Genetically Evolved Fuzzy Predictor for Photovoltaic Power Output Estimation

Pavel Krömer, Václav Snášel, Jan Platoš, Ajith Abraham
Department of Computer Science
FEECS, VŠB Technical University of Ostrava
Ostrava, Czech Republic
Email: {pavel.kromer,vaclav.snasel,jan.platos}@vsb.cz
ajith.abraham@ieee.org

Lukáš Prokop, Stanislav Mišák
Department of Electric Power Engineering
FEECS, VŠB Technical University of Ostrava
Ostrava, Czech Republic
Email: {lukas.prokop,stanislav.misak}@vsb.cz

***Abstract*—Fuzzy sets and fuzzy logic can be used for efficient data mining, classification, and value prediction. We propose a genetically evolved fuzzy predictor to estimate the output of a Photovoltaic Power Plant. Photovoltaic Power Plants (PVPPs) are classified as power energy sources with unstable supply of electrical energy. It is necessary to back up power energy from PVPPs for stable electric network operation. An optimal value of back up power can be set with reliable prediction models and significantly contribute to the robustness of the electric network and therefore help in the building of intelligent power grids.**

***Keywords*-genetic programming; fuzzy; power output prediction; intelligent power grid**

I. Introduction

A power grid must be operated with balanced energy levels. The electrical energy that is produced by energy sources within the network must be at the same time consumed by customers. The accumulation of reasonable quantities of electrical energy is currently still technically and financially too demanding, even though experimental systems are installed at prototype energy storage facilities, where research is underway to find advanced ways to accumulate electrical energy in large quantities [1].

Thus, in the present time the energetic balance must be still maintained. It is mostly achieved by the regulation of the sources of electrical energy, since the consumption is usually beyond grid operators control. The power grid consists of power plants with stable production of electricity such as coal, gas, and nuclear power plants. On the other hand, it might contain power plants with unstable electrical energy production whose output depends on meteorological conditions at given time and location. An example of unstable energy sources are wind power plants and solar power plants. The amount of the electrical energy produced by such power plants changes with changing weather conditions significantly.

The power grid operator has to maintain a reliable, safe and efficient operation of the energetic network. In order to achieve that, the operator must be able to predict how much electrical energy will be produced by unstable power sources. In a power grid rich in unstable energy sources, a reliable prediction is needed in order to ensure that the stable sources of electrical energy will be able to satisfy the demand for electricity by all customers. Otherwise, the power network might became unstable and unreliable. In electrical networks with a plenty of unstable power sources, it is necessary to keep stable electrical power sources as a backup. To determine the volume of this backup it is necessary to estimate the output of the unstable energy sources such as wind power plants and PVPPs.

Fuzzy classifiers constitute a class of tools and systems that exploit the fuzzy set theory to mine, label, and generally process data. There are simple fuzzy classifiers as well as complex rule-based fuzzy classification systems that usually build and maintain sophisticated rule bases. The popularity of fuzzy classifiers can be attributed among others to their ability to perform soft classification, to assign multiple labels to data samples, and to the ease of their interpretation.

Genetic programming is a nature inspired search and optimization method that was designed specifically to evolve symbolic tree like structures in an automated manner. As such, it is a good tool to evolve symbolic expressions such as the fuzzy predictor.

In this work, we genetically evolve a fuzzy predictor inspired by the area of information retrieval. The predictor was first used for data classification in [2]–[5]. When compared to more complex fuzzy classifier systems, it can be seen as a sole fuzzy rule that maps data features onto a real value from the range $[0, 1]$. The fuzzy classifier is in this study enhanced by the ability to process data as an ordered series of records and it is used to estimate the power output of a PVPP. The usefulness of this approach is illustrated by an experiment with a real world PVPP data.

II. Fuzzy classification systems evolved by evolutionary algorithms

The design of fuzzy classifiers and fuzzy rule-based systems has been successfully aided by the nature inspired methods in the recent years. In this section we summarize few examples of such an evolution or more generally nature inspired fuzzy classifier design. For a comprehensive survey

978-0-7695-4579-0/11 $26.00 © 2011 IEEE
DOI 10.1109/INCoS.2011.97

on the automated evolution of fuzzy classification tools see e.g. [6].

