AN AI INTEGRATED METHOD FOR PERMEABILITY PREDICTION IN PETROLEUM RESERVOIR

Yuantu Huang
School of Computer Science and Engineering
The University of New South Wales, Sydney NSW 2052, Australia
Phone: +61-2-9385 5660
Fax: +61-2-9385 3965
Email: yuantu.huang@unsw.edu.au

Patrick M. Wong (Corresponding author)
School of Petroleum Engineering
The University of New South Wales, Sydney NSW 2052, Australia
Phone: +61-2-9385 5189
Fax: +61-2-9385 3965
Email: pm.wong@unsw.edu.au

Tom D. Gedeon
School of Computer Science and Engineering
The University of New South Wales, Sydney NSW 2052, Australia
Phone: +61-2-9385 3965
Fax: +61-2-9385 5995
Email: tom@cse.unsw.edu.au

Preferred technical categories: **Pattern Recognition** or **Industrial Applications**

AN AI INTEGRATED METHOD FOR PERMEABILITY PREDICTION IN PETROLEUM RESERVOIR

Yuantu Huang
School of Computer Science and Engineering
The University of New South Wales, Sydney NSW 2052, Australia

Patrick M. Wong
School of Petroleum Engineering
The University of New South Wales, Sydney NSW 2052, Australia

Tom D. Gedeon
School of Computer Science and Engineering
The University of New South Wales, Sydney NSW 2052, Australia

Abstract

This paper introduces and demonstrates an AI integrated method ("AIM") for predicting reservoir permeability of sedimentary rocks in drilled wells in the petroleum exploration and development industry. The method employs Takagi-Sugeno's fuzzy reasoning, and its fuzzy rules and membership functions are automatically derived by neural networks and floating-point encoding genetic algorithms. The method is trained with known data and tested with unseen data. The results show that AIM has a good generalisation capability and is an effective approach for large scale industrial application.

AN AI INTEGRATED METHOD FOR PERMEABILITY PREDICTION IN PETROLEUM RESERVOIR

Y. Huang[1, 2], P.M. Wong[2] and T.D. Gedeon[1]
[1]School of Computer Science and Engineering
[2]School of Petroleum Engineering
The University of New South Wales, Sydney, NSW 2052, Australia

ABSTRACT

This paper introduces and demonstrates an AI integrated method ("AIM") for predicting reservoir permeability of sedimentary rocks in drilled wells in the petroleum exploration and development industry. The method employs Takagi-Sugeno's fuzzy reasoning, and its fuzzy rules and membership functions are automatically derived by neural networks and floating-point encoding genetic algorithms. The method is trained with known data and tested with unseen data. The results show that AIM has a good generalisation capability and is an effective approach for large scale industrial application.

KEYWORDS: neural networks, genetic algorithms, fuzzy reasoning, petroleum reservoir, well logs.

1. INTRODUCTION

1.1 Petroleum reservoir

A petroleum reservoir is a volume of porous sedimentary rock which has been filled with hydrocarbons, such as oil and gas. Reservoir properties, such as permeability (a measure of fluid conductance in porous media), are a set of parameters which are usually used to characterise the spatially varied geologic information and are important for reserves estimation and production forecasting. These properties are commonly obtained from drilled wells which are limited and sparse. Electronic equipment is used to "log" the well in such a way that multi-type digital measurements or "well logs" are obtained as a function of reservoir depth.

Reservoir permeability can be estimated by correlating well logs with the laboratory measured permeability data obtained from rock samples or "cores." Retrieving cores for laboratory testing is expensive and is only practiced at selected depths/intervals. Therefore, permeability prediction at the "un-cored" wells relies on functional transformation developed at the "cored" wells.

The problem is traditionally solved by using empirical formula which is labour-intensive, or multiple regression which is straightforward but oversimplify the natural complexity. The advent of artificial intelligence (AI) techniques, such as neural networks, fuzzy logic and genetic algorithms with modern computers, offers powerful tools for further improving permeability predictions.

1.2 Problem statements

The above problem can be viewed as a regression problem. Data (well logs and permeability) from cored wells are treated as training data which can be used to develop correlation equations, usually one for each cored well. The permeability required at the un-cored wells can be obtained by using the same well logs as inputs to the correlation equations developed. An appropriate weighting scheme can be used to weight the importance of each of the correlation equations.

