An Study of the Tree Generation Algorithms in Equation Based Model Learning with Low Quality Data

Alba Berzosa[1], José R. Villar[2], Javier Sedano[1], Marco García-Tamargo[2], and Enrique de la Cal[2]

[1] Instituto Tecnológico de Castilla y León, Lopez Bravo 70, Pol.Ind.Villalonquéjar 09001 Burgos (SPAIN) {alba.berzosa, javier.sedano}@itcl.es
[2] Computer Science Department, University of Oviedo, Campus de Viesques s/n 33204 Gijón (SPAIN) {villarjose, marco, delacal}@uniovi.es

Abstract. The undesired effects of data gathered from real world can be produced by the noise in the process, the bias of the sensors and the presence of hysteresis, among other uncertainty sources. In previous works the learning models using the so-called Low Quality Data (LQD) has been studied in order to analyze the way to represent the uncertainty. It makes use of genetic programming and the multiobjective simmulated annealing heuristic, which has been hybridized with genetic operators. The role of the tree generation methods when learning LQD was studied in that paper. The present work deals with the analysis of the generation methods relevance in depth and provides with statistical studies on the obtained results.

Keywords: Genetic Programming, Genetic Algorithm and Programming, Low Quality Data, Multiobjective Simulated Annealing

1 Introduction

With the scarce energy sources and the worsening environmental pollution, how to use the existing energy is becoming a very important challenge in various fields of modern engineering [**?**,**?**,**?**]. For example, notorious efforts have been made within the area of lighting control systems, whose aim is to control the electrical power consumption for the ballast in the installation so the luminance complies with the regulations. In [**?**,**?**] a lighting control system was considered to show the relevance of the uncertainty for an efficient energy use. The typical control loop includes a light sensor, the light ballasts and a light controller. The sensors measure the amount of light in a room, but they have some drawbacks: they operate with hysteresis and saturation [**?**] and their measurements depend on the light sensor unit. In the studied literature, when obtaining models for simulation, only crisp values are regarded as the measurements from light sensors. Obviously, the inputs and outputs of the light sensor models obtained are also crisp variables. But several studies have presented the decrease in the performance of crisp algorithms as data uncertainty increases [**?**].

An approach for learning white box models when LQD is available is presented in [**?**,**?**], where the variables are assumed with an unknown uncertainty value modelled as a fuzzy number. The white box models include an equation -represented as a nodes tree- setting the relationship of the output variable with the feature space and a set of constants. Some of the constants are use to model the vagueness of the variables and others are used as terminal nodes in the equation. A genetic programming hybridized with Genetic Algorithm (hereafter GAP) which is evolved by means of a Multi Objective Simulated Annealing algorithm (hereinafter MOSA) is proposed, and a random tree generation algorithm to create the individuals is carried out in the evolutionary algorithm. A MOSA hybridized with genetic operators is proposed (hereinafter, SAP), and a random tree generation algorithm to create the individuals is carried out in the evolutionary algorithm. The results show that the proposed algorithm remails with the same performance index even though LQD is given. The relevance of three generation algorithms in MOSA hybridized with genetic operators is studied in [**?**]. The approach makes use of fuzzy fitness funcions. Consequently, the relevance of the tree generation methods is focused on the first generations due to the temperature grading and the need to search in the whole variables space. Two algorithms were compared. On the one hand, the so-called GROW GP tree generation algorithm, described in [**?**], chooses a node type with equal probability, including the terminal and non-terminal ones. On the other hand, [**?**] offers an alternative tree-creation algorithm, the Probabilistic Tree Creation 1 (PTC1), which gives the user control over the expected tree size, the maximum deth and the probabilities of appearance of functions within the tree, providing in addition, very low computational complexity.

The present work aims to extend it with the representation and comparison of the model learning evolution when genetic programming and uncertainty are considered for the two tree generation methods proposed, GROW and PTC1. For this purpose, an statistical study of the results has been done. The remainder of this manuscript is as follows. Firstly, a description of the simulated annealing approach for learning white box models with LQD is included. Then, the different tree generation methods employed in this comparison are detailed. The experiments and the results are discussed next, while in the last section the conclusions are drawn and the future work is proposed.

