Nikolay I. Nikolaev, Hitoshi Iba, and Vanio Slavov

This chapter presents an immune version of Genetic Programming (GP). This is a GP version that conducts progressive search controled by a dynamic fitness function. The new fitness function is based on analogy with a model of the biological immune system, such that the programs are viewed as lymphocyte clones that compete to recognize most of the examples, viewed as antigens. The programs are reinforced with rewards for matched important examples and stimulated to match different examples. Examples recognized by a small number of programs are considered important. The motivation for using the immune dynamics for GP navigation is to maintain a high population diversity and to achieve enhanced search performance. Empirical evidence for the efficacy of this immune version on practical inductive machine learning and time-series prediction tasks is provided.

15.1 Introduction

Genetic Programming (GP) [Koza, 1992; Iba and Sato, 1992; Banzhaf et *al.*, 1998] is a stochastic search method suitable for addressing inductive learning tasks. Inspired by evolutionary processes in natural living organisms, the GP system maintains a population of programs to accomplish robust search. The search is considered robust when it locates a global, or nearly global, solution reliably. Such a behavior is not guaranteed, however, since the population is often trapped in local optima. If there are no intrinsic forces to push continuously the population on the search landscape, the system converges prematurely to a sub-optimal solution.

Recent studies into the principles of the biological immune system [Farmer et *al.*,1986; Bersini and Varela, 1991; Smith et *al.*,1993] provide innovative ideas about how to improve the search control and counteract early convergence to inferior solutions. They inspire us to develop an immune version of GP using the micro-mechanisms of the traditional GP and the dynamics of the immune system in order to achieve enhanced macroscopic performance. The motivation is the close similarity in the behavior of the immune system and the GP system: 1) they both perform search for generalizations of recognized patterns, that is, antigens or examples; and 2) they both use similar search mechanisms involving pattern matching, heuristic selection and modification of their hypothesis, namely antibodies or programs.

We employ an idiotypic network model of the immune system [De Boer and Hogeweg, 1989] and elaborate a dynamic fitness function which may be used in any GP implementation [Koza, 1992; Iba and Sato, 1992; Banzhaf et *al.*, 1998]. The fitness function sustains progress in the sense of improving search by driving the programs to compete continuously via mutual behavioral interactions, and to pursue evolution of a program that recognizes most of the examples. This fitness function consists of two dynamic models that exert influence on each other: 1) a model for propagating programs recognizing more important examples, and stimulating the programs to match examples from different subsets; and 2) a model for changing the importance of examples in dependence of the number of programs that recognize it.

The fitness function navigates the search through the interactions by encouraging complementary programs considered as a network. The network connectivity is a source of diversity. The diversity is a spontaneous macroproperty that enables continuous search influenced by the importance of the examples. Program reinforcements by rewards for matched important examples occasionally provoke network perturbations which contribute to moving the population on the landscape and enable avoidance of premature convergence.

We perform comparative studies of the evolutionary search performance of an immune version of the GP system STROGANOFF [Iba et *al.*, 1992, 1993, 1994], which differs from the original only in the use of the dynamic fitness function. STROGANOFF is implemented with programs representing multivariate, high order polynomials [Iba et *al.*, 1992, 1993, 1994] but uses the micromechanisms of inductive GP (iGP) [Nikolaev and Slavov, 1998]: proportional selection, biased context-preserving mutation, and biased crossover. The results are analysed using two kinds of measures: 1) estimates of the learning accuracy by the mean squared error attained by the trees in the population, and 2) estimates of the population diversity by the number of clusters in the population. Empirical investigations show that the search control by immune dynamics improves the performance of GP on practical inductive machine learning and time-series prediction tasks.

This chapter presents the immune version of GP in section 15.2, and the counterparts from biological immune networks, and the elaborated dynamic fitness function. Section 15.3 describes the micromechanisms of iGP. In section 15.4 we study the performance of immune GP on machine learning and time-series prediction problems from practice. Finally, a discussion is made and conclusions are derived.

15.2 Immune Version of the GP System

GP systems are suitable for inductive learning as they allow rapid exploration of huge search spaces. The search space of a problem may be considered a *fitness landscape* on which the search is navigated [Jones, 1995]. The fitness function determines the landscape ruggedness, and has a great impact on the GP system performance. When the fitness function supports diversity in the population, the GP system usually has the capacity to flow continuously on the landscape and to improve. The presented research suggests that such robust GP performance can be achieved not only by stimulating the programs to recognize more unmatched examples, but also by stimulating the programs to recognize examples from different subsets. We show that this could be fulfilled with a formula describing the dynamics of an idiotypic network model of the biological immune system.

15.2.1 Biological Idiotypic Networks

The biological *idiotypic networks* recognize and learn foreign *antigens* [Farmer et *al.*, 1986; Perelson, 1989]. An idiotypic network consists of *lymphocyte clones* that are pairs of a lymphocyte cell and one type of antibody on its surface. A lymphocyte has attached only one type of antibody, which may recognize different antigens. Pattern recognition occurs when a region from the antibody structurally binds a region from the antigen with complementary shape. Biological theory [Jerne, 1974] reveals that the lymphocytes make protective antibodies which not only recognize antigens, but also recognize other types of antibodies to preserve their own specificity. The portion of the antibody with which it matches another antibody is called *idiotype*, hence the antibody *interactions* lead to formation of idiotypic networks.

The idiotypic network acquires an immune response[1] against external antigens by concentration-proportional *clonal selection* and *differentiation* (*somatic hypermutation*) mechanisms. This kind of immune learning includes the following phases [Perelson, 1989]: 1) generation of lymphocyte cells with encoded diverse antibody types from the gene segments in the bone marrow; 2) stimulation of the lymphocyte clones that bind a large amount of structurally related antigens to secrete free antibodies and to reproduce; 3) differentiation of some activated lymphocyte clones; and 4) clonal selection of lymphocyte clones.

The *concentration* of lymphocyte clones increases proportionally to their involvement in response to antigens, but it is also influenced by the antibody interactions. Lymphocyte clones participate in idiotypic interactions that can be stimulatory or suppressing [De Boer and Hogeweg, 1989]. In this way, the clones control their abilities to learn, and they self-regulate by mutual interactions. The stimulation of the lymphocyte clones suggests that cell activation through cross-linking of antibody will increase insofar as the concentration of its complementary antibody increases.

15.2.2 Computational Counterparts in GP

Motivated by the powerful learning abilities of the immune idiotypic networks, we incorporate their principles in GP in order to achieve robust search navigation. The idea is to simulate the way in which the immune system fosters lymphocytes that match more antigens while at the same time stimulating the lymphocytes to complement each other. A slighly more specific interpretation of complementarity is assumed here. We use complementarity to encourage competition between the programs for evolving one program that matches most of the examples, rather than programs which together cover the examples.