Multi-objective evolutionary algorithms were used for the evolution of linguistic fuzzy rule-based classification systems in the work of Cordón et al. [7]. Another multi-objective evolutionary approach to the evolution of fuzzy rule-based systems was proposed by Ishibuchi and Nojima [8]. They used a hybrid 2-stage approach that combined an initial heuristic stage to select fuzzy rules and evolutionary stage to optimize and tune the system.

Wang et al. [9] used genetic algorithms to integrate fuzzy rule sets and membership functions learned from various information sources. In [10], Freischlad et al. used an evolutionary algorithm to generate fuzzy rules for knowledge representation. Zhou and Khotanzad [11] used genetic algorithm to learn various parameters of fuzzy classification system from a training data set.

The usage of another nature inspired method - the particle swarm optimization - to fuzzy classification system design was studied recently in [12].

III. FUZZY PREDICTOR

The proposed predictor is based on the extended Boolean information retrieval (IR) model. The extended Boolean IR uses the concepts of the fuzzy set theory and fuzzy logic in the area of information retrieval [13], [14]. In the framework of the fuzzy predictor, we use similar data structures, basic concepts, and operations as in the fuzzy IR and we apply them to general data processing (i.e. classification, prediction, and so forth).

The data base used by the fuzzy predictor is a real valued matrix. Each row of the matrix corresponds to a single data record which is interpreted as a fuzzy set of features. Such a general real valued matrix $\boldsymbol{D}$ with m rows (data records) and n columns (data features) can be mapped to an IR index that describes a collection of documents.

The fuzzy predictor has the form of a weighted symbolic expression roughly corresponding to an extended Boolean query in the fuzzy IR analogy. The predictor consists of weighted feature names and weighted aggregation operators. The evaluation of such an expression assigns a real value from the range $[0, 1]$ to each data record. Such a valuation can be interpreted as an ordering or a fuzzy set over the data records.

A. Fuzzy predictor structure

Fuzzy predictor is a symbolic expression that can be parsed into a tree structure. The tree structure consists of nodes and leafs (i.e. terminal nodes). In our fuzzy predictor, we recognize three types of terminal nodes:

- *feature* node - which represents the name of a feature (a search term in the IR analogy). It defines a requirement on a particular feature in the currently processed data record.

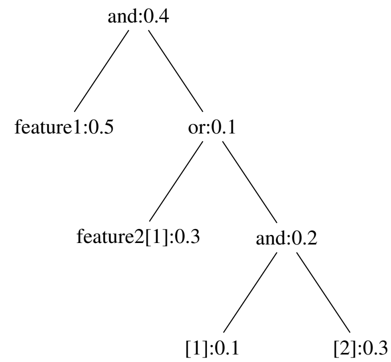

Figure 1: Tree form of a fuzzy predictor

- *past feature* node - which defines a requirement on certain feature in a previous data record. The index of the previous data record (current - 1, current - 2 etc.) is a parameter of the node.
- *past output* node - which puts a requirement on a previous output of the predictor. The index of the previous output (current - 1, current - 2) is a parameter of the node.

The last two node types allow the fuzzy predictor to take into account the order of the data samples, i.e. to see it as a complex time series rather than a simple valuation of unordered records in the data base. Consider the following example of the fuzzy predictor:

> *feature1:0.5 and:0.4 (feature2[1]:0.3 or:0.1 ([1]:0.1 and:0.2 [2]:0.3))*

In our inline syntax, the feature node is defined by feature name and its weight (*feature1:0.5*), past feature node is defined by feature name, index of previous record, and weight (*feature2[1]:0.3*), and past output node is defined by the index of previous output and weight (*[1]:0.5*). The tree that corresponds to the example given above is shown in Fig. 1.

The operator nodes supported currently by the fuzzy predictor are *and*, *or*, and *not* node. Both nodes and leafs are weighted to soften the criteria they represent.

B. Fuzzy predictor evaluation

The fuzzy predictor evaluation procedure is inspired by the fuzzy IR in a similar manner as the data structures it uses. The most important part of the predictor evaluation is the matching of feature values in data records to predictor feature weights. In the IR is the result of such a feature value - feature weight matching called the retrieval status value (RSV) and it evaluates how a document satisfies the criteria represented by a single search criterion defined by the terminal node.

