Raport Badawczy
Research Report

RB/23/2001

# Evolutionary algorithm to derive classification rules from sets of examples

Jarosław Stańczak, Grażyna Szkatuła

Instytut Badań Systemowych
Polska Akademia Nauk

Systems Research Institute
Polish Academy of Sciences

**POLSKA AKADEMIA NAUK**

**Instytut Badań Systemowych**

ul. Newelska 6

01-447 Warszawa

tel.: (+48) (22) 8373578

fax: (+48) (22) 8372772

Pracę zgłosili: Jarosław Stańczak,
Grażyna Szkatuła

Warszawa 2001

# EVOLUTIONARY ALGORITHM TO DERIVE CLASSIFICATION RULES FROM SETS OF EXAMPLES

**Jarosław Stańczak** **Grażyna Szkatuła**

Systems Research Institute Polish Academy of Sciences

ul. Newelska 6, 01-447 Warsaw, Poland

## Abstract

In this paper we propose a method to derive classification rules that correctly describe all the examples belonging to a class and do not describe all the examples not belonging to this class. The method bases on an evolutionary algorithm with dedicated to that problem specialized operators and a method of valuing their behavior. The new concept of the proposed method is that every solution obtained from the algorithm (every member of the population in the evolutionary algorithm) contains rules which describe all classes of the training data. So it is a complete solution that covers all the examples presented to the algorithm. The results are very encouraging.

**Keywords:** Learning from examples, evolutionary algorithm (EA), genetic algorithm, heuristic operators, adaptive evolutionary algorithm.

## 1 Introduction

Machine learning from examples is a process of inferring a classification rule of a class from descriptions of some individual elements of the class called *positive examples*, with some elements from outside of the class, called *negative examples,* which are used for narrowing the solution space.

In practice, due to imperfect data and other elements of the process, the requirements to be satisfied by learning procedures are:

- *a completeness*, i.e. that the classification rule must correctly describe *all* the positive examples;
- *a consistency*, i.e. that the classification rule must describe *none* of the negative examples;
- *convergence*, i.e. the classification rule must be derived in a *finite* number of steps;
- the classification rule of *minimal "length"* is to be found, e.g. with the minimum number of attributes (or, more generally being "simple").

The sense of the first three is quite natural, and the sense of the fourth reflects an obvious fact that long rules are not "legible" to the humans, hence their practical usefulness may be limited.

Examples are described [cf. Michalski, 1983] by a set of $K$ "attribute - value" pairs written as $e = \bigwedge_{j=1}^{K} [a_j \# v_j]$, where $a_j$ denotes attribute $j$ with value $v_j$ and # is a relation exemplified by $=, <, >, \approx, \geq$, etc.

For instance, if the attributes are: *height, color_of_hair, color_of_eyes*, than the concept "look of a one men" may be described by

[*height* = "high"] $\wedge$ [*color_of_hair* = "blond"] $\wedge$ [*color_of_eyes* = "blue"].

We propose here a procedure based on the specialized evolutionary algorithm which derives classification rules directly from training data. The requirements[1] of the problem, denoted above, lead to the idea that the whole set of rules for all classes of data can be treated as a solution of the evolutionary algorithm, than can extract them from the set of training data. The simplest way to check the correctness or not of that idea is to apply the algorithm, which can be easily adopted to solve the problem. During conducted tests it

---

[1] There is a problem with a convergence of EA, which theoretically requires an infinite number of iterations, but in practical cases the number of them is not so high.

turned out that word „easily" is not proper in that situation, but obtained results are very promisive and proved that it was good idea to prepare more sophisticated, specialized evolutionary algorithm for that problem. Our modifications to the evolutionary algorithm included:

- specialized data structure with dynamically allocated number of rules;
- specialized set of genetic operators with 10 elements;
- a method of valuing and choosing operators for modification of solutions;
- controlled method of selection individuals to the next generation.

Fitness function is also a specialized element of used EA, but it is rather common fact that it depends on the solved problem.

The paper is organized as follows. In Section 2 the inductive learning problem is represented. In Section 3 the EA algorithm is described. We will test the method proposed in this paper on the thyroid cancer problem. In Section 4 computation results are given.