[1]Immune response to antigen means that there is at least a partial match between an antibody receptor region and a complementary shaped antigen.

We develop an immune version of a traditional inductive GP system. The novelty is in the elaboration of a dynamic fitness function by analogy with the biological immune system counterparts as follows: a lymphocyte clone corresponds to a program; the concentration of a lymphocyte clone is the fitness of the program; the interaction between two antigens is the complementarity in the recognition potential of two programs; the antigens correspond to examples; the antigen concentration is the importance of an example; the clonal selection process is associated with fitness proportional selection.

15.2.3 The Dynamic Fitness Function

The *dynamic fitness function* includes two models that influence each other: 1) a model of the program dynamics; and 2) a model of the examples' dynamics. The fitness of a program should increase when it recognizes more examples, as the concentration of a lymphocyte increases when it detects more antigens.

The fitness function for control of the immune version of inductive GP is elaborated upon a dynamic model describing the changes in the concentrations of the lymphocyte clones [De Boer and Hogeweg, 1989]. The *program dynamics* F_i^{n+1} is formed from a permanent quantity for initial supply, plus an amount proportional to the previous fitness F_i^n and proliferation due to arousal by recognized examples and evoked excitory interactions, without a constant death rate:

$$F_i^{n+1} = Z + F_i^n \cdot (\ p \cdot \mathrm{Prol}(Ag_i^n,\, \mathrm{Id}_i^n) - \ d\ )$$

where: Z is influx constant, d is turnover constant and p is proliferation constant;
Ag_i^n is the antigen score of the i-th program at generation n;
Id_i^n is the total anti-idiotype excitation of program i;
Prol is the proliferation function.

The *proliferation* Prol of a program i in the next generation according to this difference equation model depends not only on the recognized examples Ag_i^n, but also on the extent of its interactions Id_i^n with the other programs:

$$\mathrm{Prol}(Ag_i^n,\, \mathrm{Id}_i^n) = \frac{Ag_i^n + \mathrm{Id}_i^n}{p_1 + Ag_i^n + \mathrm{Id}_i^n}$$

where p_1 is a free constant parameter. The original proliferation function of the immune system is bell-shaped [De Boer and Hogeweg, 1989] but we use here only the exciting part of the bell-shaped curve. In this way programs with high idiotypic interactions are propagated to survive.

The antigen score of a program should account how many examples it matches and also it should depend on the importance of these eliciting examples. This is because the antigen of a lymphocyte clone depends on the number and on the specificity of the structurally related antigens that it binds. We define the *antigen score* Ag_i^n of program i as linearly proportional to the importance I_j^n of the examples R which it matches, $1 \leq j \leq R$, among all N_E examples, $R \leq N_E$:

$$Ag_i^n = \sum_{j=1, i \neq j}^{R} B_{ij} \cdot I_j^n$$

where the binding B_{ij} of a program i is 1 if the program recognizes the example j and 0 otherwise. Such a definition holds for discrete program outcomes, while for continuous outcomes it should be redefined (see section 15.4.4).

Our suggestion is that the program fitness dynamics should be in interplay with the dynamics of the examples. This resembles the objective of the biological immune system to recognize all antigens if possible. During the acquisition of an immune response, the idiotypic network topology changes to accommodate lymphocyte clones that match exceptional antigens. Making an association with the iGP system behaviour, if more programs recognize an example then it should become less attractive. Therefore, when a small number of unmatched exceptional examples remain, they have to reinforce these programs in the population which cover them, and so provoke perturbation of the search process. The *importance* I_e^n of an example e is defined as proportional to the number of programs in the population that correctly recognize it plus a term for constant recruitment γ_e:

$$I_e^{n+1} = I_e^n \cdot (\alpha - \sum_{j=1}^{N} B_{je} \cdot F_j^n / (F_{\max}^n \cdot N)) + \gamma_e$$

where α is a free constant parameter.

Specifying the program interactions A_{ij} is essential for driving the programs to compete and for supporting diversity. In the immune network the lymphocyte clones should be complementary in shape to antigens in order to defeat them. In context of inductive GP, this complementarity could be regarded as a behavioral characteristic. The programs will have complementary behaviour if the affinity accounts for the difference in their mutual learning potential. Two programs are considered behaviorally complementary when they recognize examples from disjoint sets. We define the *affinity* A_{ij} between two programs i and j as the set difference between the subset of examples which i recognizes and the subset which j recognizes:

$$A_{ij} = |E_i^n - E_j^n|$$

where E_i^n is the subset of examples correctly recognized by program i at generation n from all provided examples E: $E_i^n \subseteq E$, $E_j^n \subseteq E$.

The *network* comprises all programs in the population. We realize that the network is symmetric $A_{ij} = A_{ji}$ for $1 \leq i, j \leq N$, and also that the programs do not recognize self $A_{ii} = 0$. The affinity A_{ij} stimulates the programs to: 1) recognize more examples; and 2) match slightly overlapping sets of examples. Therefore, the fitness function with this affinity contributes for breeding distinct, non-similar programs. Such program interactions entail diversity in the population.

The idiotypic influence among the programs should estimate their mutual behavioral complementarity through the affinity interactions. The biological immune system uses such a factor to self-regulate so that the lymphocyte clones have together the power to detect and eliminate all the available antigens. We define the *anti-idiotype excitation* Id_i^n to favor a program i if its interaction A_{ij} with the other $1 \leq j \leq N$ programs in the population is high:

$$\mathrm{Id}_i^n = (1/(N_E \cdot N)) \cdot \sum_{j=1, i \neq j}^{R} A_{ij} \cdot F_j^n$$

The evolution of the network topology determines the ability of the immune algorithm to conduct efficient search. It is important to reason how the network connectivity will correspond to the phases of evolutionary search. The GP system usually starts with a random initial population and the global excitation is large. When GP performs global search, the interactions should be relatively high indicating exploration of large landscape areas. During local search the wiring should be low as GP exploits landscape areas in the vicinities of the reached local optima. This holds even if there are a number of optima peaks located by the system.

In the experiments below we use reference values for the parameters in the fitness function as follows: $Z = 0.1$, $d = 0.5$, $p_1 = 0.25$, $p = 1.1$, $\gamma_e = 0.001$, and $\alpha = 1.025$. The initial fitness of the programs is calculated by solving the above difference equation for a steady state $F_i^0 = F_i^{n+1}$, which leads to the formula: $F_i^0 = Z(p_1 + 1)/(p_1 + 1 + d + dp_1 - p)$.

15.3 Micromechanisms of the Inductive GP

The immune inductive GP is developed with programs representing *multivariate trees* [Iba et *al.*, 1992, 1993, 1994]. We use the following micromechanisms *fitness proportional selection*, *context-preserving mutation* and *crossover* operators, *biased* by the size of the programs [Nikolaev and Slavov, 1998]. Size proportional biasing of the application of the genetic operators is necessary in evolutionary algorithms with variable-length genomes to counteract the tree bloat phenomena [Langdon and Poli, 1998]. Such biased applications of the operators act as forces that guard against degenerated behavior by deporting the system to unseen landscape areas.

