until some termination criterion is fulfilled.

Similar to GAs, Genetic Programming (GP, [7], [8]) works by imitating aspects of natural evolution: A Population of solution candidates evolves through many generations towards a solution using evolutionary operators (crossover and mutation) and a “survival-of-the-fittest” selection scheme. The goal of a GP process is to produce a computer program (or, as in our case, a formula) solving the problem at hand; thus, GP provides a way to successfully conduct the search for a computer program in the space of computer programs [7]. The overall procedure is graphically illustrated in Figure 1 (showing also Offspring Selection [2] which is in fact not part of the standard GP process).

Even though it is well known that every model consists of an equation set (the structure) and of values (parameters) and that system identification actually implies both, still usually the definition of the structure is considered either obvious or as the less critical issue, while the consistent estimation of the parameters especially in presence of noise receives most attention. By its very general problem statement, GP allows to approach the problem of structure identification and the problem of parameter identification simultaneously. As a consequence, GP techniques are used for identifying various kinds of technical systems; the goal of a GP-based structure identification process therefore is to identify a model reproducing the system’s target (output) data. The identification algorithm is executed using solution candidates which represent mathematical models (formulae); Figure 2 shows examples of genetic operations (single point crossover and single point mutation) on exemplary model structure trees representing nonlinear models.

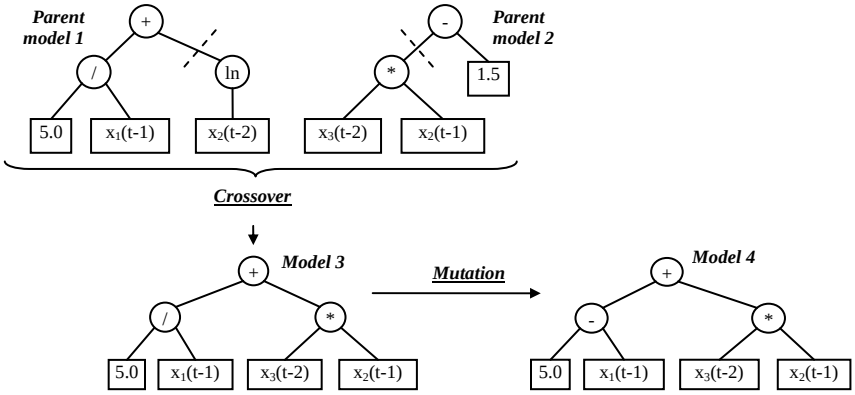

Fig. 2. Exemplary formula trees and genetic operations applied on them: The crossover of the two parent models 1 and 2 produces model 3; mutating this model for example produces model 4.

Within the last years, more and more sophisticated algorithmic concepts for GAs and GP have been developed and implemented also within the HEAL research group[1]; several new hybrid evolutionary concepts have been combined, one of the most important ones being the so-called Offspring Selection concept ([2] and [1]). The basic idea of Offspring Selection is that individuals are first compared to their own parent solution candidates and accepted as members of the new generation's population if they meet certain criteria. In the context of structure identification and machine learning we have realized that the use of very rigid settings yields best results ([11], [13]). I.e., new models are kept and thus inserted in the next generation's population only if they outperform their own parent individuals. I.e., optimization is not restricted to the chromosome level but also incorporates allele level information. In fact, this Offspring Selection principle is integrated in the GP cycle displayed in Figure 1 even though it is not a part of the standard GP workflow.

Apart from Offspring Selection, pruning, parameters optimization and sliding window behavior represent some of the most important additional optimization phases integrated into the GP process. In addition to an extended functional basis providing flexible variable and terminal definitions, there is a strongly increasing interest in analyzing the genetic diversity that is existent in the population. In the following sections we describe statistical methods which can be used to do so and exemplary demonstrate how the results are to be displayed and interpreted.

2. MEASURING GENETIC DIVERSITY IN GP POPULATIONS

For measuring the genetic diversity in GP with respect to the variables used we have developed the following features that are able to measure the genetic diversity within a population:

- A very simple approach is to calculate an occurrence feature for each data variable (in the case of time series analysis also considering each possible time offset) as the number of models that include the respective variable. Obviously, this approach can easily be transferred to function definitions as well as terminal definitions.
- As a first extension to this approach, the quality of the models has to be incorporated into this calculation model as for example by multiplying the occurrence values with a weighting factor (which depends on the model's quality in relation to the quality of all other models which are included in the current generation's population).