Figure 1 illustrates the problem by using three wells, namely W1, W2 and W3, drilled in the same petroleum reservoir. W1 and W2 are cored and W3 is un-cored. We may write the input-output correlation as follows:

$$\vec{Y}_i = f\left(\vec{X}_i, \vec{\theta}_i\right) \quad i = 1,2 \tag{1}$$

$$\vec{Y}_3 = \frac{\sum_{i=1}^{2} \mu_i^{\beta_i} \cdot f\left(\vec{X}_3, \vec{\theta}_i\right)}{\sum_{i=1}^{2} \mu_i^{\beta_i}} \tag{2}$$

where $\vec{X}_i$ denotes well logs at well i as an input vector, $\vec{\theta}_i$ denotes a set of functional parameters, $\vec{Y}_i$ denotes permeability as an output vector, $\vec{\mu} = (\mu_1, \mu_2)$ is a weighting vector, and $\vec{\beta} = (\beta_1, \beta_2)$ is a real number in $[0, \infty)$. $f(.)$ is correlation equation for transforming an input vector to the output vector. The problem is to obtain $f(.), \vec{\beta}, \vec{\theta}$, and hence $\vec{Y}_3$ (the permeability at W3), given that $\vec{X}_i$ $(i = 1, 2, 3)$ and $\vec{Y}_i$ $(i = 1, 2)$ are known.

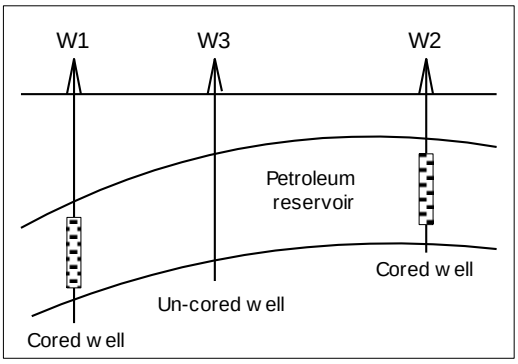

Fig. 1: Permeability prediction problem.

1.3 Previous works

There are many solutions to the above problem, ranging from multiple regression [1,2], neural networks [3-4], and neural-fuzzy techniques [5-6]. Both multiple regression and neural networks firstly generate $f(.)$ in Equation (1) from W1 and/or W2, then use $\vec{X}_3$ in Equation (1) and obtain $\vec{Y}_3$. On the other hand, the neural-fuzzy techniques [6] employ the Takagi-Sugeno's fuzzy reasoning [7] (see Equation (2)) and use the separation distances between W1 and W3, and W2 and W3 as the fuzzy membership function values $\vec{\mu} = (\mu_1, \mu_2)$. When we set $\vec{\beta} = (-1, -1)$, Equation (2) becomes an inverse distance estimator which assigns a higher weighting to the estimate from the cored well closer to the un-cored well. The previous approach, however, ignores the internal similarity of the data sets among the drilled wells which may be ineffective when applying to complex reservoirs.

We have used a neural-driven fuzzy reasoning method [8] to optimise the membership function values by genetic algorithms (GAs) with dramatically improved results [9]. In [10], we applied binary-encoding GAs [11] to optimise the connecting weights in neural networks. Our results were improved compared to weight training by backpropagation algorithm (BP), but were computationally more expensive.

In order to improve the speed and accuracy, one solution is to use the connection weights trained by BP to initialise the chromosomes in GAs. This requires floating-point encoding GAs which are fast and accurate [12], but are not yet popular for industrial application to date.

1.4 Objective

The objective of this paper is to combine neural networks optimised by floating-point encoding GAs with Takagi-Sugeno's fuzzy reasoning to develop a fully AI integrated method ("AIM") for permeability prediction.

Section 2 presents the AIM methodology with a detailed description of the floating-point encoding GAs. Section 3 applies AIM to predict permeability in an oil and gas well located in North West Shelf, offshore western Australia.

Data from three cored wells are available for this study. The first two data sets are used for training, while the third set is treated as unseen data and is used to test the performance of AIM. This is similar to the situation illustrated in Figure 1.

2. "AIM" METHODOLOGY

2.1 Neural networks

A standard three layer (input, hidden, and output) neural network is used to generate $f(.)$ in Equation (1). The number of input neurons is determined by the number of well logs available and one output neuron is used to represent permeability. The number of hidden neurons is obtained by trial and error.