Consider a to be the weight of feature f in the predictor (i.e. the predictor contains a terminal node *f:a*) and $F(d, f)$

to be the value of feature f in data record $d \in \boldsymbol{D}$. The terminal node represents a single criterion. To evaluate this criterion the function $g : [0,1] \times [0,1] \to [0,1]$ will be used. The value of $g(F(d,f),a)$ is the actual RSV for data sample d, feature f, and predictor feature weight a. The key point for RSV evaluation is the interpretation of the predictor feature weight a. The most commonly used IR interpretations understand the predictor feature weight as an importance weight, a threshold, or an ideal feature value description [13], [14].

We are using the threshold interpretation of a and the equation for RSV evaluation in this case is shown in (1) [13], [14]. In (1), $P(a)$ and $Q(a)$ are coefficients used for tuning the threshold curve. An example of $P(a)$ and $Q(a)$ could be e.g. $P(a) = \frac{1+a}{2}$ and $Q(a) = \frac{1-a^2}{4}$. For the threshold interpretation, a node containing feature f with the weight a is a request satisfied by data samples having $F(d,f)$ equal or greater to a. The data samples satisfying this condition will be awarded by high RSV and contrariwise data records having $F(d,f)$ smaller than a will be awarded by small RSV.

$$g(F(d,f),a) = \begin{cases} P(a)\frac{F(d,f)}{a} & \text{for} \quad F(d,f) < a \\ P(a) + Q(a)\frac{F(d,f)-a}{1-a} & \text{for} \quad F(d,f) \geq a \end{cases} \tag{1}$$

The operators *and*, *or*, and *not* can be evaluated using fuzzy set operations. Fuzzy set operations are extensions of crisp set operations on fuzzy sets [15]. They are defined using the characteristic functions of operated fuzzy sets [16]. In [15] L. Zadeh defined basic methods for the complement, union, and intersection of fuzzy sets but besides these standard fuzzy set operations, whole classes of prescriptions for the complements, intersections, and unions on fuzzy sets were defined [17].

In this study, we use the standard t-norm (2) and t-conorm (3) for the implementation of *and* and *or* operators and fuzzy complement for the evaluation of the *not* operator (4).

$$t(x,y) = \min(x,y) \tag{2}$$

$$s(x,y) = \max(x,y) \tag{3}$$

$$c(x) = 1 - x \tag{4}$$

However, the use of other common t-norm and t-conorm pairs is of course possible.

C. Summary

The fuzzy predictor presented in this work is a simple version of a fuzzy classifier. In contrast to more complex fuzzy rule-based systems that usually constitute traditional fuzzy classifiers, it consists of a single expression that states soft requirements on data records in terms of data features. Moreover, conditions can be put on past feature values and past output values and therefore allow the predictor to see the data base as an ordered sequence of records similar to a time series.

IV. Genetic programming

The evolution of the fuzzy predictor for the PVPP power output estimation utilizes genetic programming. In this section, we provide brief introduction into the area of genetic algorithms and genetic programming.

Genetic algorithms are a popular member of the wide chapter of evolutionary algorithms. They are based on the programmatic implementation of genetic evolution and they emphasize selection and crossover as the most important operations in the evolutionary optimization process [18], [19].

Genetic programming (GP) is an extension to the popular nature inspired stochastical optimizer, the genetic algorithms [18], [19]. Genetic algorithms perform an artificial (software) evolution of a population of chromosomes representing potential solutions to an investigated problem encoded into a suitable data structures, most often fixed length strings of low cardinality alphabets (e.g. bit strings). The evolution is performed by iterative application of genetic operators modifying the chromosomes, i.e. the encoded forms of problem solutions, in order to emulate the principles of Darwinian evolution, the survival of fittest, and Mendelian inheritance.

The GP extends the genetic algorithms by enabling work with hierarchical, often tree-like, chromosomes with an uneven and unlimited length [18], [20]. The GP was introduced as a tool to evolve whole computer programs and represented a step towards adaptable computers that could solve problems without being programmed explicitly [21]. This is an important ability because solutions to most problems can be formulated by the means of computer programs. Moreover, the GP can be used to develop solutions in the field of machine learning, symbolic processing, or any other domain that can formulate its solutions by means of parseable symbolic expression. GP allows the efficient evolution of such symbolic expressions with well-defined syntax and grammar. GP chromosomes take the form of hierarchical variably-sized expressions, point-labeled structure trees. The trees are constructed from nodes of two types, terminals and functions.

The chromosomes are evaluated by the execution of instructions corresponding to tree nodes [21]. Terminal nodes are evaluated directly (e.g. by reading an input variable) and functions are evaluated after left-to-right depth-first evaluation of their parameters.