[1] HEAL: Heuristic and Evolutionary Algorithms Laboratory; website: http://www.heuristiclab.com

- The most informative (and, unfortunately, also most run-time consuming) calculation model takes into account the impact of variables by evaluating all models assuming that all information included in the respective data variables was deleted temporarily. There are several possibilities how to remove information from a variable, for example by replacing all values by the mean value of the respective training data samples or a given constant, by using linear regression for calculating the variables' trend (again using the respective training data samples) or by adding a synthetic Gaussian noise.

2.1. CALCULATING THE RELEVANCE OF VARIABLES WITH RESPECT TO A MODEL

In a first step, the relevance *rel* for each variable with respect to each model of the current population has to be calculated; this is done either by frequency analysis or by measuring its impact by evaluating it on modified data bases.

2.1.1. FREQUENCY BASED ANALYSIS OF VARIABLES AND OFFSETS

The relevance of a variable (at index *i*) with respect to a given model M can either be defined as the number of references in this model (calculated using function $freq_1$) to this variable or simply as 1 if there is a reference to this variable and 0 if not ($freq_2$). All terminals *t* in the model are considered for this.

$$freq_1(i,M) = |\{t \mid t \in M.terminals \,\&\, t.VarIndex = i\}| \tag{1}$$

$$freq_2(i,M) = \begin{cases} 1 & if\ \exists t : (t \in M.terminals \,\&\, t.VarIndex = i) \\ 0 & else \end{cases} \tag{2}$$

In the case of time series analysis this variable frequency analysis can be extended to the analysis of time lags as variables are possibly referenced using time offsets (as for example in $f(x) = u_{(t-2)} * v_{(t-1)}$). Thus, we calculate the frequency based relevance of a variable (at index *i*) with time offset t with respect to a model M as follows:

$$freq_1(i,t,M) = |\{t \mid t \in M.terminals \,\&\, t.VarIndex = i \,\&\, t.TimeOffset = t\}| \tag{3}$$

$$freq_2(i,t,M) = \begin{cases} 1 & if\ \exists t : (t \in M.terminals \,\&\, t.VarIndex = i \,\&\, t.TimeOffset = t) \\ 0 & else \end{cases} \tag{4}$$

2.1.2. IMPACT BASED ANALYSIS OF VARIABLES

For estimating a variable's impact to the evaluation of a model we temporarily replace this variable's values and evaluate the model on these manipulated data. Thus, first replacement strategies have to be designed; we here present methods using averaging, constants, linear regression and additive Gaussian noise. Thus, a given variable (at index *i*) is replaced without changing any other part of the data basis; we transfer the original data basis *Data* consisting of *N* variables with *n* samples each to a manipulated data basis $Data_i$ with manipulated variable no. *i* using a given replacement function *r* as

$$Data_i(r) = \{Data_1, Data_2, \ldots, Data_{i-1}, r(Data_i), Data_{i+1}, \ldots, Data_N\}. \tag{5}$$

The replacement functions introduced here realize the replacement of a given variable using a given constant value (r_{const}), its mean value (r_{mean}), its linear trend (r_{linreg}) and the addition of

Gaussian noise (r_{agn}). Whereas the first two methods are rather straightforward (6, 7), the other two ones are more complex: For replacing a variable with its linear trend, we calculate the parameters needed for linear regression using the method of minimizing the sum of squared errors [4] (8 – 13), and a random number generator is needed for adding Gaussian noise (additionally, we also use the respective variable's range and a scaling factor σ) (14, 15):

$$\forall j \in [1,n] : [r_{const}(Data_i, c)]_j = c \tag{6}$$

$$\forall j \in [1,n] : [r_{mean}(Data_i)]_j = \frac{1}{n}\sum_{j=1}^{n} Data_i \tag{7}$$

$$\bar{j} = \frac{1}{n}\sum_{j=1}^{n} j = \frac{n+1}{2} \tag{8}$$

$$\forall j \in [1,n] : y_j = [Data_i]_j,\ \bar{y} = \frac{1}{n}\sum_{j=1}^{n}[Data_i]_j = \frac{1}{n}\sum_{j=1}^{n} y_j \tag{9), (10}$$