## 2 Problem formulation of inductive learning from examples

Suppose that we have a finite set of examples $U$ and a finite set of attributes $A = \{a_1, ..., a_K\}$. $V_{a_j} = \{v_{i_1}^{a_j}, ..., v_{i_j}^{a_j}\}$ is a domain of the attribute $a_j$, $j = 1, ..., K$, $V = \bigcup_{j=1,...,K} V_{a_j}$. $f: U \times A \to V$ is a total function such that $f(e, a_j) \in V_{a_j}$ for $\forall a_j \in A$, $\forall e \in U$, called an informational function.

Each example $e \in U$ is described by $K$ attributes, $A = \{a_1, ..., a_K\}$ and is represented by

$$e = \bigwedge_{j=1}^{K} [a_j = v_i^{a_j}] \tag{2.1}$$

where $v_i^{a_j} = f(e,a_j)$, $v_i^{a_j} \in V_{a_j}$, denotes the $j$-th attribute $a_j$ taking on a value $v_i^{a_j}$ for example $e$. An example $e$ in (2.1) is composed of $K$ "attribute-value" pairs (*selectors*), denoted $s_j = [a_j = v_i^{a_j}]$. Conjunction of $l \leq K$ "attribute-value" pairs, i.e.

$$C^l = \bigwedge_{j \in I} s_j = \bigwedge_{j \in I} [a_j = v_i^{a_j}] = [a_{j_1} = v_i^{a_{j_1}}] \wedge ... \wedge [a_{j_l} = v_i^{a_{j_l}}] \tag{2.2}$$

where $I \subseteq \{1,...,K\}$, $card(I) = l$ is called a *complex*.

Suppose now that we have example $e$ [cf. (2.1)] and we consider a complex [cf. (2.2)] $C^l = [a_{j_1} = v_i^{a_{j_1}}] \wedge ... \wedge [a_{j_l} = v_i^{a_{j_l}}]$ that corresponds to the set of indices $I = \{j_1,...,j_l\} \subseteq \{1,...,K\}$; the set of indices $\{j_1,...,j_l\}$ is clearly equivalent to a vector $x = [x_j]^T$, $j = 1,...,K$, such that $x_j = 1$ if a selector $s_j = [a_j = v_i^{a_j}]$ occurs in the complex $C^l$, and 0 otherwise.

For instance, for $K$ = *3* and example $e$ = [height = "high"] $\wedge$ [color_of_hair = "blond"] $\wedge$ [color_of_eyes = "blue"], the vector $[0,1,0]^T$ is equivalent to the complex [color_of_hair = "blond"]; the complex [height = "high"] is equivalent to the vector $[1,0,0]^T$.

A complex $C^l$ *covers* an example $e$ if all the conditions on attributes given as selectors are covered by (equal to) the values of the respective attributes in the example, i.e. $f(C^l, a_j) = f(e, a_j)$, $\forall j \in I$. For instance, for $K$=*3* the complex [$a_1$ = "woman"] $\wedge$ [$a_3$ = "35 years"] covers the example [$a_1$ = "woman"] $\wedge$ [$a_2$ = "married"] $\wedge$ [$a_3$ = "35 years"] but does not cover the example [$a_1$ = "man"] $\wedge$ [$a_2$ = "married"] $\wedge$ [$a_3$ = "35 years"].

We assume that $a_d$ is a decision attribute, $V_{a_d} = \{v_{i_1}^{a_d}, ..., v_{i_d}^{a_d}\}$ is a domain of $a_d$ and we have a set of attributes $\{a_1, a_2, ..., a_K\} \cup \{a_d\}$. Let us observe that the attribute $a_d$ determines a partition $\{Y_{v_{i_1}^{a_d}}, Y_{v_{i_2}^{a_d}}, ..., Y_{v_{i_d}^{a_d}}\}$ of the set $U$, where

$Y_{v_{i_t}^{a_d}} = \{e \in U: f(e, a_d) = v_{i_t}^{a_d}\}$, $v_{i_t}^{a_d} \in V_{a_d}$ for $t = 1,...d$, and $Y_{v_{i_1}^{a_d}} \cup ... \cup Y_{v_{i_d}^{a_d}} = U$, $Y_{v_{i_t}^{a_d}} \cap Y_{v_{j_t}^{a_d}} = \varnothing$ for $i \neq j$. The set $Y_{v_{i_t}^{a_d}}$ is called the *t*-th *decision class.*