Genetic operators are applied to the nodes in the tree-shaped chromosomes. A crossover operator is implemented as the mutual exchange of randomly selected sub-trees of the parent chromosomes. Mutation has to modify the chromosomes by pseudo-random arbitrary changes in order to prevent premature convergence and broaden the coverage of the fitness landscape. Mutation could be implemented as:

i) removal of a sub-tree at a randomly chosen node

ii) replacement of a randomly chosen node by a newly generated sub-tree
iii) replacement of node instruction by a compatible node instruction (i.e. a terminal can be replaced by another terminal, a function can be replaced by another function of the same arity)
iv) a combination of the above

The GP facilitates an efficient evolution of symbolic expressions, even whole computer programs. In this work, we use the GP for automated fuzzy predictor learning.

A. GP for fuzzy predictor evolution

To use the GP for fuzzy predictor learning, we need to define the encoding, genetic operators, and the fitness function. The encoding is straightforward because the fuzzy predictor is in fact a tree (see Fig. 1). We create a random population of such trees (candidate predictors) and apply the GP to evolve the population. The generation of random predictors is done with respect to the probabilities summarized in .

The implementation of the crossover operator is also simple: for each two trees, we swap randomly selected branches. Such an operation results in valid fuzzy predictors. The mutation operator is more complex because it has to reflect the domain of the problem and properties of each node that should be mutated. The mutation types that were implemented and their respective probabilities are shown in table Ib.

Table I: Random query generation an mutation probabilities.

(a) Probabilities of generating random nodes.

Event	Probability
Generate feature node	0.17
Generate past feature node	0.17
Generate past output node	0.17
Generate op. *and*	0.24
Generate op. *or*	0.24
Generate op. *not*	0.02

(b) Probabilities of mutation operations.

Event	Probability
Mutate node weight	0.5
Insert or delete *not* node	0.1
Replace with another node or delete *not* node	0.32
Replace with random branch	0.08

The goal of the fuzzy predictor evolution is to find such a predictor that would describe the same fuzzy set of data records as indicated in the training data base.

The similarity of two fuzzy sets can be defined as:

$$\rho(X|Y) = \begin{cases} \frac{\|X \cap Y\|}{\|Y\|} & \|Y\| \neq 0 \\ 1 & \|Y\| = 0 \end{cases} \tag{5}$$

where $\|A\|$ is the $\Sigma-$count, i.e. the sum of the values of characteristic function for all members of the fuzzy set A [22]:

$$\|A\| = \sum_{x \in A} \mu_A(x) \tag{6}$$

Precision P and recall R are two measures that can be used to evaluate the effectiveness of an IR system and we use them to determine the suitability of a candidate fuzzy predictor. In the IR, precision corresponds to the probability of retrieved document to be relevant and recall can be seen as the probability of retrieving relevant document. We use precision P and recall R to evaluate the similarity of two fuzzy sets:

$$P = \rho(t(\boldsymbol{D})|p(\boldsymbol{D})) \qquad R = \rho(p(\boldsymbol{D})|t(\boldsymbol{D})) \tag{7}$$

where $t(\boldsymbol{D})$ stands for the target fuzzy set and $p(\boldsymbol{D})$ for the fuzzy set generated by the predictor f. For an easier evaluation, measures combining precision and recall into one scalar value were developed. The F-score F is among the most used scalar combinations of P and R:

$$F = \frac{(1+\beta^2)PR}{\beta^2 P + R} \tag{8}$$

We use the F-score F as a fitness function when evolving the fuzzy predictor. A good fuzzy predictor that generates fuzzy set of data records that is similar to the training fuzzy set of records will yield high precision, recall, and F-score.

V. Experiment

We have used the fuzzy predictor for a PVPP output estimation. We have recorded the power output of one czech PVPP during the period of two weeks. Next, we have gathered hourly estimates of solar radiation intensity in the same location for the same period of time.

The input data for the prediction model were obtained from the Czech Hydrometeorological Institute, which operates a weather prediction model ALADIN for the Czech Republic. ALADIN provides (among others) estimated values of the intensity of solar radiation I_R (Wm^{-2}) with a time step one hour and it is able to predict these values for up to 72 hours in the future. We have matched these predictions to the output power of the PVPP. Both input and output data was for the purpose of the prediction algorithm normalized into the interval $[0, 1]$.