$$b = \frac{\frac{1}{n}\sum_{j=1}^{n}(j-\bar{j})(y_j-\bar{y})}{\frac{1}{n}\sum_{j=1}^{n}(j-\bar{j})^2} = \frac{\sum_{j=1}^{n}(j-\bar{j})(y_j-\bar{y})}{\sum_{j=1}^{n}(j-\bar{j})^2} \tag{11}$$

$$a = \bar{y} - b\bar{j} \tag{12}$$

$$[r_{linreg}(Data_i)]_j = a + b \cdot j \tag{13}$$

$$\forall j \in [1,n] : range_i = \max(Data_i) - \min(Data_i) \tag{14}$$

$$[r_{agn}(Data_i, RG, \sigma)]_j = [Data_i]_j + range_i * RG.next() * \sigma \tag{15}$$

Now it is possible to calculate the variable's impact with respect to the model *M* by evaluating the model on the manipulated data set $Data_i$ and measuring the resulting difference between the original output values and those calculated on the manipulated data. This measurement can be done on the basis of the average absolute difference function $impact_{mad}$, the mean squared difference function $impact_{msd}$, $impact_{cc}$ using the correlation coefficient and $impact_{ff}$ using a (predefined) external fitness function *FF*.

The first two functions measure the sample-wise difference between the evaluations on the original and manipulated data, respectively:

$$impact_{mad}(i,M) = \frac{1}{n}\sum_{j=1}^{n}\left|[eval(M, Data_i)]_j - [eval(M, Data)]_j\right| \tag{16}$$

$$impact_{msd}(i,M) = \frac{1}{n}\sum_{j=1}^{n}\left([eval(M, Data_i)]_j - [eval(M, Data)]_j\right)^2 \tag{17}$$

The correlation coefficient *cc* is a nondimensional measure for the linear interrelationship between two variables. Its domain is [-1; +1] with cc=+1 indicating a perfect positive and cc=-1 a perfect negative linear relation between the signals investigated; if the correlation coefficient equals 0, the variables show no linear correlation at all.

Thus, the correlation based method $impact_{cc}$ for calculating the impact of a variable i with respect to a given model M can be calculated in the following way: The model is on the one hand evaluated on the (whole) manipulated data set yielding X and on the other hand on the original data set yielding Y; within the analysis approach presented here the absolute value of the correlation coefficient of X and Y is returned as the resulting impact value:

$$cc(X,Y) = \frac{\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2} \cdot \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(y_i - \bar{y})^2}} \tag{18}$$

$$\begin{aligned} &impact_{cc}(i,M): \\ &\quad \forall j \in [1,n]: X_j = eval(M, Data_i), Y_j = eval(M, Data) \\ &\quad impact_{cc}(i,M) = 1 - |cc(X,Y)| \end{aligned} \tag{19}$$

Finally, it also seems to be a good idea to calculate the impact of a variable i with respect to a given fitness function FF. The main idea here is that (for example in the context of evaluating classifier models) it is well possible that the manipulation of a certain variable results in significant changes of the output values of models, but the evaluation using some given fitness measure does not change to the same extent (for example because the ratio of correctly classified samples is no changed). So, again the manipulated as well as the original data set are evaluated using the given model M[2]; the resulting output values are compared to the original target values stored in the data base, and the impact factor is calculated as the ratio between the original evaluation and the evaluation on the basis of the manipulated data set:

$$\begin{aligned} &impact_{ff}(i,M,FF): \\ &\quad \forall j \in [1,n]: X_j = eval(M, Data_i), Y_j = eval(M, Data), T_j = [Data_t]_j \\ &\quad q = eval(T,X,FF) / eval(T,Y,FF) \\ &\quad impact_{ff}(i,M,FF) = \begin{cases} q & if\ (q \geq 1) \\ 1/q & if\ (q < 1) \end{cases} \end{aligned} \tag{20}$$

2.2. WEIGHTING OF VARIABLES RELEVANCE ESTIMATIONS

Before calculating the overall relevance of a variable with respect to a population of models it is possible to calculate weighting factors for each model depending on its quality:

$$\begin{aligned} &\forall j \in [1, length(M)]: q_j = quality(M_j) \\ &qMin = \min(q),\ qMax = \max(q),\ qRange = qMax - qMin \\ &\forall j \in [1, length(M)]: weighting(M_j) = 1 - \frac{(quality(M_j) - qMin)}{qRange} \end{aligned} \tag{21}$$

[2] Of course, here the following issue has to be kept in mind: If the system's target variable is to be analyzed and therefore manipulated, the evaluation of the model using the given fitness function still has to be done on the original (not manipulated) values of this variable.