Let us class $Y_{v_{i_t}^{a_d}}$ for $v_{i_t}^{a_d} \in V_{a_d}$. Suppose that we have a set of *positive examples*:

$$S_P(Y_{v_{i_t}^{a_d}}) = \{e \in U: f(e, a_d) = v_{i_t}^{a_d}\} \tag{2.3}$$

and a set of *negative examples*:

$$S_N(Y_{v_{i_t}^{a_d}}) = \{e \in U: f(e, a_d) \neq v_{i_t}^{a_d} \text{ and } \forall e' \in S_P(Y_{v_{i_t}^{a_d}}) \ \exists a_j \in P, f(e, a_j) \neq f(e', a_j)\} \tag{2.4}$$

Thus $S_P(Y_{v_{i_t}^{a_d}}) \cap S_N(Y_{v_{i_t}^{a_d}}) = \varnothing$ and $S_P(Y_{v_{i_t}^{a_d}}) \neq \varnothing$, $S_N(Y_{v_{i_t}^{a_d}}) \neq \varnothing$, by assumption.

An implication IF $C^l$ THEN $[a_d = v_{i_t}^{a_d}]$ is called a *"elementary rule"* for the class $Y_{v_{i_t}^{a_d}}$, where $C^l$ is description of example in terms of conditions attributes $a_j$, $j \in I$ and this example belongs to the class $Y_{v_{i_t}^{a_d}}$.

In this paper we consider the classification rules to be the disjunction (via "$\cup$") of "elementary rules" consisting of complexes of type (2.2), i.e.

$$\text{IF } C^{l_1} \cup ... \cup C^{l_L} = \text{THEN } [a_d = v_{i_t}^{a_d}] \tag{2.5}$$

where $I_1, ..., I_L \subseteq \{1, ..., K\}$, $C^{l_l} = \bigwedge_{j \in I_l} [a_j = v_i^{a_j}]$, $l = 1,...,L$ and "$\cup$" corresponds to the connective "or".

## 3. Evolutionary algorithm in the problem of discovering classification rules

In spite of its universality, the evolutionary algorithm should be adjusted to solve the raised problem in order to get best results quite fast. Unchanged remains only the core idea of its work: thanks to small random changes in genotypes of population members (mutation), the recombination of genes and the selection of the best individuals, the

population develops towards better values of the problem's goal function. It is shown in Fig 3 .1. The adjustment of the genetic algorithm to the solved problem requires a proper encoding[2] of solutions and an invention of specialized genetic operators for that problem and accepted data structure.

1. Random initialization of the population of solutions.
2. Reproduction and modification of solutions using genetic operators.
3. Valuation of the obtained solutions.
4. Selection of individuals for the next generation.
5. If a stop condition not satisfied, go to 2.

*Fig. 3. 1.The evolutionary algorithm.*

### 3.1 A member of the population of solutions data structure description

In the case of decision rules, the whole information about the actual solution is stored as a set of rules (with possible variable number of them), which describe all classes of data in the solved problem. So every member of the population has its own set of rules, which describe, as good as possible, the training data (Fig. 3. 2).

The rule (Fig. 3. 3) consists of a sequence of (integer) values for its decisive variables (value 0 means „don't care" in our approach) and its conclusion (which is a number of one of the class of the labeled training data). These numbers are treated as symbols rather than real values and it is important for the algorithm to know how many symbols it is possible to use for every decisive variable of a rule. Also some additional information about the rule is bounded with it: a number of well classified data, a number of wrong classifications and a list of numbers of wrong classified data (this information is required by some genetic operators).

---

[2]Applying a binary coding, which was not long ago treated as a base of evolutionary algorithms, is not very useful for this problem and nowadays is used only when it is easy and helpful to apply it.