In order to avoid over-fitting, we apply early-stopping by using a validation set, i.e., to terminate training when the minimum error on the validation set is reached. We use the data from one cored well as the training set and to develop $f(.)$. The data from the other core well is used as the validation set. Swapping the use of the data sets give two generalised correlation equations $f(.)$. The total errors (training plus the validation errors) are used as $\vec{\beta} = (\beta_1, \beta_2)$ in Equation (2). Note that the higher the β value, the smaller the weighting μ^{β} as $\mu = (0,1)$.

2.2 Fuzzy reasoning

To generate $\vec{\mu}$, a similar neural network is used. The inputs are identical, but the number of output neurons is two. The following target data are used:

$$s_i = \begin{cases} 1 & \vec{X}_i \in Wi \\ 0 & otherwise \end{cases} \quad i = 1,2 \qquad (3)$$

This training scheme estimates the degree of membership of each set of well logs "belonging" to each of the cored wells. The value of the membership function is defined as the output of the trained neural network, i.e., $\mu_i = \hat{s}_i$, where $\hat{s}_i$ denotes the output from the network.

The weighting scheme presented here is more superior to the one proposed in [6] which used only the inverses distances and total errors.

2.3 Genetic algorithms

Genetic algorithms (GAs) mimic processes observed in nature evolution, and are stochastic global search methods. Individuals in a population are called chromosomes (strings). A genetic representation for a potential solution to a problem is encoded as a chromosome. A good initial population of potential solutions can result in fast convergence with higher accuracy for real world problems. The steps of a typical genetic algorithm are listed as follows:

a. Initialise a population of chromosomes within the range of potential solutions.
b. Evaluate each chromosome in the population based on the evaluation (fitness) function.
c. Select chromosomes in the population (according to the fitness values) as parent chromosomes to reproduce.
d. Apply genetic operators to the parent chromosomes to produce children so as to generate a new population.
e. Evaluate the chromosomes in the new population.
f. Stop and return the best chromosomes as the final solution if a termination condition is satisfied; otherwise, go to step c.

Generally speaking, the parent chromosome selection for both binary and floating-point encoding is similar. However, the genetic operators used for floating-point encoding are different from these for binary encoding. In our problem we will use the connection weights trained by BP as one of the chromosomes in the initial population, and hence, floating-point encoding is required. The following steps outline the general structure of floating-point encoding genetic algorithms:

2.2.1 Population initialisation

Let $\vec{\theta} = (w_1, w_2, ..., w_n)$ denotes a parameter vector of connection weights trained by BP, where n is the number of total connecting weights in the network. We define a real number parameter "swing" as Δw. The use of this parameter avoids

the need to define the upper and lower bounds for the parameters to be optimised. Typically the value of Δw for our problem is $[\Delta w_{\min} = 0.5,\ \Delta w_{\max} = 10]$. Assuming the size of a population is m and we keep $\vec{\theta}$ as one of the chromosomes in the initialised population, the remaining $m-1$ chromosomes are randomly generated, and the range of the i^{th} parameter in a chromosome is $[w_i - \Delta w, w_i + \Delta w]$ $(i = 1, 2, ..., n)$.

2.2.2 Performance evaluation

A "fitness" function is used to evaluate the performance of each of the chromosomes. A typical fitness function is defined as follow:

$$F(\vec{\theta}_j) = \frac{10}{1 + E(\vec{\theta}_j)} \tag{4}$$

and,

$$E(\vec{\theta}_j) = \sum_{k=1}^{N} \left(y_k - \hat{y}_{kj}\right)^2 \tag{5}$$

where $F(\vec{\theta}_j)$ with $\vec{\theta}_j = (w_{j1}, w_{j2}, ..., w_{jn})$ is the fitness value for the j^{th} $(j = 1, 2, ..., m)$ chromosome, $E(\vec{\theta}_j)$ is the sum of squared errors of the sample (observed) data y_k $(= 1, 2, ..., N)$ and the model predictions $\hat{y}_{kj}$ defined in Equations (1) or (2) and N is the total number of sample data. The higher the $F(\vec{\theta}_j)$ value, or the lower the $E(\vec{\theta}_j)$, the better the solution.