2.3. CALCULATING THE RELEVANCE OF VARIABLES IN POPULATIONS

Finally, for each variable the (potentially weighted) relevance values calculated for each model are summarized and so give a measure for the overall relevance of this specific signal. By calculating this relevance measure for all variables we finally get an analysis for the genetic diversity within the given population. So, on the basis of all relevance values for each variable (with index *i*) on each model *M* (22) and after possibly weighting them (23) we calculate the relevance of a variable with index *i* with respect to a whole population *P* as given in Formula (24):

$$rel(i,M) = \begin{cases} freq(i,M) & if\ (frequencyBasedAnalysis) \\ impact(i,M) & else \end{cases} \tag{22}$$

$$rel'(i,M) = \begin{cases} rel(i,M) * weighting(M) & if\ (weightingActivated) \\ rel(i,M) & else \end{cases} \tag{23}$$

$$rel'(i,P) = \sum_{\forall M \in P} rel'(i,M) \tag{24}$$

3. APPLICATION EXAMPLES

For testing the analysis methods described in the previous section and illustrating graphical representations of the results of these tests we have used two data sets:

- The *NOx* data set contains the measurements taken from a 2 liter 4 cylinder BMW diesel engine at a dynamical test bench (simulated vehicle: BMW 320d Sedan). Several emissions (including NO_x, CO and CO_2) as well as several other engine parameters were recorded over approximately 30 minutes and downsampled to 20 Hz. 40 signals were recorded, but only 9 variables were considered by the identification algorithm. The reason for this is that information about other emissions should not be incorporated in the model because of redundancies and relatively high costs of exhaust sensors – an emission model using other emission measurements is much easier to be found, but not very significant. Therefore we have only used parameters which are directly measured from the engine's control unit and not in any sense connected to emissions (as for example oil temperature, air pressure, injection parameters etc.). We cordially thank members of the Institute for Design and Control of Mechatronical Systems at JKU, Linz[3] who provided and helped us with these data.
- The *Thyroid* data set is a widely used machine learning benchmark data set containing the results of medical measurements which were recorded while investigating patients potentially suffering from hypotiroidism[4].

Both data collections have been split into training and validation / test data partitions taking the first 80% of each data set as training samples available to the identification algorithm. The HeuristicLab framework [10], a generic and extensible optimization framework developed by members of the HEAL research group[1], was used as underlying basic framework. All tests were executed using 12% mutation rate and single point crossover and mutation operators; the most relevant test settings are summarized in Table 1.

[3] The homepage of the Institute for Design and Control of Mechatronical Systems at the Johannes Kepler University, Linz can be found at http://desreg.jku.at/.

[4] Further information about the data set used can be found on the UCI homepage (http://www.ics.uci.edu/~mlearn/).

Test	Data Set	GP Algorithm	Population Size	Other Parameter Settings
I	Thyroid	Standard GP	1000	Number of generations = 1,500
II	NOx	Offspring Selection GP	500	MSP = 200, Success Ratio = 1.0
III	NOx	Offspring Selection GP	500	MSP = 200, Success Ratio = 0.9
IV	Thyroid	Offspring Selection GP, Sliding Window GP	1000	Sliding Window Moving MSP = 20, MSP = 200, Success Ratio = 1.0

Table 1. Summary of data and algorithm specific settings of the GP tests investigated. MSP hereby stands for "maximum selection pressure", the "success ratio" parameter gives the ratio of new individuals that have to fulfill the Offspring Selection requirements for building up the next generation's population; the rest of the population is filled up with not necessarily successful individuals (as is explained in further detail in [2]).

Test case IV was taken from the test series executed in the course of investigations of sliding window behavior for GP [14]. The main difference here is that the algorithm starts considering only a part of the training data available; after reaching a certain maximum selection pressure, the data scope used for evaluating models is shifted until the end of the data set is reached. This approach significantly increases the speed of GP based structure identification as well as it helps the method to avoid overfitting. The test case analyzed here is part of test series 5 explained in detail in [14].