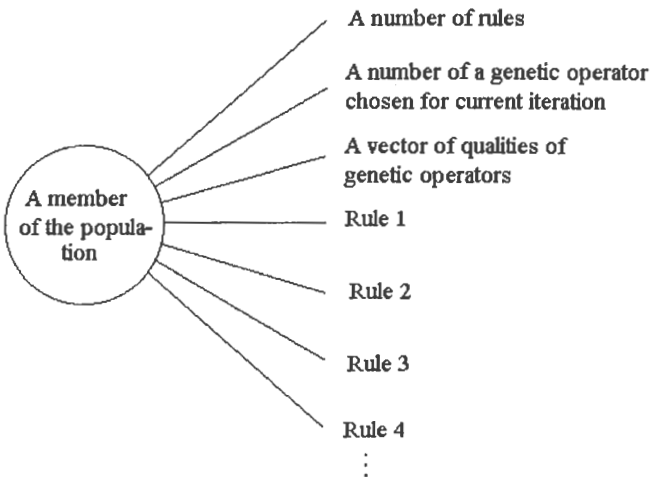



*Fig. 3. 2. A member of the population of solutions.*

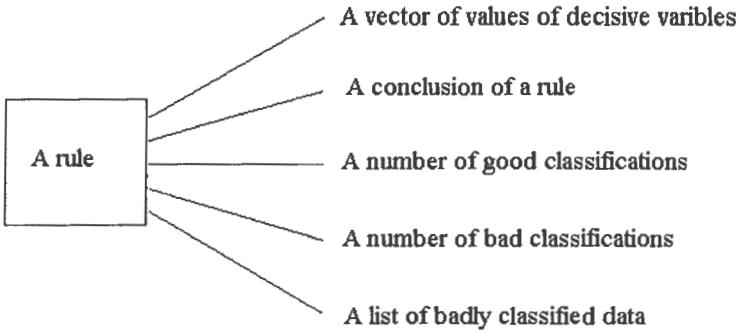



*Fig. 3. 3. A rule.*

Beside it, the member of the population contains several more data including: the number of rules (some genetic operators modify the number of rules), a vector of real numbers, which describe its knowledge about genetic operators and a number of the chosen operator for current iteration (the meaning of those values will be described later in this chapter).

### 3.2 Fitness function

The problem's quality function is closely connected with the fitness function, which values the members of the population - solutions of the problem. In the problem of discovering decisive rules, the fitness function is rather an algorithm than a mathematic formula (3.1).

$$F_k = \sum_{i=1}^{i=n\_ru_k} \left( \sum_{j=1}^{j=n\_da} c_{ij} \right) - \alpha \cdot n\_ru_k - \beta \cdot n\_nz_k \qquad (3.1)$$

where:

$F_k$ - a fitness function of the solution $k$;

$n\_ru_k$ - number of rules in the currently valued solution $k$;

$n\_da$ - number of the training data in the solved problem;

$n\_nz_k$ - number of non-zero values of decisive variables from all rules of the solution $k$;

$\alpha$, $\beta$ - small coefficients of proportionality;

$c_{ij}$ - a value of the rule $i$ from the datum $j$, obtained from a simple algorithm:

```
if (attributes_of_datum[j] match[3] rule[i])
{
   if(conclusion_of_rule[i] ==label_of_datum[j])
   {
      if (flag[j]==0) {c_ij=1; flag[j]=1;} else c_ij=0;
   }
   else c_ij=-1;
}
else c_ij=0;
```

For every rule of the valued solution the whole set of labeled training data is presented. If a datum obeys the rule, the rule gets one point unless that datum has been

[3] „Match" means that values of attributes are the same or 0.

fixed by earlier valued rule[4], in that case it gets no prize. If the rule says nothing about the datum, its score is unchanged. If it fails it loses one point, regardless the flag of the actually checked datum. Additionally rules are valued in random order (not according to their numbers as it is shown on Fig. 3. 2) to avoid situations, where rules which are covered by others exist only because they are earlier on the list of rules of the valued solution.

The fitness function for the solution is a sum of points got by its rules minus a small value proportional to the number of rules and the number of non-zero values of decisive variables in every rule. Parameters $\alpha$ and $\beta$ have been tuned „manually" but their values depend rather on the used evolutionary algorithm (selection method) than parameters of solved problem. In conducted tests, results for $\alpha=0.01$, $\beta=0.001$ and $\alpha=0.001$, $\beta=0.0001$ were similar and accepted values were valid for all tested problems. Also applying only parameter $\alpha$ or only $\beta$ gave worse results.

The main aim of the evolutionary algorithm is to maximize the formula (3. 1). Better solutions cover more training data and are shorter and simpler than worse. Ideally the evolutionary algorithm should produce the simplest solutions with properly classified all training data. For such solutions their fitness function should be very close to the size of the training data set. This is why such an „artificial" manner of valuing rules is used - it is easy to determine whatever all training data are properly classified.

### 3.3 Specialized operators

The described data structure requires specialized genetic operators, which modify the

---

[4] Every datum has a flag which shows if it was correctly classified by some rule or not. Only the first correct classification sets the flag and the rule responsible for it gets one point. This model prevents the situation, where the fitness function increases without any limits due to multiple classification of the same data.

population of solutions. Simple random operators are easy to think out, and similar to the widely used:

- mutation - random change one value to another from the set of possible values (it works on conditions an conclusions);
- crossover - an exchange of randomly found rules between individuals.