15.3.1 Inductive Learning and Regression

The GP method is useful for addressing inductive learning tasks. The problem of these tasks can be formulated as a multivariate regression problem. Given instantiated vectors of several independent variables, that is patterns $\mathbf{x}_i = \{x_{i1}, x_{i2}, ..., x_{il}\}$, and corresponding values $r_i \in \mathcal{R}$ of the dependent variable y_i, as predefined examples $E_i = \{(\mathbf{x}_i, y_i) | y_i = r_i \in \mathcal{R}\}$, the goal is to find a function mapping $y = f(\mathbf{x})$. This is a general definition since solving regression problems implies abilities for solving classification problems as well, where the dependent variable is discrete, often categorical: $E_i = \{(\mathbf{x}_i, y_i) | y_i = d_i \in \mathcal{N}\}$. In essence, GP is an evolutionary search paradigm suitable for learning a functional description that best approximates the dependent variable. The intention is to use this function for predicting real values associated with unknown example vectors.

15.3.2 Multivariate Trees

High-order multivariate polynomial regression models can be represented by trees with functions in the internal nodes, and independent variables in the leaves [Iba et *al.*, 1992, 1993, 1994]. A *multivariate tree* hierarchy of non-linear polynomial combinations may capture very accurately the dependencies in the examples.

There are three important implementation issues in the construction of evolutionary GP systems with multivariate trees: 1) how to find the coefficients in the polynomials; 2) what kind of polynomials to use and with which variables; and 3) how to avoid overfitting with the provided examples.

Recent research on learning by inductive GP demonstrated results of high accuracies using binary multivariate trees with second-order polynomials [Iba et *al.*, 1992, 1993, 1994]. The advantage of such trees is that they enable evaluation and finding of complex, high-order models for acceptable time by composing simple, second-order models with coefficients that are computed relatively fast. We adopt the multivariate tree regression model consisting of *cascaded quadratic polynomials* from the polynomial theory of complex systems [Ivakhnenko, 1971]:

$$y(\mathbf{x}) = \sum_{i=0}^{5} a_i z_i(\mathbf{x}) = a_0 z_0(\mathbf{x}) + a_1 z_1(\mathbf{x}) + a_2 z_2(\mathbf{x}) + a_3 z_3(\mathbf{x}) + a_4 z_4(\mathbf{x}) + a_5 z_5(\mathbf{x})$$

of terms: $z_0(\mathbf{x}) = 1$, $z_1(\mathbf{x}) = x_1$, $z_2(\mathbf{x}) = x_2$, $z_3(\mathbf{x}) = x_1 x_2$, $z_4(\mathbf{x}) = x_1^2$, and $z_5(\mathbf{x}) = x_2^2$. Considering the functions z_i as a vector $\mathbf{z} =(z_0(\mathbf{x}), z_1(\mathbf{x}), z_2(\mathbf{x}), z_3(\mathbf{x}), z_4(\mathbf{x}), z_5(\mathbf{x}))$, we may write:

$$y(\mathbf{x}) = \mathbf{a}^T \mathbf{z}(\mathbf{x})$$

A multivariate tree has such polynomials in the lowest functional nodes with leaf children, and independent variables in the leaves. Higher in the tree the same components are employed, where the variables may be outcomes of lower level polynomials: $z(xy)$ or $z(\mathbf{y}) = y_1 y_2$. Using these trees, polynomial approximation may be pursued by attempts to reduce the deviation when modelling data. According to the Group Method of Data Handling (GMDH) [Ivakhnenko, 1971] the polynomial coefficients at each node of the tree-like multilayer program structure are calculated in stepwise manner by the matrix formula:

$$\mathbf{a} = (\mathbf{Z}^T\mathbf{Z})^{-1}\mathbf{Z}^T\mathbf{y}$$

where $\mathbf{Z}$ is the 6×6 matrix of vectors $\mathbf{z}_i = \{z_{i0}, z_{i1}, ..., z_{i5}\}$, $i = 1..N_E$, and $\mathbf{y}$ is the output vector. This is known as a solution of the general least-squares fitting problem by the method of normal equations. The normal equations we solve using LU-decomposition [Press et *al.*, 1992].

Following the GMDH algorithm, the evaluation of a tree-like polynomial starts from the lowest functional nodes, and then proceeding up to the tree root. The output vector at each step is used as independent variable vector for finding the coefficients in the next step higher in the tree. When a tree is modified, the coefficients must be recalculated to reflect the modification by mutation or crossover [Iba et *al.*, 1993]. The recent implementations save some computations by using the old coefficients from the subtrees not affected by the genetic operators.

15.3.3 Context-Preserving Mutation

Inductive GP uses a *context-preserving mutation* operator [Nikolaev and Slavov, 1998]. It transforms a multivariate tree so that only the closest to the chosen mutation point neighboring vertices are affected. The elementary constituent submutations are: 1) substitution of a functional node by another one, or random substitution of a terminal; 2) insertion of a node as a parent of a subtree so that the subtree becomes leftmost child of the new node; and 3) deletion of a node only when no subtree below is to be cut.

The *biased mutation* operator for inductive GP performs context-preserving mutation with probability $p_m = m \cdot |g|^2$, where m is a free parameter [Goldberg et *al.*, 1989]. This operator usually modifies large programs.

15.3.4 Crossover by Cut and Splice

The *biased crossover* for inductive GP splices two multivariate trees with probability $p_c = c/\sqrt{|g|}$, where c is a free parameter, and $|g|$ is size of tree g, or swaps them. The cut points are selected randomly. This operator produces offspring with larger size than their parents if the parents are of very small size.

The proper values for the free parameters m and c are identified with the autocorrelation function. These free parameters m and c serve as knobs with which one may regulate carefully the search efficiency [Nikolaev and Slavov, 1998].

15.4 Practical Induction by Immune Dynamics

15.4.1 Traditional and Immune Versions of iGP

The evolutionary performance of the immune version of GP is studied here in comparison with the traditional GP system STROGANOFF [Iba et *al.*, 1993, 1994] that uses the multivariate tree representation from the previous section. STROGANOFF was implemented with the micromechanisms of inductive GP given above [Nikolaev and Slavov, 1998]: biased context-preserving mutation, biased crossover by cut and splice, and a stochastic complexity (MDL) fitness function with an improved complexity component:

$$F_{MDL} = k \cdot S_i^2 + \log(n_t + n_f) + n_t + n_t \cdot \log T + n_f + n_f \cdot \log F$$

where n_t are the leaves in the tree, n_f are the functional nodes, T are all terminals, F are all functions, k is a balancing parameter, and S_i^2 is the mean squared error:

$$S_i^2 = (1/N_E) \cdot \sum_{i=1}^{N_E} |y_i - y(\mathbf{x}_i)|^2$$

where y_i is the true outcome given with the i-th example, and $y(\mathbf{x}_i)$ is the estimated outcome from the program given the i-th input vector $\mathbf{x}$.