First we report on the results obtained using standard GP: In Figures 3, 4, and 5 we illustrate selected analysis results for test run I. Figure 3 shows the impact of all variables over time using the "mean" replacement strategy (7) and the sum of squared differences function (17) for calculating the impact values. Figures 4 and 5 show the total number of occurrences for all variables at generations 1400 and 1450, respectively. There is obviously no notable variables selection process and also no clear statement possible regarding which variables are more important for modeling the given target variable than others.

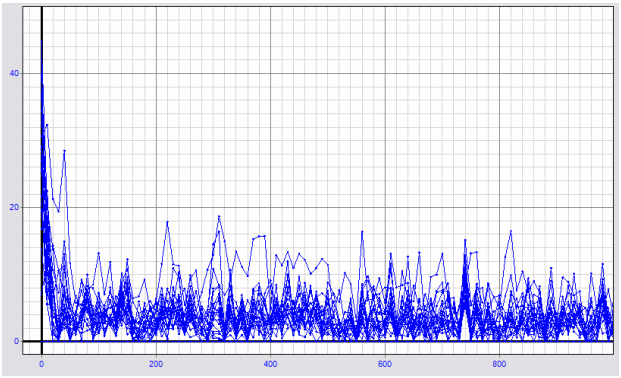

Fig. 3. Test run I: The impact of all variables is shown over time for the first 1000 generations.

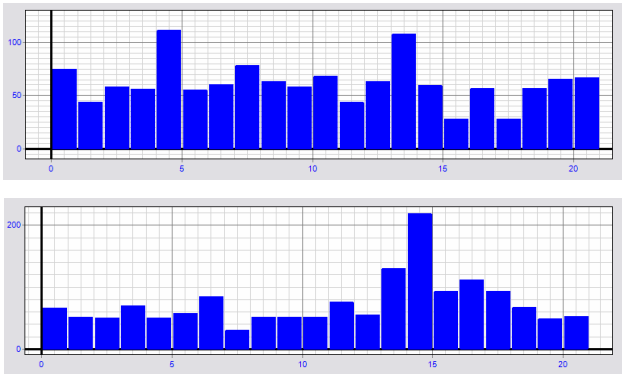

Fig. 4 and 5. Test run I: Total occurrences at generations 1400 and 1450 (in the upper and lower figure, respectively).

In Figures 6, 7, and 8 we visualize the impact of all variables for test run II. In Figure 6 we show the impact using the "linear regression" (13) replacement strategy and the correlation coefficient (19) impact function; Figures 7 and 8 show the variables' impact at the end of the algorithm's execution (based on r_{mean} / $impact_{msd}$ and r_{linreg} / $impact_{corr}$ strategies, respectively). Here the variables selection functionality of GP becomes obvious, still the results differ quantitatively depending on the selection of replacement and impact strategies applied. The Figures 9 and 10 characterize the algorithmic behavior in test run III: Even though several variables occur rather often in the population, only one variable dominates all other ones with respect to occurrence (Fig. 9) as well as impact (applying r_{linreg} and $impact_{corr}$ strategies, Fig. 10).

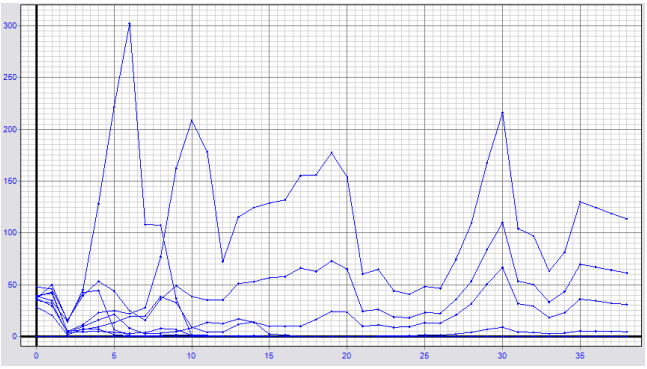

Fig. 6. Test run II: The impact of all variables over time.