Also other „blind" but problem specific operators can be easily developed:

- adding a new rule with randomly chosen values of parameters;
- deleting a randomly chosen rule;
- mutation II - randomly chooses a condition an inserts 0 instead of its previous value (often leads to simpler rules).

Simulations of the evolutionary algorithm with „blind" operators proved that some more sophisticated, enriched with some heuristics should be used. Random operators have big difficulties to find a set of rules that covers all the training data. That's why a list of heuristic operators was worked out and successfully tested:

- adding an unclassified data as a new rule;
- deleting rules with no properly classified data or with long lists of wrong classifications (simplifies obtained solutions). An example:

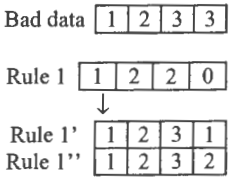
| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Rule 1 | 1 | 2 | 2 | 0 | $val_1=2$ | → | Rule 1 | 1 | 2 | 2 | 0 |
| Rule 2 | 0 | 2 | 3 | 1 | $val_2=0$ | → | | deleted | | | |
| Rule 3 | 2 | 0 | 1 | 1 | $val_3=5$ | → | Rule 2 | 2 | 0 | 1 | 1 |
| Rule 4 | 0 | 0 | 3 | 0 | $val_4=-1$ | → | | deleted | | | |

The symbol $val_i$ is a part of fitness function and may be described as $val_i = \sum_{j=1}^{j=n\_da} c_{ij}$ (all symbols like in formula 3.1).

- connecting similar rules (with the same conclusions and similar conditions) with adding 0 on positions which are different. An example:

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Rule 2 | 1 | 2 | 2 | 1 | → | Rule 2 | 0 | 2 | 0 | 1 |
| Rule 4 | 0 | 2 | 3 | 1 | → | | deleted | | | |

- breaking rules which have several bad classifications and many good (subtraction of bad data, basing on logical transformations of rules). An example:




  It is assumed that possible values of attribute 4 are: 0 (don't care), 1, 2, 3 and label of „Bad data" was different than conclusion of „Rule 1".

- aggressive breaking rules - an operator which consists of previously described version of breaking operator, a function which values rules (similar to fitness function) and the deletion of wrong or poor classificators. This group of operators works together in a loop until it has nothing to do - very powerful operator which speeds up the process of covering all testing data.

### 4.4 Evolutionary algorithm used to solve the problem

Using such a big number of genetic operators requires applying some method of sampling them in all iterations of the algorithm. In the used approach (based on works [Mulawka and Stańczak 1999], [Stańczak 1999] and [Stańczak 2000]) it is assumed that an operator that generates good results should have bigger probability and more frequently effect the population. But it is very likely that the operator, that is good for one individual, gives worse effects for another, for instance because of its location in the domain of possible solutions. So every individual may have its own preferences. Every individual has a vector of floating point numbers - $q$ (beside encoded solution). Each number corresponds to one genetic operation. It is a measure of quality of the genetic operator. The higher the number is, the higher is the probability of the operator. This relationship may be written as follows:

$$p_{ij}(t) = \frac{q_{ij}(t)}{\sum_{i=1}^{i=L(t)} q_{ij}(t)} \tag{3.2}$$

where:

$q_{ij}(t)$ - a quality coefficient of the *i-th* operation in the moment $t$ for *j-th* member of population;

$p_{ij}(t)$ - a probability of an appearance of the *i-th* operation in the moment $t$ for *j-th* member of population;

$L(t)$ - a number of genetic operators (may vary during genetic computations).

This ranking becomes a base to compute the probabilities of appearance and execution of genetic operators. This set of probabilities is also a base of experience of every individual and according to it, an operator is chosen in each epoch of the algorithm. Due to the gathered experience one can maximize chances of its offspring to survive. The quality factors are computed according to the formula (3.3):

$$q_{ij}(t+1) = \begin{cases} q_{0ij}(t) + x_{ij}(t+1) + \alpha_{ij}(t) \bullet q_{ij}(t) & \text{for } i = l \\ q_{ij}(t) & \text{for other } i \end{cases} \tag{3.3}$$

where:

$q_{ij}(t+1)$, $q_{ij}(t)$ - a quality of *i-th* operation for *j-th* individual in following generations;

$q_{0ij}(t)$ - a credit value, which can be modified during iterations;

$x_{ji}(t)$[5] - an improvement of the problem's quality function, obtained by *i-th* operation for *j-th* member of population. In the case of lack of the improvement equal to zero,

[5] In the work [Stańczak 1999] the improvement of the problem's goal function is normalized to value between 0 and 1, but in the solved problem it is unnecessary, because values of membership functions belong to this interval.