The only difference between STROGANOFF and the immune version is that the fitness in the immune iGP is calculated with the difference equation for F_i^{n+1}.

15.4.2 Performance Measures

The evolutionary performance of the traditional and the immune versions of inductive GP is evaluated with two kinds of measures: 1) estimates of the *learning accuracy*, and 2) estimates of the *population diversity*. The learning accuracy provides quantitative evidence for the development of the learning process. The diversity is a macroproperty that enhances the GP system potential to find highly fit programs.

The diversity can be analyzed with the *number of clusters* in the population [Bersini, 1997], computed using the K-means clustering algorithm [Hartigan and Wong, 1979]: split the current population of multivariate trees into groups by maximizing the tree-to-tree distance between them. The tree-to-tree distance here is the minimal number of context-preserving mutations necessary to produce one of the trees from the other [Nikolaev and Slavov, 1998].

With the experiments below we show that the immune version attains fitter programs and maintains higher population variety than the traditional inductive GP on machine learning and time-series prediction problem instances. All of the plots in the figures that follow are from the best runs, taken among the 1000 runs which were performed with each the immune iGP and STROGANOFF.

15.4.3 Machine Learning

Machine learning considers computational induction as a problem of finding a function f from provided $\{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_1), ..., (\mathbf{x}_N, y_C)\}$ examples. The task is to acquire such a function that will distinguish correctly unseen examples of the same classes. The dependent variable assumes discrete values, often denoting only several categories $y_i = d_i \in \mathcal{N}$, which are called classes.

The Glass recognition task is assumed as a benchmark machine learning problem. The goal is to learn the configuration of glass pieces collected after a car accident [Merz and Murphy, 1998]. The Glass data set includes $|E| = 214$ examples of 9 numeric features $E_i = \{(\mathbf{x}_i, y_i) | x_{i1} = r_1 \in \mathcal{R}, ..., x_{i9} = r_9 \in \mathcal{R}\}$, each associated with one of 6 classes: $y_1 = 1, y_2 = 2, ..., y_6 = 6$.

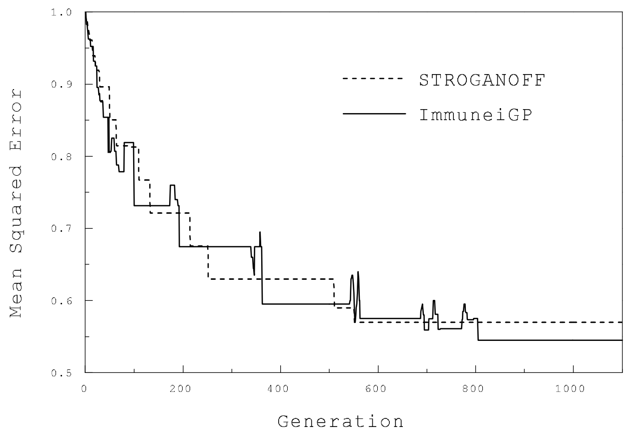
We compare inductive GP with the Non-linear Decision Trees algorithm (NDT) [Ittner and Schlosser, 1995] since it produces multivariate binary tree classifiers with quadratic polynomials of features in the nodes as inductive GP. This means that NDT is similar to inductive GP in that it also makes non-linear partitioning of the feature space. NDT differs in that it performs top-down induction of decision trees having the classes in their leaves. Several polynomials generalizing the examples are extracted from the decision tree by traversing it from the root to the leaves, while GP generates one polynomial. The NDT algorithm outperforms some of the best machine learning approaches [Ittner and Schlosser, 1995].

The presented results are derived using 10-fold cross validation: 10 disjoint subsets of 21 examples were formed, then the algorithms were trained 10 times using the outside $214 - 21$ examples of each partition and tested on the remaining 21 examples. We study the learning accuracy with the percentage of correctly recognized examples by the best tree. The best tree accuracies can be obtained from Table 15.1 by summation of the accuracies in the second column with the standard deviations in the third column. For example, the best tree found by the immune iGP from all 1000 runs was with 72.11% accuracy. One can see in Table 15.1 that the accuracies of the NDT trees deviate less, but the iGP may discover more accurate trees. An advantage of the two GP systems is that they derive smaller trees. NDT tends to overfit the examples due to the recursive partitioning of the examples.

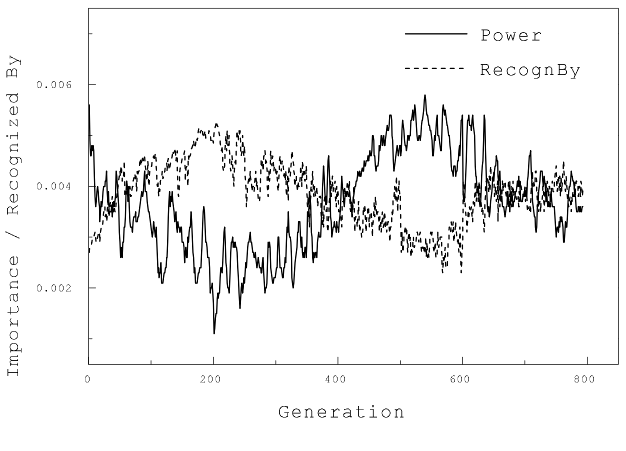
Table 15.1
Learning accuracy of the best multivariate trees produced by the NDT, the immune iGP and the traditional STROGANOFF with the Glass recognition data

Approach	*Best Tree Accuracy*	*Standard Deviation*	*Best Tree Size*
NDT	67.43%	±2.53%	29
Immune inductive GP	68.22%	±3.89%	24
STROGANOFF	67.78%	±3.66%	25

The *learning performance* of the immune inductive GP and STROGANOFF may be analyzed with the curves given in Figure 15.1a,b. The mean squared error of the best tree recorded with the immune GP is occasionally perturbed by the system dynamics, which enables its progressive improvement. We note that after phases of local search between generations $350 - 540$, $560 - 690$, $720 - 770$, the system conducts global search into different directions across the fitness landscape.



a)