2 White Box Models SAP Learning With LQD

Soft Computing includes the set of techniques that allow learning models using the knowledge in the data [**?**,**?**,**?**,**?**]. There are several uncertainty sources in data gathered from real processes [**?**]. In this work, we study data which has been gathered as crisp data but that are highly imprecise, i.e., the data gathered from light sensors [**?**,**?**]. In [**?**] a SAP approach for learning white box models from this kind of data is presented.In that study, the representation of the vagueness in a GP model is represented by two constants C^- and C^+ assigned to each imprecise variable which evolves in the learning process. These constants represent the

limits of a triangular membership function for an α-cut= 0 which is associated with each imprecise variable. Let us suppose the training data set being the set $\{d_i^j\}$, with $i = \{0, \ldots, D\}$ for each of the D variables X_i and $j = \{1, \ldots, N\}$ and N the number of examples in the data set. Then, whenever an imprecise variable X_i is evaluated for the example j, a fuzzy number with a triangular fuzzy membership defined through the three following values $[d_i^j - C^-, d_i^j, d_i^j + C^+]$ is returned. If symmetrical membership functions are considered, only one constant per imprecise variable is needed. As in classical fuzzy logic literature, crisp values from constants or from crisp variables are extended to fuzzy singletons, so only operations with fuzzy numbers are required. In order to reduce the computational cost, the solution presented in [**?**] is used, and evaluations are calculated only for certain predefined α-cuts.

An individual in this study is a compound of the equation representation, the constants vector and the specification of the uncertainty, which is provided with the number of constants used to represent the uncertainty and the list of indexes of the input variables in the dataset that are supposed to manage LQD. As in Genetic Programming hybridized with Genetic Algorithms (hereinafter, GAP) models, the equation representation consists of a nodes tree, each internal node corresponds with a valid operator, and the leaf nodes correspond with a variable or a constant index. The number of constants is predefined, so the constant vector in all individuals has the same size. The first group of constants in the constant vector is assigned to the uncertainty management. The generation of a nodes tree is the well-known random strategy given by the GROW method [**?**].

Evolving GAP individuals introduce four genetic operators, two come from GP evolution (the GP crossover and mutation) and two come from GA (GA crossover and mutation). The GP operators introduce variability in the structure of the model, in other words, the equation itself. The GA operators modify the vector of constants. In all the cases, there is a predefined probability of carrying out each of these genetic operators. In each run, the type of operation to carry out is chosen, that is only GP or GA operations can take place, but never both in the same run. The fitness of an individual is calculated as the mean squared error, which in fact is a fuzzy number. To reduce the width of the intervals and to obtain models that include the desired output crisp data two more objectives have also been considered, so multi-objective techniques are needed. The evolutionary algorithm is the MOSA proposed in [**?**,**?**].

3 Tree-Generation Algorithms

Tree-creation plays an important role in GP algorithms since a good random tree-creation is needed to create the number of trees that will compose the initial population and the subtrees used in subtree mutation. Besides, as stated in [**?**], tree creation is also related with tree bloat, that is, the tendency of GP trees to grow during the evolutionary process [**?**] which causes the slowdown of the evolutionary process by making individuals more resistant to change. This section discusses the role of the tree generation methods when learning LQD.

The above mentioned SAP approach is used for learning models with such kind of data, and two different techniques for tree generation are compared: the GROW and the PTC1 methods.

In [**?**] the so-called GROW GP tree-generation algorithm has been widely used since its formulation despite having several weaknesses. Although originally not designed to control the depth and the size of the tree, it can be easily extended to do so. To generate a tree, the algorithm chooses a node type with equal probability. The choice of a node includes the terminal and non-terminal nodes. Once a node has been chosen, and attending to the arity of the node, the algorithm moves to each of its operands and is executed recursively. This process continues either until all the leaf nodes are terminal nodes, or either the depth limit or the tree size limit is reached. In these two latter cases, the tree is filled with terminal nodes.