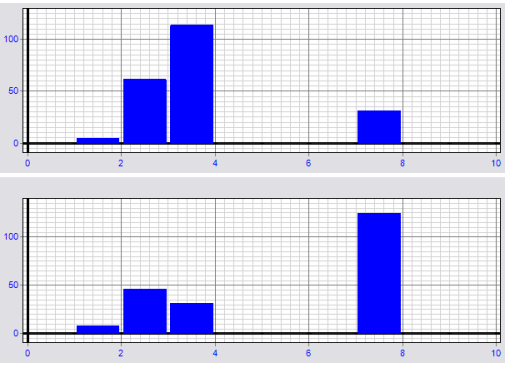

Fig. 7 and 8. Test run II: Final impact analysis.

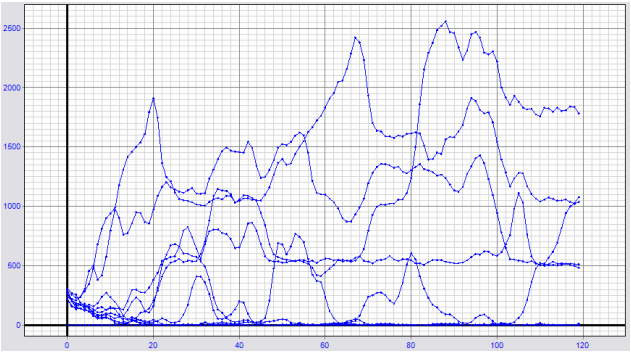

Fig. 9. Test run III: Occurrence of variables over time.

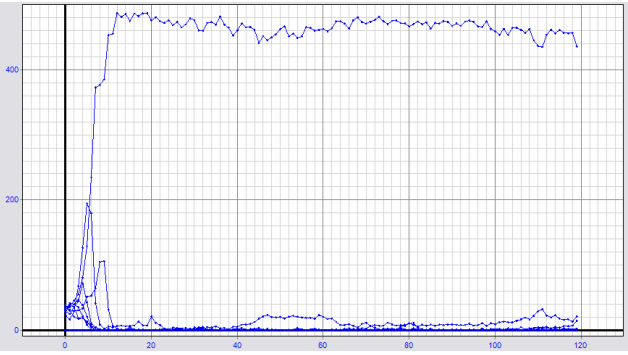

Fig. 10. Test run III: Impact of variables over time.

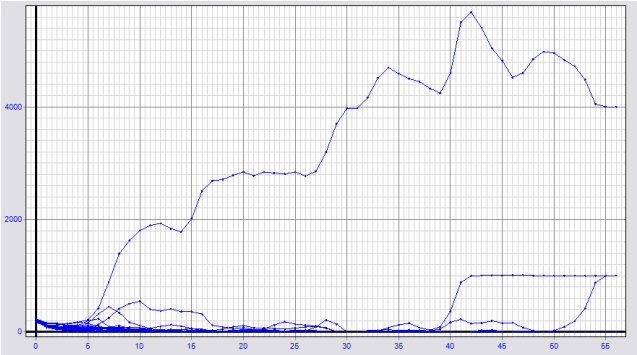

Fig. 11. Test run IV: Occurrences of variables over time.

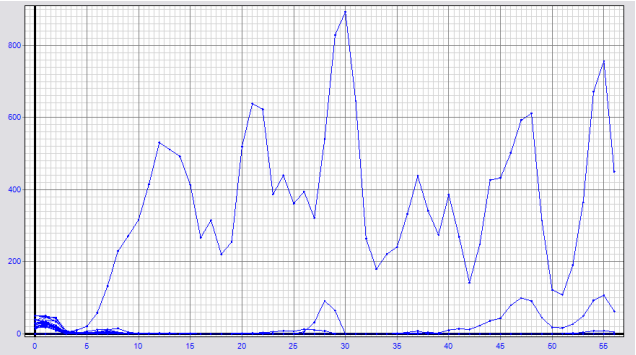

Fig. 12. Test run IV: Impact of variables over time.

Figures 11 and 12 finally show the total occurrences of all variables during the execution of test run IV and the impact of variables (again applying the r_{mean} / $impact_{msd}$ strategy). Here it is even more obvious that one variable dominates all other ones; genetic diversity almost seems to have disappeared since all models only use one variable and simply seem to neglect all other ones.