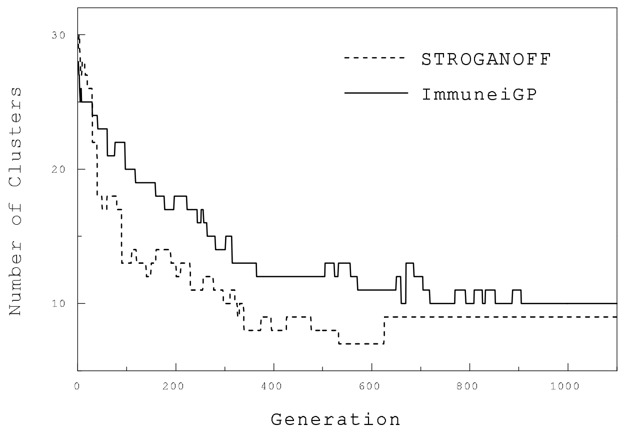
b)

Figure 15.1
Learning accuracy: a) Mean squared error, of the best tree evolved by the immune iGP and STROGANOFF using *MaxSizeGP*=29, *PopSize*=60, and the Glass data, and b) Influence of a particular important example on the fitnesses of programs that recognize it

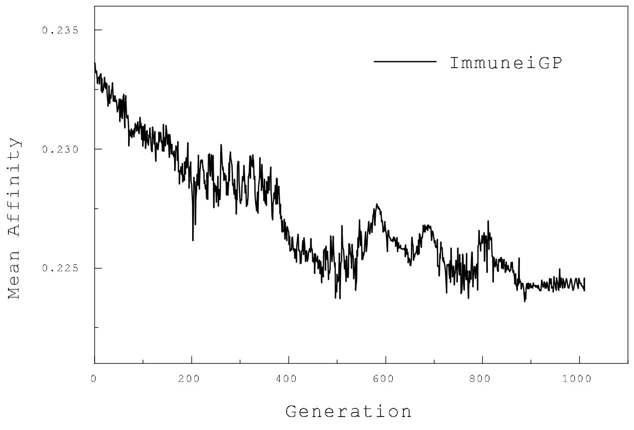
The reasons for such perturbations may be discovered by an experimental analysis of the changes in the examples' importance during a typical run of the immune iGP. We conducted an experiment with the immune iGP and selected a particular important example, in order to investigate how it is approached by the programs in the population in the sense of whether depending on its importance they are attracted to match it. Figure 15.1b shows the interplay between the changes of the importance of the selected important example and the number of programs in the population that recognize it. This figure displays that there is an inverse relation between the example's importance and the number of the programs in the population recognizing it. When more programs tend to recognize the example its power decreases, in the interval $20-200$ generation. After that, between $220-400$ generation, the programs begin to avoid this example since its importance has become low which means that it is no more so attractive. This causes again an increase of the example's importance, between generations $220-600$, as there are a small number of programs in the population that match it. Next, they strive to recognize it and decrease the example's importance again. Therefore, the changes in the importances of the examples really drive the programs to match more powerful examples and so contribute to reorientation of the evolutionary search.

The *population diversity* maintained by the immune iGP and STROGANOFF can be analyzed with the two plots in Figure 15.2. In Figure 15.2a we observe that the immune version of inductive GP sustains higher population diversity than STROGANOFF measured by the number of clusters of programs. Since the number of clusters is measured with the tree-to-tree distance characteristics between the tree-like programs in the population, one may assume that this is an estimate of the syntactic variety in the population during evolutionary search. It is interesting to note that the number of clusters during evolutionary search conducted by immune iGP slightly decreases when this system performs local search. This is evident from the horizontal line segments which correspond to the local search periods in Figure 15.1. The performance of the traditional inductive system STROGANOFF maintains almost 25% less clusters of programs.

Another measure of the diversity is the mean anti-idiotype excitation among the programs in the population (Figure 15.2b). The mean affinity shown in this figure presents the diversity from a different, semantic perspective as it estimates the variety in the recognition potential of the programs, or whether there are programs in the population with non-similar learning capacity. Figure 15.2b demonstrates that the mutual interactions between the programs in the immune iGP are occasionally perturbed. These could be considered search perturbations caused by changing the importances of some examples. The search obviously is kind of a shaken and next improves. This claim can be explained with the varying sloping down affinity curve on Figure 15.2b, which falls and raisings delimit approximately the same periods of local search between generations $380-580$, $590-680$, and $700-790$.



a)



b)

Figure 15.2
Population diversity: a) Clustering, recorded with the immune iGP and STROGANOFF using *MaxSizeGP*=29 and *PopSize*=60, on the Glass data and b) Mean anti-idiotype excitation

15.4.4 Time-Series Prediction

Time-series prediction may also be regarded as an inductive problem. The task is to identify the regularities among given series of points: ..., x^t, x^{t+1}, x^{t+2}, ..., sampled at discrete time intervals. This is accomplished by search for a series description with only the most relevant from the available points, so that future points of the series can be predicted. We assume an autoregressive model with non-linear polynomials, represented as multivariate trees. The available points are considered coordinates of the independent variable vector and serve as data for learning. The independent variable vectors $\mathbf{x}_i$ are created with embedding dimension k, and delay time $\tau = 1$ [Farmer and Sidorowich, 1987]:

$$\mathbf{x}_i = \{ \ x^{t-(k-1)\cdot\tau}, x^{t-(k-2)\cdot\tau}, ..., x^t \}$$

that is window vectors from k nearest previous points starting at a point t. Dependent variable is the immediate next point to the starting $y_i = x^{t+1}$. Thus, examples of the kind $E_i = \{(\mathbf{x}_i, y_i) | y_i = r_i \in \mathcal{R}\}$ are formed. Difficulties in time-series prediction arise from the high dimensionalities of the series. That is why, the choice of the embedding dimension k and delay time τ are important design issues. We select empirically[2] embedding dimension $k = 10$, and delay time $\tau = 1$.

The immune iGP for time-series prediction is implemented with modified binding B_{ij} of a program i to recognized examples since the outcome $y(\mathbf{x}_j)$ is continuous:

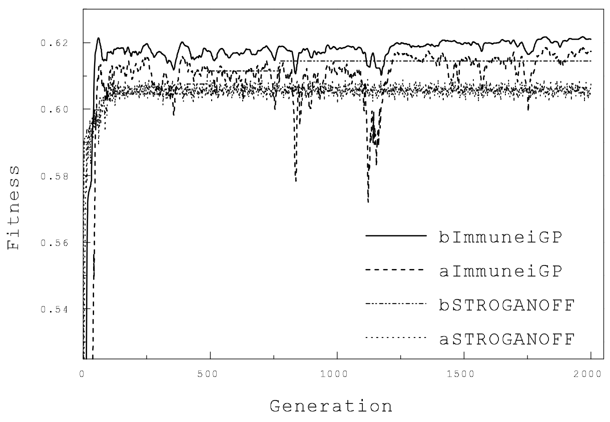
$$B_{ij} = \begin{cases} 1 - S_i^2/S_A^2 & \textit{if } S_{ij}^2 < S_A^2 \\ 0 & \textit{otherwise} \end{cases}$$

where $S_{ij}^2 = |y_i - y(\mathbf{x}_j)|^2$ is the squared error between the outcome $y(\mathbf{x}_j)$ of program i given input $\mathbf{x}$, and y_i is the true outcome given with example j. The mean squared error S_i^2 is computed with the formula defined in section 15.4.1. The value of S_A^2 is calculated as the average mean squared error from all programs. The rationale is that the binding B_{ij} should be large enough in order to influence essentially the fitness function through the fitting error.