On the other hand, [**?**] propose an alternative tree-creation algorithm, the Probabilistic Tree Creation 1 (PTC1). This algorithm gives the user control over the expected tree size, a method parameter. Instead of attempting to generate completely uniformly distributed tree structures, PTC1 allows user-defined probabilities of appearance of functions within the tree plus a desired maximum depth D, providing in addition, very low computational complexity. However, PTC1 does not provide any control over the variance in tree size generated, which limits its usefulness. In this algorithm, the set of functions F is divided into two disjoint subsets: terminals nodes set T, each one with probability q_t, and non-terminals nodes N, each one with probability q_t. The recursive method chooses between terminal and non-terminal nodes type for the current node with probability p (see Eq. **??**), where b_n is the arity of non-terminal n. When a terminal node type is chosen, variable or constant is decided according to q_t. In case of non-terminal nodes, the node type will be selected according to q_n. Each of its operators is chosen recursively. Both $\{q_t\}$ and $\{q_n\}$ are given by the user.

$$p = \frac{1 - \frac{1}{E_{tree}}}{\sum_{n \in N} q_n b_n} \tag{1}$$

4 Experimentation and results

In the experimentation stage, the performance of GROW and PTC1 algorithms when learning models with LQD is compared when both regression and time series problems are faced. Several different synthetic LQD data set are generated. In order to obtain such LQD, a two step procedure has been carried out: firstly, the crisp data sets have been generated and then the uncertainty have been introduced to the data. Four different problems are proposed, three of them correspond with regression problems and one is a time series problems. In regression problems the model can not be a function of the output variable, while in the time series problems this variable can be included in the equations. The output variable should be included in an equation and it must be affected with

a delay operator at least. Time series problems models should be evaluated recursively, that is, the output at time t should be calculated using the previously calculated values of the output variable. The regression problems are defined through f_1, f_2 and f_3 in Table **??**, while f_4 represents a time series problem. In all the cases, four input variables are considered ($\{x_0, x_1, x_2, x_3\}$. The variable time t is used to calculate the values of the examples $\{x_i, \forall i\}$. It is included the output variable calculated with the corresponding formulae f_i. The second step for generating the LQD data sets is the aggregation of uncertainty to the data. For each variable in an example, a random value in the range $[-1e^{-4}, 2e^{-4}]$ is added. All the involved variables in a data set are affected, including the output variable, so that all the variables are parameterized as imprecise. The parameters used are presented in Table **??**. Although the number of iterations is set to 1000, data has been gathered each 200 iterations in order to determine the fitness evolution during the simulated annealing run.

Table 1. Formulae for the data sets generation.

$f_1 = x_1 + x_0 * (x_2 - 0.5)$	$t = \{1, \ldots, 100\}$
$f_2 = 2 * x_1 * x_2$	$x_0 = abs(\cos(t))$
$f_3 = \cos(x_0) * (x_2 - x_1)$	$x_1 = abs(\sin(t))$
$f_4 = 2 * x_2 * delay(f_4)$	$x_2 = abs(\cos(t) * \sin(t))$
	$x_3 =$ random in the range $[0, 1]$

Table 2. Parameters used in the experimentation stage. When LQD is assumed, all the variables are set as imprecise variables. In all the cases symmetrical triangular membership functions are used, so only one constant in the GA constants vector is needed per variable. Whenever time series learning is carried out the models are evaluated recursively.

Parameter	Value	Parameter	Value
$\alpha-$cut	0.95	population size	50
Constants range	[-10,10]	GP mutation prob.	0.25
$C^- == C^+$ range	[0, 0.01] %	GP crossover prob.	1
SA Δ value	0.1	GA mutation prob.	0.5
SA initial temperature	1	GP crossover prob.	1
SA final temperature	0	stop generation	{50, 100, 300}
PTC1 Constant prob.	0.3	generations	1000
PTC1 Operators probability	+ 0.145 - 0.145 * 0.145 / 0.145 max 0.08 min 0.08 delay 0.1 sin 0.08 cos 0.08		
N^o of GA constants	7	Maximum size	5
Maximum depth	5		

For both the GROW and the PTC1 methods with each of the four datasets, ten runs have been carried out. Results are shown in Table **??** and Figure **??**. The former shows the values for the means of the MSE of the individuals with the best value for the sum of squared values for the considered fitness in the MOSA: the MSE (mean squared error), number of covered examples and mean output width. On the other hand, in the depicted boxplots, it is shown that PTC1 has a faster convergence than GROW as results are more stable during the succesive iterations. Values for the medians when GROW is used are slightly better than the ones for the PTC1 method, consequently, better results could be obtained if the number of iterations is increased.