4. SUMMARY, DISCUSSION

In this paper we have introduced a collection of statistical functions that can be used for estimating the genetic diversity in Genetic Programming based structure identification. The frequency and the impact of variables with respect to the evaluation of the models included in the respective algorithm's population; by observing these properties for all variables over time during

the execution of the identification algorithm we obtain a statistical analysis of genetic diversity and dynamics. Using these measures we have analyzed several parameter settings for GP based model identification using a mechatronical as well as a biomedical data set.

On the one hand, this analysis (in combination with appropriate graphical representations) provides an easily useable tool for demonstrating the effects of the implicit variables selection functionality automatically included in GP based modeling. On the other hand, the authors are confident that these statistical measures will enable further analysis of genetic dynamics in Genetic Programming, for example in the context of robustness analyses or multi-population aspects.

ACKNOWLEDGEMENTS

The work described in this paper was done within the Translational Research Project L284 “GP-Based Techniques for the Design of Virtual Sensors” sponsored by the Austrian Science Fund (FWF). This project is executed as a joint venture by the Upper Austrian University of Applied Sciences, Campus Hagenberg, and the Johannes Kepler University Linz, Austria.

REFERENCES

[1] AFFENZELLER M., *Population Genetics and Evolutionary Computation: Theoretical and Practical Aspects*, Schriften der Johannes Kepler Universität Linz, Universitätsverlag Rudolf Trauner, ISBN 3-85487-823-0, 2005.

[2] AFFENZELLER M., WAGNER S., *Offspring Selection: A New Self-Adaptive Selection Scheme for Genetic Algorithms*, Adaptive and Natural Computing Algorithms, Springer Computer Science, 2005, pp. 218-221.

[3] ALBERER D., DEL RE L., WINKLER S., LANGTHALER P., *Virtual Sensor Design of Particulate and Nitric Oxide Emissions in a DI Diesel Engine*, Proceedings of the 7th International Conference on Engines for Automobile ICE 2005, Capri, Napoli, 2005, paper nr. 2005-24-063.

[4] DRAPER N.R., SMITH H., *Applied Regression Analysis*, Wiley, New York, 1998.

[5] GOLDBERG D.E., *Genetic Algorithms for Search, Optimization, and Machine Learning*, Addison-Wesley, Reading, 1989.

[6] HOLLAND J.H., *Adaptation in Natural Artificial Systems*, University of Michigan Press, Ann Arbor, 1975.

[7] KOZA J.R., *Genetic Programming: On the Programming of Computers by Means of Natural Selection*, The MIT Press, ISBN 0-262-11170-5, 1992.

[8] LANGDON B., POLI R., *Foundations of Genetic Programming*, Springer, 2002.

[9] WAGNER S., AFFENZELLER M., *HeuristicLab: A Generic and Extensible Optimization Environment*. Adaptive and Natural Computing Algorithms, Springer Computer Science, 2005, pp. 538-541, Springer.

[10] WINKLER S., AFFENZELLER M., WAGNER S., *New Methods for the Identification of Nonlinear Model Structures Based Upon Genetic Programming Techniques*, Journal of Systems Science, Vol. 31 (2005), Oficyna Wydawnicza Politechniki Wroclawskiej, pp. 5-13.

[11] WINKLER S., AFFENZELLER M., WAGNER S., *Advances in Applying Genetic Programming to Machine Learning, Focussing on Classification Problems*, Proceedings of the 20th IEEE International Parallel & Distributed Processing Symposium IPDPS 2006, IEEE Catalog Number 06TH8860, paper nr. NIDISC-012.

[12] WINKLER S., EFENDIC H., DEL RE L., *Quality Pre-Assessment in Steel Industry Using Data Based Estimators*, Proceedings of the IFAC Workshop MMM’2006 on Automation in Mining, Mineral and Metal Industry, International Federation for Automatic Control (IFAC) 2006, pp. 185-190.

[13] WINKLER S., AFFENZELLER M., WAGNER S., *Advanced Genetic Programming Based Machine Learning*, Journal of Mathematical Modelling and Algorithms, ISSN 1570-1166 (print), 1572-9214 (online), DOI 10.1007/s10852-007-9065-6, Springer Netherlands, 2007.

[14] WINKLER S., AFFENZELLER M., WAGNER S., *Selection Pressure Driven Sliding Window Genetic Programming*, accepted to be published in Proceedings of Computer Aided Systems Theory: EuroCAST 2007, Springer Lecture Notes in Computer Science, 2007.