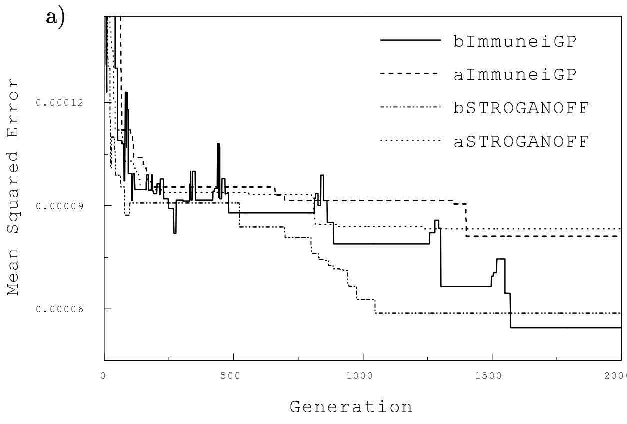
We used data series produced with the Mackey-Glass differential equation for prediction [Mackey and Glass, 1977]. This is an equation for simulating blood flood caused by the irregularity of the heart beats. Trajectories of 1400 points were generated with parameters: $a = 0.2, b = 0.1$, and three differential delays $\Delta = 17$, $\Delta = 23$, and $\Delta = 30$. The initial points in each of the cases were taken randomly. The first subseries of 1000 points were discarded. We considered the next 100 points for training, and the remaining 300 points for testing.

The plots in Figure 15.3 provide empirical evidence for the *learning performance* of the immune iGP and STROGANOFF. One observes that the immune iGP exhibits stable evolutionary performance, in the sense that it is able to search progressively. Despite some degradations in its performance, it continuously improves since the dynamics sustains the motion of the population on the fitness landscape.

[2]The theoretical studies advise to select embedding dimensions k corresponding to the concrete attractor dimensions D with the inequality $k \geq D + 1$ [Farmer and Sidorowich, 1987].



a)



b)

Figure 15.3
Learning accuracy: a) Average and best fitnesses, and b) Mean squared error, recorded with the immune iGP and STROGANOFF using *MaxSizeGP*=29, *PopSize*=60, and 100 points from the Mackey-Glass equation series derived with $\Delta = 17$

The phases of global and local search conducted by the immune version of GP can be clearly identified. During local search the mean squared error does not change, shown by the regions with horizontal lines in Figure 15.3b between generations $490 - 750, 820 - 1250, 1300 - 1500, 1540 - 2000$. These evidence for local search are supported by the horizontal slightly oscillating fitness values in the same generation intervals in Figure 15.3a. When global search is performed near generations $470, 780, 1275,$ and 1520, the mean squared error sometimes fluctuates (Figure

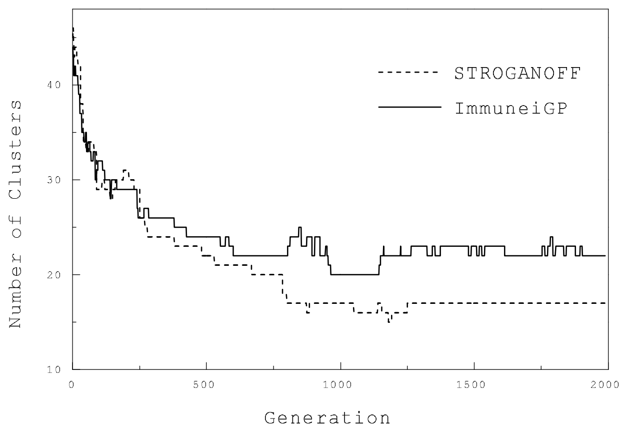
15.3b), and after that there are sharp falls of the corresponding average and best fitnesses (Figure 15.3a). These sudden performance changes cause stepwise error decrease and finding more accurate best programs (Figure 15.3a,b). The traditional GP climbs on a local peak and remains there (after generation 1050 in Figure 15.3b), since the locally optimal program takes over the population.

Table 15.2
Learning accuracy, forecast and size of the best trees found by the immune iGP version and STROGANOFF, on Mackey-Glass equation series derived using: embedding dimension $k = 10$, delay time $\tau = 1$, and three differential delays Δ (the errors are in units of $1e - 5$)

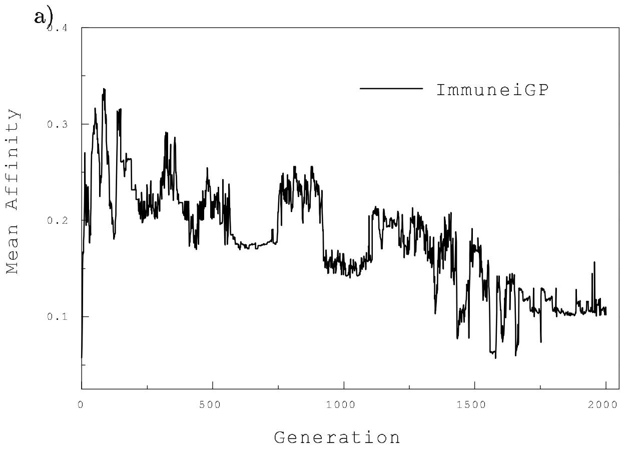
	Mean Squared Error S_i^2		
	Best Tree TRAIN Error		
differential delay	$\Delta = 17$	$\Delta = 23$	$\Delta = 30$
Immune inductive GP	5.458376	8.137821	9.015442
STROGANOFF	5.870631	8.458412	9.527129
	Best Tree TEST Error		
Immune inductive GP	6.236278	8.852356	9.798612
STROGANOFF	6.762336	8.817821	10.153613
	Best Program Tree Size		
Immune inductive GP	$n_f = 10$, $n_t = 11$	$n_f = 11$, $n_t = 12$	$n_f = 13$, $n_t = 14$
STROGANOFF	$n_f = 9$, $n_t = 10$	$n_f = 11$, $n_t = 12$	$n_f = 12$, $n_t = 13$

In Table 15.2 we show the accuracies of the best trees from 1000 runs recorded with series derived with a slightly large embedding dimension, which amplifies the noise in the examples, but facilitates the comparison with other GP approaches to the same problem instance [Iba et *al.*, 1993; Zhang et *al.*, 1997; Mulloy et *al.*, 1996]. We generated three series with different differential delays Δ and, thus, we produced three Mackey-Glass equation series of different complexities [Farmer and Sidorowich, 1987]. The first 17, respectively 23 and 30 points in each series, were randomly generated. The most frequently changing curve and hence most difficult to learn is the one obtained with $\Delta = 30$. It is visible that the immune version of iGP identifies solutions with higher approximation accuracy as well as predictability both on the training and on the testing subseries. The immune GP is stable as it shows abilities to search better in cases of data series with different complexities.