$x_{jt}(t)=Q_j(t)$ - $Q(t-1)$ (maximization), $Q_j(t)$ - solution of *j-th* individual, $Q(t-1)$ - the global best solution, found till the moment $t-1$;

$\alpha(t)$ - a coefficient of forgetting, $\alpha \in (0, 1)$, its value also can be adjusted during evolutionary process[6];

$l$ - an index of the operator chosen for modification of the particular solution.

The first element of the formula (3.3) - $q_0(t)$ plays a role of credit - a small value, which supports small level of $q_{ij}$ even if the operator does not give any advantages for a long term. Dropping this value to zero would eliminate corresponding to it operation for current individual and for its possible offspring. This fact is not profitable, because it is possible that they will work better on other stages of the evolution process. For exploring operators like mutation it is often necessary to let them work even without any visible improvements of the fitness function (as it can be noticed later from the presented simulation results).

The second addend is an improvement of the problem's quality function in the current generation or zero when no improvement is achieved.

The third part of the formula (3.3) remembers old achievements of an operator multiplied by forgetting factor $\alpha(t)$. It is responsible for balancing influence the quality factor of old and new improvements of the operator. Decreasing the value worked out by an operator is introduced to make the evolutionary algorithms more flexible. It should be noticed that some genetic operators may achieve good results in some phase of simulation, and then draw out their abilities, but others, probably better in next phases, would have small probabilities of appearance, so it would take a lot of time to change this situation. The effect of forgetting former achievement can overcome this problem. When operators

[6] The case with adjusted values of $\alpha_{ij}(t)$ and $q_{0ij}(t)$ is shown in the paper [Stanczak 2000].

don't change the global best solution for some time, the probabilities of operators become equal. After every generation only bounded with the chosen operator value $q_{ij}(t+1)$ is updated, the others remain unchanged. Only one operator is executed in one generation for one individual, so there is no reason to change coordinates corresponding to the other, not selected operators.

### 3 5. Controlled selection

The applied selection method consists of two methods with different properties: a histogram selection (increases the diversity of the population) and a deterministic roulette (strongly promotes best individuals) [Stanczak 1999], which are selected in random during the execution of the algorithm. The probability of executing of the selection method is obtained from the method shown on Fig. 4. 4.

1. If $3*\sigma(F) < \max(F_{mean} - F_{min}, F_{max} - F_{mean})$ then $p_{his} = p_{his} - a*p_{det}$;
2. If $0,5*\sigma(F) > \max(F_{mean} - F_{min}, F_{max} - F_{mean})$ then $p_{his} = p_{his}*(1 + a)$;
3. If $0,5*\sigma(F) \leq \max(F_{mean} - F_{min}, F_{max} - F_{mean}) \leq 3*s(F)$ then $p_{his} = p_{his}*(0,5-p_{det})*a$;

*Fig. 4. 4. The algorithm of automatic tuning probabilities of the mixed selection elements*

The symbols used in Fig. 4.4 are: $p_{his}$ - probability of histogram selection appearance, $p_{det}$ - probability of deterministic roulette, $F_{mean}$, $F_{min}$, $F_{max}$ - average, minimal and maximal values of fitness function in the population.

If individuals in the population are described by too small standard deviation of the fitness function ($\sigma(F)$) with respect to the extent of this function ($max(F_{mean}- F_{min}, F_{max}- F_{mean})$), then it is desirable to increase the probability of appearance of the histogram selection. On the contrary the probability of the deterministic roulette selection can be increased. As far as parameters of the population are located in some range, considered as profitable we may keep approximately the same probabilities of appearance for both methods of selection. It is important that always $p_{his}+p_{det}=1$- it means that some method of

selection must be executed.