The data was divided into two halves. The first half was used for training, i.e. for the evolution of a fuzzy predictor that would estimate the output of the PVPP based on the predicted intensity of solar radiation in the area.

The second half was used to verify the fuzzy predictor. We have used the evolved predictor to estimate the PVPP output from the solar radiation intensity estimates obtained from the ALADIN model and compared it to the real output of the PVPP. The real and estimated output of the PVPP

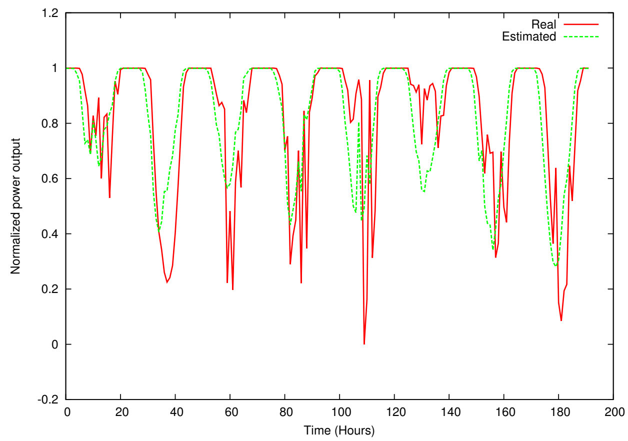

Figure 2: Estimated and real output of the PVPP

during the experiment is shown in Fig. 2. We can clearly see that the predictor has learned the main trends in power production and that the estimate was able to copy the real power output of the facility quite well.

VI. Conclusions

In this paper, we have designed a fuzzy predictor to estimate the power output of a PVPP. A real world experiment was conducted and a predictor generated from a data set containing one week of estimates of solar radiation intensity and real power output of a PVPP was able to estimate the output of the same PVPP from solar intensity predictions in the following week. Accurate predictions of the power output of PVPPs can be seen as a building block of intelligent power grids. It shows that soft computing and nature inspired algorithms can contribute to the creation of smart electrical networks.

The experiment presented in this paper is indeed initial. The period of one week is rather small for both, training an evaluation of the predictor. Moreover, the training and testing data base could be created in a more elaborate way. For instance, the solar radiation intensity prediction exists for every hour while the PVPP power output measurement is available for every minute. A more precise data base could yield more precise predictions. In our future work we want to improve the experimental data set, perform the experiments in a larger scale, compare our approach with other classification and prediction techniques and last but not least tune all the parameters of the algorithm used to create the fuzzy predictor.

Acknowledgement

This paper has been elaborated in the framework of the IT4Innovations Centre of Excellence project, reg. no. CZ.1.05/1.1.00/02.0070 supported by Operational Programme 'Research and Development for Innovations' funded by Structural Funds of the European Union and state budget of the Czech Republic.

References

[1] P. Kačor, S. Mišák, and L. Prokop, "Optimization and redesign of vertical axis wind turbine for generator of independent source of energy," in *Annals of DAAAM for 2010 & Proceedings of the 21st International DAAAM Symposium* (B. Katalinic, ed.), pp. 1053–1054, DAAAM International Vienna, Vienna, October 2010.

[2] P. Krömer, V. Snášel, J. Platoš, and A. Abraham, "Evolving fuzzy classifier for data mining - an information retrieval approach," in *Computational Intelligence in Security for Information Systems 2010* (. Herrero, E. Corchado, C. Redondo, and . Alonso, eds.), vol. 85 of *Advances in Intelligent and Soft Computing*, pp. 25–32, Springer Berlin / Heidelberg, 2010. 10.1007/978-3-642-16626-6-3.

[3] V. Snásel, P. Krömer, J. Platos, and A. Abraham, "The evolution of fuzzy classifier for data mining with applications," in *SEAL* (K. Deb, A. Bhattacharya, N. Chakraborti, P. Chakroborty, S. Das, J. Dutta, S. K. Gupta, A. Jain, V. Aggarwal, J. Branke, S. J. Louis, and K. C. Tan, eds.), vol. 6457 of *Lecture Notes in Computer Science*, pp. 349–358, Springer, 2010.

[4] P. Kromer, J. Platos, V. Snasel, and A. Abraham, "Towards intrusion detection by information retrieval and genetic programming," in *2010 Sixth International Conference on Information Assurance and Security (IAS 2010)*, (Atlanta, Georgia, USA), pp. 148–153, 8 2010. IEEE Catalog numberCFP1061C-CDR, ISBN978-1-4244-7408-0.