The lowest section of Table 15.2 displays the program sizes, given by the number of functional nodes n_f and terminal leaves n_t of the best multivariate tree found. The differences in the program complexities justify the accuracy deviations between the training and testing errors. The trees evolved by the traditional GP have a smaller number of functional nodes and terminal leaves, but they have lower accuracies. An advantage of the immune GP is in the high quality of the attained multivariate trees, and so suggest that the trees evolved by the immune GP are better approximators. Therefore, the experiments indicate that the immune iGP usually discovers best multivariate trees of larger size, which fit better the examples than predict. This, however, can be regulated by tuning the stoping criterion in the sense of number of generations to evolve.



a)



b)

Figure 15.4
Population diversity: a) Clustering, recorded with the immune iGP and STROGANOFF using *MaxSizeGP*=29, *PopSize*=60, and 100 data points from the Mackey-Glass equation series derived with $\Delta = 17$; and b) Mean anti-idiotype excitation

We demonstrate in Figure 15.4a the changes of the *population diversity*. The numbers of clusters are taken during the same run for time-series prediction as above, in order to explain why the GP system behavior has the performance characteristics displayed in figures 15.3a and 15.3b. It seems that the immune version of iGP enforces the programs to occupy different fitness landscape areas, and keeps the population distributed. Initially, in the phase of global search the number of clusters is relatively high. During local search phases the number of clusters slightly diminishes and stays unchanged between $550 - 750$ and $900 - 1150$ generations. When the GP system performs global search the number of clusters increases and fluctuates. The traditional inductive GP has been deceived in the vicinity of some local optima after generation 1500 and further can not discover better solutions.

The network of programs in the immune version of iGP remembers examples by switching between different combinations of programs. That is why, the population diversity can be estimated by the evolution of the network connectivity since it influences the distribution of the search effort on different landscape areas. We evaluate the strength of idiotypic interactions with the affinities between the programs (Figure 15.4b). The high mean affinity in Figure 15.4b is an indication for the high diversity supported by the immune GP. It is interesting to observe that the affinity decreases between generations $550 - 750$ and $900 - 1150$, which makes us certain that search perturbations really occur during these periods.

15.5 Discussion

Understanding the intrinsic dynamics of the immune inductive GP is important for control of the evolutionary search carried by them.

The dynamics of the immune version of GP has three aspects [Farmer et *al.*, 1990]. The first aspect is the net topology dynamics. In our implementation we assumed that the network is symmetric and completely connected, but these are simplifications. The choice of programs for interaction should be suggested by an appropriate reformulation of the affinity formula. Theoretically, the network should comprise programs interacting with a small number of other programs. The affinity formula should not stimulate, however, formation of very sparse networks since they make it impossible for perturbations to occur.

The second aspect is the parameter dynamics. The large number of free parameters in the fitness formula creates difficulties for finding and tuning their values. The stability of the results is sensitive to these values, and the problem is whether the selected reference parameter values are the most relevant ones.

The third aspect is the concentration model dynamics of the lymphocyte-like programs. This dynamics is very difficult to analyze because of the bell-shaped character of the activating proliferation function.

15.6 Relevance to Other Works

The time-series prediction problem have been also addressed by many other GP systems [Oakley, 1994; Zhang et *al.*, 1997; Mulloy et *al.*, 1997]. All these GP systems use static fitness functions: Oakley [1994] uses the sum of squared errors, while Mulloy and colleagues [1997] use an adjusted version of the sum of squared errors. The Minimum Description Length (MDL) fitness function of Zhang, Ohm, and Muhlenbein [1997] is close to this which was employed in STROGANOFF in that it balances between the fitness and the program size, leading to high quality approximations. These static fitness functions, however, lack of intrinsic power to push the GP system to improve continuously its evolutionary search performance. This is because the static fitness functions do not have a reinforcing effect to counteract premature convergence to suboptimal programs.

An essential difference in the design of the above GP systems, the immune GP system and the STROGANOFF system is that the last two produce polynomials represented as specific cascaded multivariate trees. The polynomial coefficients in these specific multivariate trees are directly computed as least-squares solutions by the method of normal equations, which avoids the need to evolve them. Thus, the GP systems like STROGANOFF achieve very accuracte results trying only to evolve the basis polynomials which constitute the target polynomial.

Empirical investigations of the performance of the immune version and the GP version STROGANOFF have been presented [Iba et *al.*, 1993]. The accuracies of the programs induced by the immune GP are slightly better than these of STROGANOFF. This is reasonable since the inductive learning problems that were used are known as hard benchmark instances for learning. The most significant result is that the immune dynamics reduces the probability of GP to become stuck in local optima. This impacts the GP system abilities to move continuously the population on the fitness landscape. Our hypothesis is that this robust performance of the immune version of GP, is due to two factors: 1) the rewarding program interactions; and 2) the reinforcing examples' importance changes. These provoke occasional network perturbations, and so contribute to the search reorientation toward different fitness landscape areas.

The presented fitness function is a dynamic model that distributes the population into niches [Horn and Goldberg, 1996]. This dynamic function makes coexistent in the population programs from different niches on the fitness landscape, which is a kind of fitness sharing. Fitness sharing means that the fitness of a program decreases if there are similar, slightly behaviorally different programs than it. In the immune GP the fitness sharing is implicit, and it is achieved through resource sharing with the affinity interactions. The immune dynamics via such affinity interactions encourages competition between the programs to cover more example resources from distinct resource niches, and does not divide the reward. Therefore, the immune

GP exhibits implicit niching like the classifier system of Horn and Goldberg [1996]. The immune GP differs in that the programs partially recognize the examples and mutually compete till one program individually attains a complete solution.

15.7 Conclusion

This chapter has proposed an inductive GP system navigated by a fitness function based upon a network model of the biological immune system. Viewing the learning as immunity phenomena, it is an attempt to employ the immune idiotypic networks as inductive computational mechanisms. The immune version of GP is not specific to any genetic mutation or crossover operators, and can be used for improving any traditional GP system. The implementation of the immune GP version is more sophisticated but features more learning power.

It has been reported that the immune GP system outperforms STROGANOFF in solving benchmark machine learning and time-series prediction tasks. We are inclined to think that the robust evolutionary behaviour will be retained if the same micromechanisms and dynamic fitness function are used. Because of computational efficiency reasons the immune GP will be particularly useful for machine learning tasks with close categorical examples distribution, and for time-series prediction tasks with close series size and embedding dimension to these studied here.