## 4. Application of the EA algorithm to solve a thyroid cancer problem

The medical data set published by Nakache and Asselian have been collected on patients with thyroid cancer at Hospital Ambroise Paré, from 1960 to 1980. They concern 281 patients, all submitted to a surgical treatment. The time of analysis has been fixed in July 1981. At that time, the patient is dead or alive. The survival time is then fully known for those patients who have died before the time of analysis. For those who are still alive, we only know the inferior limit of their survival time. Patients who survive may be completely cured and their survival pattern might have no relationship to those who died. Each patient was described by the following 12 attributes in a discrete coding:

$a_1$ : *sex, {male, female}*

$a_2$ : *age, {<40, 40-60, 60-70, >70}*

$a_3$ : *histology, {well differentiated, poorly differentiated}*

$a_4$ : *metastasis, {yes, no}*

$a_5$ : *enlargement, {uni-lobe, uni-lobe+isthm, all the thyroid}*

$a_6$ : *clinical lymph nodes, {yes, no}*

$a_7$ : *clinical aspect, {unique nodule, multi nodules, important enlargement}*

$a_8$ : *pathological lymph nodes, {yes, no}*

$a_9$ : *compressive syndromes, {yes, no}*

$a_{10}$ : *invasion, {no, small, average, large}*

$a_{11}$ : *survival time, {in months}, length in month of survival time from the entrance in the study (between 1960 and 1980) to the time of analysis,*

$a_{12}$ : *survival, {survivor, non survivor at time of analysis}.*

Two of the 12 attributes are important: survival time (in month) of a patient at the time of analysis, and survival or non survival at the time of analysis. The aim of such a study is to identify prognostic elements of disease evolution and to define a prognostic rule for a new case coming from the same population and being in the same conditions. The class attribute is the survival time. The patients (training examples) have been divided into two classes:

*class 1*: the patients will be alive over 7 years,

*class 2*: the patient will be dead during 7 years.

The first class includes patients with the survival time over 7 years. The patients with the survival time below 7 years belong to the second class. The problem of learning from examples is formulated as to find two classification rules [Nakache 1983]:

IF $R_W^{1*}$ THEN [*survival time = over 7 years*]

IF $R_W^{2*}$ THEN [*survival time = below 7 years*]

using all examples as patterns. The $R_W^{1*}$ and $R_W^{2*}$ are specified by "elementary" conditions depending on the attributes. We assume, that the classification rules for elements belonging to class $l$, $l = 1,2$, must correctly describe all the examples. The measure of classification accuracy $A_{learning}$ is the ratio of examples correctly classified to the class 1 (or class 2) to the total number of examples, in percent. The ratio of correct classification decisions to the total number of decisions made was taken as the measure of classification accuracy, in percentage.

The rough set theory, introduced by Z. Pawlak is here chosen as a tool to selection of a set of the most important attributes. A *reduct* of attributes is the minimal subset of attributes ensuring the same quality of the classification as the complete set of attributes; a *core* is an intersection of all reducts in the information system.

All classes are definable, i.e. precisely characterized by ten attributes. First, we looked for the core of attributes, and then we computed all reducts. The core is composed of three attributes: $a_2$: *age*, $a_4$: *metastasis* and $a_5$: *enlargement*. We found seven following reducts: $\{a_1, a_2, a_3, a_4, a_5, a_{10}\}$, $\{a_1, a_2, a_4, a_5, a_7, a_{10}\}$, $\{a_1, a_2, a_4, a_5, a_8, a_{10}\}$, $\{a_2, a_3, a_4, a_5, a_9, a_{10}\}$, $\{a_2, a_3, a_4, a_5, a_8, a_9\}$, $\{a_2, a_4, a_5, a_8, a_9, a_{10}\}$, $\{a_2, a_4, a_5, a_7, a_9, a_{10}\}$. Let us notice that attribute $a_6$ does not occur in any of reducts. Then we performed the analysis of frequency of the attributes in all reduct. The distribution of the attributes is presented in Table 4.1.

| Attribute | $a_1$ | $a_2$ | $a_3$ | $a_4$ | $a_5$ | $a_6$ | $a_7$ | $a_8$ | $a_9$ | $a_{10}$ |
|---|---|---|---|---|---|---|---|---|---|---|
| Number of reducts | 3 | 7 | 3 | 7 | 7 | 0 | 2 | 3 | 4 | 6 |

*Tab. 4.1. The frequent attributes in chosen reducts.*

The attributes $a_2$, $a_4$, $a_5$, $a_{10}$ occur in all reducts more frequent than other attributes, therefore, they are the most important attributes in system.