[5] P. Krömer, V. Snásel, and J. Platoš, "Learning patterns from data by an evolutionary-fuzzy approach," in *Soft Computing Models in Industrial and Environmental Applications, 6th International Conference SOCO 2011* (E. Corchado, V. Snásel, J. Sedano, A. Hassanien, J. Calvo, and D. Slezák, eds.), vol. 87 of *Advances in Intelligent and Soft Computing*, pp. 127–135, Springer Berlin / Heidelberg, 2011. 10.1007/978-3-642-19644-7-14.

[6] O. Cordón, "A historical review of evolutionary learning methods for mamdani-type fuzzy rule-based systems: Designing interpretable genetic fuzzy systems," *International Journal of Approximate Reasoning*, vol. 52, no. 6, pp. 894 – 913, 2011.

[7] O. Cordón, M. J. del Jesus, and F. Herrera, "Genetic learning of fuzzy rule-based classification systems cooperating with fuzzy reasoning methods," *International Journal of Intelligent Systems*, vol. 13, pp. 1025–1053, 1998.

[8] H. Ishibuchi and Y. Nojima, "Multiobjective formulations of fuzzy rule-based classification system design.," in *EUSFLAT Conf.* (E. Montseny and P. Sobrevilla, eds.), pp. 285–290, Universidad Polytecnica de Catalunya, 2005.

[9] C.-H. Wang, T.-P. Hong, and S.-S. Tseng, "Integrating membership functions and fuzzy rule sets from multiple knowledge sources," *Fuzzy Sets Syst.*, vol. 112, pp. 141–154, May 2000.

[10] M. Freischlad, M. Schnellenbach-Held, and T. Pullmann, "Evolutionary generation of implicative fuzzy rules for design knowledge representation," in *EG-ICE* (I. F. C. Smith, ed.), vol. 4200 of *Lecture Notes in Computer Science*, pp. 222–229, Springer, 2006.

[11] E. Zhou and A. Khotanzad, "Fuzzy classifier design using genetic algorithms," *Pattern Recogn.*, vol. 40, pp. 3401–3414, December 2007.

[12] R. Prado, S. Garcia-Gala andn, J. Exposito, and A. Yuste, "Knowledge acquisition in fuzzy-rule-based systems with particle-swarm optimization," *Fuzzy Systems, IEEE Transactions on*, vol. 18, pp. 1083 –1097, dec. 2010.

[13] F. Crestani and G. Pasi, "Soft information retrieval: Applications of fuzzy set theory and neural networks," in *Neuro-Fuzzy Techniques for Intelligent Information Systems* (N. Kasabov and R. Kozma, eds.), pp. 287–315, Heidelberg, DE: Springer Verlag, 1999.

[14] D. H. Kraft, F. E. Petry, B. P. Buckles, and T. Sadasivan, "Genetic Algorithms for Query Optimization in Information Retrieval: Relevance Feedback," in *Genetic Algorithms and Fuzzy Logic Systems* (E. Sanchez, T. Shibata, and L. Zadeh, eds.), (Singapore), World Scientific, 1997.

[15] L. A. Zadeh, "Fuzzy sets," *Information and Control*, vol. 8, pp. pp. 338–353, 1965.

[16] J. Jantzen, "*Tutorial On Fuzzy Logic*," Technical Report 98-E-868 (logic), Technical University of Denmark, Dept. of Automation, 1998.

[17] T. Feuring, *Fuzzy-systeme*. Institut fr Informatik, Westflische Wilhelms Universitt, Mnster, 1996.

[18] J. Koza, "*Genetic programming: A paradigm for genetically breeding populations of computer programs to solve problems*," Technical Report STAN-CS-90-1314, Dept. of Computer Science, Stanford University, 1990.

[19] M. Mitchell, *An Introduction to Genetic Algorithms*. Cambridge, MA: MIT Press, 1996.

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

[21] M. Affenzeller, S. Winkler, S. Wagner, and A. Beham, *Genetic Algorithms and Genetic Programming: Modern Concepts and Practical Applications*. Chapman & Hall/CRC, 2009.

[22] L. A. Zadeh, *Empirical Semantics*, vol. 1 of *Quantitative Semantics, Vol. 12*, ch. Test-score semantics dor natural languages and meaning representation via Pruf, pp. 281–349. Bochum: Studienverlag Brockmeyer, 1981.