Table 3. Mean of the MSE values for the individuals with the best value of the fitness sum in each of the ten runs performed for the four datasets. Results shown should be multiplied by 10^{-3}.

	200		400		600		800		1000	
Dataset	GROW	PTC1	GROW	PTC1	GROW	PTC1	GROW	PTC1	GROW	PTC1
f_1	1.153	1.197	1.208	1.336	1.166	1.391	1.166	1.480	3.613	1.480
f_2	12.254	3.096	11.677	2.751	11.637	2.751	11.557	3.076	0.628	1.611
f_3	0.038	0.190	0.038	0.179	0.042	0.299	0.042	0.299	0.041	0.629
f_4	2.529	2.591	3.076	2.440	2.359	2.206	2.170	2.490	2.138	2.553

5 Conclusions and future work

An statistical study on the results gathered from the run of two tree generation algorithms in SAP learning problems whith LQD has been studied. Data has been gathered each 200 iterations in order to study the fitness evolution. This study reveals that the choice of the suitable tree generation algorithm is relevant to the obtained results. The algorithms used are the GROW and the PTC1. Both behave in a similar way, nevertheless the PTC1 is shown to have a better convergence, although the medians values obtained when GROW is used are slighty better, so it can be assumed that if the number of iterations were increased, the performance would improve. Future work includes the study of the evolution of the diversity that each of the tree generation algorithms induces through the temperature evolution in the MOSA learning process, should be considered.

Acknowledgments. This research has been funded by the Spanish Ministry of Science and Innovation, under project TIN2008-06681-C06-04, the Spanish Ministry of Science and Innovation [PID 560300-2009-11], the Junta de Castilla y Len [CCTT/10/BU/0002] and by the ITCL project CONSOCO.

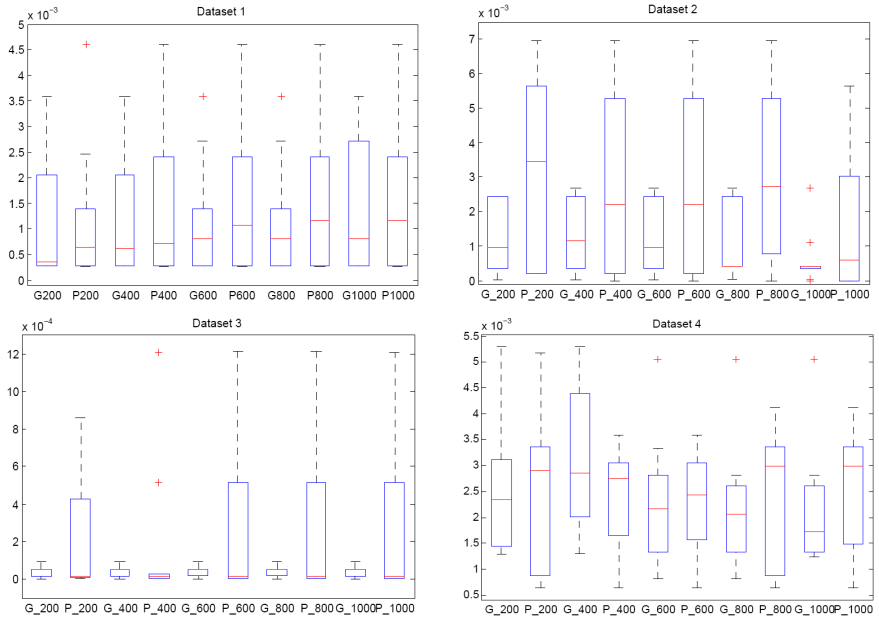

Fig. 1. Boxplots for the values of the individuals with the best value of the fitness sum in each of the ten runs performed for the four datasets.