2.2.3 Reproduction

Selecting parents for reproduction is a very important aspect of FGA. There are many methods of selection. A parent solution can be selected more than once. The most popular selection method is Goldberg's roulette wheel parent selection [13]. The roulette wheel has slots sized according to the fitness of each chromosome. The purpose is to give more reproductive chances to those population members who are the most fit. After roulette wheel parent selection, we copy the best chromosome twice to replace two of the worst.

In order to accelerate convergence, before crossover and mutation we have to dynamically update the swing and the range of the parameters for the chromosomes as follow:

$$\Delta w = \Delta w_{\min} + \frac{(M - k) \cdot (\Delta w_{\max} - \Delta w_{\min})}{M} \tag{6}$$

where M is the desired number of iterations and $k\ (\le M)$ is the current iteration counter. Let $\vec{\theta} = (w_1, w_2, ..., w_n)$ be the best chromosome, then the new range of the i^{th} parameter in a chromosome for a new population is $[w_i - \Delta w, w_i + \Delta w]$ $(i = 1, 2, ..., n)$.

2.2.4 Crossover

Crossover produces offspring by exchanging genetic information between the selected parent solutions. The selection criteria are based on a user-defined probability for crossover, P_c (generally between 0.5 to 0.9). This probability defines the number of candidates for crossover. For example, if P_c is 0.7, it means about 70% of the parent chromosomes in the population will randomly be selected and mated in pairs. We use two point arithmetical crossover.

Let the i^{th} chromosome $(w_{i1}, w_{i2}, ..., w_{in})$ and the j^{th} chromosome $(w_{j1}, w_{j2}, ..., w_{jn})$ be selected for crossover between the p^{th} and the q^{th} parameters $(1 \le p \le q \le n)$. The offspring become:

$$(w_{i1}, ..., w_{ip-1}, w^j_{ip}, ..., w^j_{iq}, w_{iq+1}, ..., w_{in})$$

and $(w_{j1}, ..., w_{jp-1}, w^i_{jp}, ..., w^i_{jq}, w_{jq+1}, ..., w_{in})$

where $w^j_{jk} = \alpha \cdot w_{jk} + (1 - \alpha) \cdot w_{ik}$

and $w^j_{ik} = \alpha \cdot w_{ik} + (1 - \alpha) \cdot w_{jk}$ for $p \le k \le q$

and α is a uniform random number in $[0,1]$.

2.2.5 Mutation

The reproduction and crossover operation would only exploit the known regions in the solution space, which could lead to premature convergence for the fitness function with the consequence of missing the global optimum by

exploiting some local optimum. Mutation is a genetic process to avoid such a problem. This process allows the introduction of new characteristics to the offspring, which are unrelated to the parent solutions. It first requires a user-defined probability for mutation P_m (generally between 0.01 to 0.2). We use non-uniform arithmetical mutation. Let the j^{th} parameter in the i^{th} chromosome w_{ij} is selected for mutation. Thus, the new parameter is $w_{ij}^{*} = \alpha \cdot w_{ij} + (1-\alpha) \cdot v$, where v is a uniform random number in $[w_i - \Delta w, w_i + \Delta w]$. More details about genetic operators for floating-point encoding can be found in [14].

2.3 Neural-driven fuzzy reasoning

After generating the functions $f(.)$ with optimised $\vec{\theta}$ as shown in Equation (1), we can extract two rules from the two cored wells, W1 and W2:

Rule 1: If $\vec{X}_3 \in$ W1 then $\vec{Y}_1 = f(\vec{X}_3, \vec{\theta}_1)$

Rule 2: If $\vec{X}_3 \in$ W2 then $\vec{Y}_2 = f(\vec{X}_3, \vec{\theta}_2)$

Similar optimisation routines can be run to obtain the membership function values in Equation (3). Equation (2) can then be applied to obtain the final estimate. The procedure is considered the neural-driven fuzzy reasoning.

3. FIELD EXAMPLE

In this case study, data from three wells, W1, W2 and W3, located in North West Shelf, offshore western Australia, were used. The well logs available for the analyses were: gamma ray (GR), deep resistivity (LLD), sonic travel time (DT), bulk density (RHOB) and neutron porosity (NPHI). The classification of the rock was also incorporated in the input data set as a discrete variable. Permeability measurements were available at selected well depths. The number of data pairs in each well was 152, 156, and 140 points, respectively. There are a total of six inputs and one output. All the input data were normalised in the range of $[0,1]$. All the permeability values were normalized in the range of $[0.1,0.9]$. The objective of this example is to predict the permeability values at W3 using the transformations developed by W1 and W2 data.