The EA method [Stańczak, 2000], method IP2_GRE with elements of greedy algorithm [Kacprzyk and Szkatuła, 1999, 2000] and method IP2_RS with elements of the rough set theory were applied to the data described above. We assume that the classification rules must correctly describe all training examples, $A_{learning}$ = 100 %, by assumption. The results of applying this methods to medical data are presented and described below.

| Algorithm | Number of iterations | Number of selectors in rule |
|---|---|---|
| IP2_RS | 4 | 12 |
| IP2_GRE | 3 | 11 |
| EA | 3 | 9 |

*Tab. 4.2. Some parameters describing the process of finding a classification rule for the class 1.*

| Algorithm | Number of iterations | Number of selectors in rule |
|---|---|---|
| IP2_RS | 5 | 9 |
| IP2_GRE | 8 | 15 |
| EA | 5 | 9 |

*Tab. 4.3. Some parameters describing the process of finding a classification rule for the class 2*

As can be seen, the shortest classification rules were obtained by using the method EA. A classification rule for the first and the second class (by using the EA method) are presented below. Each rule is additionally described by the number of covered objects.

IF [*metastasis* = *no*] ∧ [*enlargement* = *uni-lobe*] ∧ [*invasion* = *no*] THEN
[*survival time* = *over 7 years*] : 10;
IF [*age*<*40*] ∧ [*compressive syndrom* = *no*] ∧ [*invasion* = *no*] THEN
[*survival time* = *over 7 years*] : 3;
IF [*metastasis* = *no*] ∧ [*pathological lymph nodes*=*no*] ∧ [*compressive syndrom* = *no*]
THEN [*survival time* = *over 7 years*] : 35;

IF [*invasion* = *large*] THEN [*survival time* = *below 7 years*] : 5;
IF [*metastasis* = yes] THEN [*survival time* = *below 7 years*] : 11;
IF [*enlargement* = *uni-lobe*+*isthm*] ∧ [*clinical aspect* = *multi nodules*] THEN
[*survival time* = *below 7 years*] : 4;
IF [*age* ∈ *40-60*] ∧ [*enlargement* = *uni-lobe*] ∧ [*invasion* = *average*] THEN
[*survival time* = *over 7 years*] : 1.

## 5. Concluding Remarks

We have presented an procedure EA to derive classification rules from sets of positive and negative examples. The computational results are very encouraging and give a strong impulse for further investigations on a hybrid method which can deal with imperfect data or with not precisely described classes of data.

## References

**Kacprzyk J. and Szkatula G. (1996)**: *An algorithm for learning from erroneous and*

*incorrigible examples*, *Int. J. of Intelligent Syst.* 11, pp. 565 - 582.

**Kacprzyk J. and Szkatuła G. (1997)**: *Deriving IF-THEN rules for intelligent decision support via inductive learning*, in N.Kasabov et al. (eds.): Progress in Connectionist-Based Information Systems (Proceedings of ICONIP'97, ANZIIS'97 and ANNES'97 Conference, Dunedin, New Zealand), Springer, Singapore, vol. 2, pp. 818 - 821.

**Kacprzyk J. and Szkatuła G. (1999)**: *An inductive learning algorithm with a preanalysis of data*, International Journal of Knowledge - Based Intelligent Engineering Systems, vol. 3, pp. 135-146.,

**Michalski R. S. (1983)**: *A theory and methodology of inductive learning*, in: R. Michalski, J. Carbonell and T.M. Mitchell (Eds.), Machine Learning. Tioga Press.

**Mulawka J. and Stańczak J. (1999)**: *Genetic Algorithms with Adaptive Probabilities of Operators Selection*, Proceedings of ICCIMA'99, New Delhi, India, 1999.

**Nakache J. P., Asselain B. (1983)**: Medical data set proposed for the workshop on data analysis. EIASM Workshop, April 1983.

**Stańczak J. (1999)**: *Rozwój koncepcji i algorytmów dla samodoskonalących się systemów ewolucyjnych*, Ph. D. thesis, Politechnika Warszawska, 1999.

**Stańczak J. (2000)**: *Algorytm ewolucyjny z populacją "inteligentnych" osobników*, Materiały IV Krajowej Konferencji Algorytmy Ewolucyjne i Optymalizacja Globalna, Lądek Zdrój, 2000;

**Szkatuła G. (1996)**: *Machine learning from examples under errors in data*, Ph.D. thesis, SRI PAS Warsaw, Poland.