References

1. J. Alcal-fdez, L. Snchez, S. Garca, M. J. Del Jesus, S. Ventura, J. M. Garrell, J. Otero, J. Bacardit, V. M. Rivas, J. C. Fernndez, and F. Herrera. Keel: A software tool to assess evolutionary algorithms for data mining problems ?
2. Berzosa Berzosa, José R. Villar, and Marco Sedano, Javier García-Tamargo. Tree generation methods comparison in gap problems with low quality data. *accepted for publication in the Proceedings of the International Conference on Soft Computing Models in Industrial and Environmental Applications SOCO'2011, Advances in Intelligent and Soft Compputing Series*, 2011.
3. E. Corchado and A. Herrero. Neural visualization of network traffic data for intrusion detection. *Applied Soft Computing*, 2010.
4. Scott Ferson, Vladik Kreinovich, Janos Hajagos, William Oberkampf, and Lev Ginzburg. Experimental uncertainty estimation and statistics for data having interval uncertainty. Technical report, Technical Report SAND2007-0939; http://www.ramas.com/intstats.pdf, 2007.
5. Andres Folleco, Taghi M. Khoshgoftaar, Jason Van Hulse, and Amri Napolitano. Identifying learners robust to low quality data. *Informatica (Slovenia)*, 33(3):245–259, 2009.
6. Robin De Keyser and Clara Ionescu. Modelling and simulation of a lighting control system. *Simulation Modelling Practice and Theory*, 18(2):165 – 176, 2010.
7. John R. Koza. *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. MIT Press, Cambridge, MA, USA, 1992.

8. Danny H.W. Li, K.L. Cheung, S.L. Wong, and Tony N.T. Lam. An analysis of energy-efficient light fittings and lighting controls. *Applied Energy*, 87(2):558 – 567, 2010.
9. Sean Luke. Two fast tree-creation algorithms for genetic programming. *IEEE Transactions on Evolutionary Computation*, 4(3):274–283, September 2000.
10. Luciano Sánchez, M. Rosario Suárez, J. R. Villar, and Inés Couso. Mutual information-based feature selection and partition design in fuzzy rule-based classifiers from vague data. *Int. J. Approx. Reasoning*, 49:607–622, November 2008.
11. Javier Sedano, Leticia Curiel, Emilio Corchado, Enrique de la Cal, and José R. Villar. A soft computing method for detecting lifetime building thermal insulation failures. *Integr. Comput.-Aided Eng.*, 17:103–115, April 2010.
12. Terence Soule, James A. Foster, and John Dickinson. Code growth in genetic programming. In *Proceedings of the First Annual Conference on Genetic Programming*, GECCO '96, pages 215–223, Cambridge, MA, USA, 1996. MIT Press.
13. José Villar, Enrique de la Cal, Javier Sedano, and Marco García-Tamargo. Analysing the low quality of the data in lighting control systems. In Manuel Graña Romay, Emilio Corchado, and M. García Sebastián, editors, *Hybrid Artificial Intelligence Systems*, volume 6076 of *Lecture Notes in Computer Science*, pages 421–428. Springer Berlin / Heidelberg, 2010.
14. José Villar, Adolfo Otero, José Otero, and Luciano Sánchez. Taximeter verification with gps and soft computing techniques. *Soft Comput.*, 14:405–418, October 2009.
15. José R. Villar, Alba Berzosa, Enrique de la Cal, Javier Sedano, and Marco García-Tamargo. Multi-objecve simulated annealing in genetic algorithm and programming learning with low quality data. *Submitted to Neural Computing*, 2010.
16. José R. Villar, Enrique de la Cal, and Javier Sedano. A fuzzy logic based efficient energy saving approach for domestic heating systems. *Integrated Computer-Aided Engineering*, 16:151–163, April 2009.
17. W.-D. Yu and Liu Y.-C.Sedano. Ahybridization of cbr and numeric soft computing techniques for mining of scarce construction databases. *Autom. in Constr.*, 15:33–46, 2006.