In this study, a three layer neural network with five hidden neurons was found to be the best structure. Due to the presence of the bias weights (only in the input layer), a total of 40 connection weights was used to generate Equation (1). For Equation (2), the use of two output neurons resulted in 45 connection weights. The GA configuration is shown in Table 1.

Iteration times	5000
Population size, m	20
Probability for crossover, P_c	0.8
Probability for mutation, P_m	0.1

Table 1: GA configuration.

Table 2 shows the training and validation errors from the neural networks in Equation (1), optimised by the floating-point encoding Gas using BP trained weights as the initial population. The results using only BP are also tabulated in the same table. From these results, the errors from BP followed by GA were smaller and hence the rules were more reliable for prediction. The corresponding $\vec{\beta} = (\beta_1, \beta_2)$ was also calculated. Note that two β values were similar which means that there is not substantial bias when swapping the data sets for training and validation.

	Training error	Validation error	β value
W1 for training W2 for validation	0.065 (0.072)	0.075 (0.084)	0.140 (0.156)
W2 for training W1 for validation	0.073 (0.075)	0.076 (0.086)	0.149 (0.161)

Table 2:Root mean square errors from BP followed by GA. The bracketed values are results from BP alone.

Figures 2 and 3 show the trained hyper-surface membership function values $\bar{\mu} = (\mu_1, \mu_2)$ obtained by GAs. These values were calculated by using input data from W3. Hence the plots show the degree of membership of W3 data belonging to W1 and W2 respectively. We can see that majority of the W3 patterns are similar to the W2 data.

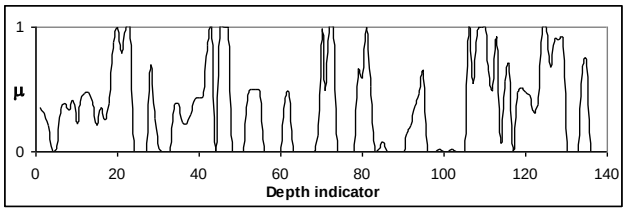

Fig. 2: Membership values showing W3 patterns "belonging" to W1.

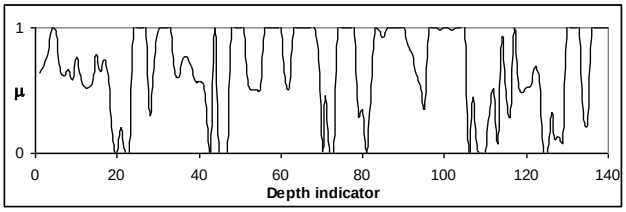

Fig. 3: Membership values showing W3 patterns "belonging" to W2.

4. RESULTS

The results from AIM are tabulated in Table 3. The results from using separate rules trained by BP alone are also shown in the same table. Clearly, the AIM gave the smallest error. When comparing predictions from separate rules, the performance from AIM was 36% and 26% better than rule 1 and rule 2 respectively.

	TSS
AIM	0.939
Rule 1 from BP alone	1.273
Rule 2 from BP alone	1.180

Table 3. Comparison of the total sum of error squares (TSS) to 140 data points at W3 using AIM and separate rules developed by BP alone.

Figure 4 shows the predictions at W3 along with the actual permeability data. The AIM predictions matched well with the actual values. Figure 5 also shows the scatter-plots of the 140 predictions versus the actual values with the correlation coefficient $r^2 = 0.73$. It was the highest compared to others with $r^2 = 0.66$ using rule 1 and $r^2 = 0.63$ using rule 2.

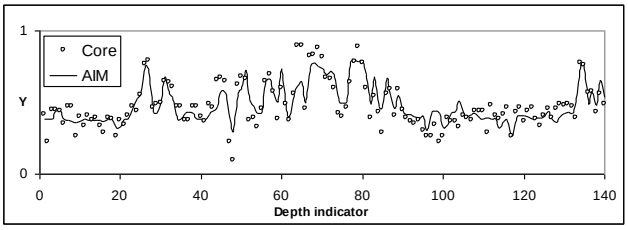

Fig. 4: Permeability profiles at W3.