Further research should be directed toward deeper understanding of the computational properties and principles of this connectionist immune version of inductive Genetic Programming. Theoretical analysis should be made to clarify what this computer immune system can do.

Bibliography

Banzhaf, W., Nordin, P., Keller, R. E. and Francone, F. D. (1998). *Genetic Programming: An Introduction. On the Automatic Evolution of Computer Programs and Its Applications.* San Francisco, CA: Morgan Kaufmann.

Bersini, H. and Varela, F. (1991). Hints for Adaptive Problem Solving Gleaned from Immune Networks. In: Schwefel, H. P. and Mühlenbein, H. M. (eds.), *Proceedings of the First International Conference Parallel Problem Solving from Nature*, PPSN I, Berlin: Springer, 343-354.

Bersini, H. (1997). Frustration and Clustering in Biological Networks. In: Langdon, C. G. and Shimohara, T. (Eds.), *Artificial Life V: Proceedings of the of the Fifth International Workshop on the Synthesis and Simulation of Living Systems*, Cambridge, MA: The MIT Press.

De Boer, R. G. and Hogeweg, P. (1989). Idiotypic Networks Incorporating T-B Cell Cooperation. The Condition for Percolation. *Journal of Theoretical Biology.* 139, 17-38.

Farmer, J. D., Packard, N. H. and Perelson, A. S. (1986). The Immune System, Adaptation and Machine Learning. *Physica*, **22**D, 187-204.

Farmer, J. D. and Sidorowich, J. J. (1987). Predicting Chaotic Time Series. *Physical Review Letters,* **8***(59): 845-848.*

Farmer, J. D. (1990). A Rosetta Stone for Connectionism. *Physica*, **42**D, 153-187.

Goldberg, D., Korb, B. and Deb, K. (1989). Messy Genetic Algorithms: Motivation, Analysis and First Results. *Complex Systems*, **3**:493-530.

Hartigan, J. A. and Wong, M. A. (1979). A K-means clustering algorithm, *Applied Statistics,* **28***(1): 100-108.*

Horn, J. and Goldberg, D. E. (1996). Natural Niching for Evolving Cooperative Classifiers. In: Koza, J. R., Goldberg, D. E., Fogel, D. E. and Riolo, R. L. (Eds.), *Genetic Programming 1996: Proceedings of the First Annual Conference*, Cambridge, MA: The MIT Press, 553-564.

Iba, H. and Sato, T. (1992). Meta-level Strategy for Genetic Algorithms based on Structured Representations, In: *Proceedings of the Second Pacific Rim International Conference on Artificial Intelligence*, 548-554.

Iba, H., Kurita, T., de Garis, H. and Sato, T. (1993). System Identification using Structured Genetic Algorithms. In: Forrest, S. (Ed.), *Proceedings of the Fifth International Conference on Genetic Algorithms*, ICGA-93. San Mateo, CA: Morgan Kaufmann, 279-286.

Iba, H., de Garis, H. and Sato, T. (1994). Genetic Programming using a Minimum Description Length Principle. In: Kinnear, K. (Ed.), *Advances in Genetic Programming*, Cambridge, MA: The MIT Press, 265-284.

Ittner, A. and Schlosser, M. (1995). Non-Linear Decision Trees, In: Saitta, L. (Ed.), *Machine Learning: Proceedings of the 13th International Conference, ICML'96,* San Mateo, CA: Morgan Kaufmann, 252-257.

Ivakhnenko, A. G. (1971). Polynomial Theory of Complex Systems. *IEEE Trans. on Systems, Man, and Cybernetics.* **1** (4): 364-378.

Jerne, N. K. (1974). Towards a Network Theory of the Immune System. *Annual Immunology* (*Institute Pasteur*), **125** C, 373-389.

Jones, T. (1995). Evolutionary Algorithms, Fitness Landscapes and Search. PhD dissertation, Albuquerque, NM: The University of New Mexico.

Koza, J. R. (1992). *Genetic Programming: On the Programming of Computers by Means of Natural Selection.* Cambridge, MA: The MIT Press.

Langdon, W. B. and Poli, R. (1998). Genetic Programming Bloat with Dynamic Fitness, In: Banzhaf, W., Poly, R., Schoenauer, M. and Fogarty, T. (Eds.), *EuroGP'98: First European Workshop on Genetic Programming*, LNCS-1391, Springer, Berlin, 97-112.

Mackey, M. C. and Glass, L. (1977). Oscillation and Chaos in Physiological Control Systems. *Science*, **197**: 287-289.

Merz,C.J. and Murphy,P.M. (1998). UCI Repository of machine learning databases `[www.ics.uci.edu/~mlearn/MLRepository.html]`, Irvine, CA: University of California, Dept. of Information and Computer Science.

Mulloy, B. S., Riolo, R. L. and Savit, R. S. (1996). Dynamics of Genetic Programming and Chaotic Time Series Prediction. In: Koza, J. R., Goldberg, D. E., Fogel, D. E. and Riolo, R. L. (Eds.), *Genetic Programming 1996: Proceedings of the First Annual Conference*, Cambridge, MA: The MIT Press, 166-174.

Nikolaev, N. and Slavov, V. (1998). Concepts of Inductive Genetic Programming, In: Banzhaf, W., Poly, R., Schoenauer, M. and Fogarty, T. (Eds.), *EuroGP'98: First European Workshop on Genetic Programming*, LNCS-1391, Springer, Berlin, 49-59.

Oakley, H. (1994). Two Scientific Applications of Genetic Programming: Stack Filters and Non-linear Equation Fitting to Chaotic Data. In: Kinnear, K. (Ed.), *Advances in Genetic Programming*, Cambridge, MA: The MIT Press, 369-389.

Perelson, A. S. (1989). Immune Network Theory. *Immunological Reviews*, **110**, 5-36.

Press, W. H., Flannery, B. P., Teukolsky, S. A. and Vetterling, W. T. (1992). *Numerical Recipes in C: The Art of Scientific Computing*, 2nd ed., Cambridge, England: Cambridge University Press.

Slavov, V. and Nikolaev, N. (1998). Immune Network Dynamics for Inductive Problem Solving. In: Eiben, A. E., Bäeck, T., Schoenauer, M. and Schwefel, H. P. (Eds.), *Parallel Problem Solving from Nature-PPSN V: 5th International Conference*, LNCS 1498, Springer: Berlin, 712-721.

Smith, R. E., Forrest, S. and Perelson, A. (1993). Searching for Diverse, Cooperative Populations with Genetic Algorithms. *Evolutionary Computation*, **1**(2): 127-149.

Zhang, B. -T., Ohm, P. and Mühlenbein, H. (1997). Evolutionary Induction of Sparse Neural Trees. *Evolutionary Computation*, **5**(2): 213-236.