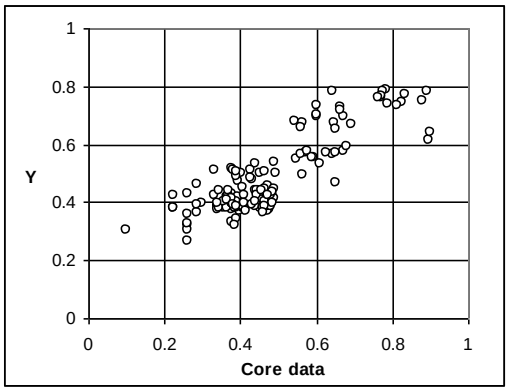

Fig. 5: Scatter-plots of the AIM predictions versus the core data at W3.

5. DISCUSSION

The performance of the AIM was good compared to the conventional approach. It not only has the intrinsic advantages of the neural-driven fuzzy reasoning, but also incorporated the floating-point encoding GAs to further optimise the neural networks trained by BP. Because the chromosomes of the initial population in GAs come from the BP-trained weights, the final results are never worse than those obtained by BP alone. Therefore, the combination of the GAs with BP can provide fast and accurate results.

6. CONCLUSIONS

In this paper, we present the use of an AI integrated method (AIM) to predict permeability in a petroleum reservoir in offshore Australia. The method uses the neural-driven fuzzy reasoning combining with the floating-point encoding genetic algorithms. It is a robust and flexible estimator for industrial applications. In the field example, the results showed that AIM provided the smallest error on unseen data compared to the conventional method using only neural networks.

REFERENCES

1. Bloch, S. Empirical prediction of porosity and permeability in sandstones. *AAPG Bulletin*, 1991, **75**, 1145-1160.
2. Jian, F.X., Chork, C.Y., Taggart, I.J., McKay, D.M. and Barlett, R.M. A genetic approach to the prediction of petrophysical properties. *Journal of Petroleum Geology*, 1994, **17**, 71-88.
3. Wong, P.M. Taggart, I.J. and Gedeon, T.D. Use of neural network methods to predict porosity and permeability of a petroleum reservoir. *AI Applications*, 1995, **9**(2), 27-38.
4. Mohaghegh, S., Arefi, R., Ameri, S., Aminiand, K. and Nutter, R. Petroleum reservoir characterization with the aim of artificial neural networks. *Journal of Petroleum Science and Engineering*, 1996, **16**, 263-274.
5. Huang, Y., Wong, P.M. and Gedeon, T.D. An improved fuzzy neural network for permeability estimation from wireline logs in a petroleum reservoir, *Proceedings of IEEE Region Ten Conference (TENCON) on Digital Signal Processing Applications*, Perth, Australia, 1996 , 2, 912-917.
6. Gedeon, T.D., Wong, P.M., Huang, Y. and Chan, C. Two dimensional neural-fuzzy interpolation for spatial data. *Proceedings of GIS AM/FM ASIA '97 & Geoinformatics '97*, Taipei, Taiwan, 1997 , 1, 159-166.
7. Takagi, T. and Sugeno, M. Fuzzy identification of systems and its applications to modeling and control. *IEEE Transactions on* Systems, Man, & Cybernetics, 1985, **15**(1), 116-132.
8. Takagi, H. and Hayashi, I. NN-driven fuzzy reasoning. *International Journal of Approximate Reasoning*. 1991, **5**, 91-212.
9. Huang Y., Wong, P.M. and Gedeon T.D., Neural-fuzzy-genetic-algorithm interpolator in log analysis. *60th EAGE Conference and Technical Exhibition*, Leipzig, Germany, Extended Abstracts, 1998, 1, P106.
10. Huang, Y., Wong, P.M. and Gedeon, T.D. Prediction of reservoir permeability using genetic algorithms. *AI Applications*, 1998, **12**(1-3), 67-75.
11. Holland, J. *Adaptation in Natural and Artificial Systems.* Ann Harbor: University of Michican Press, 1975.
12. Wang, X. and Elbuluk, M. Neural network control of induction machines using genetic algorithm training. *Proceedings of the 31st IEEE IAS Annual Meeting*, 1996, 3, 1733-1740.
13. Goldberg, D.E. *Genetic Algorithms in Search, Optimization, and Machins Learning*. Addison-Wesley, Reading, MA, 1989.
14. Michalewicz, Z. *Genetic Algorithms + Data Structures = Evalution Programs*. Springer-Verlag, Berlin, 1994.