*PERIODICA POLYTECHNICA SER. EL. ENG. VOL. 44, NO. 3–4, PP. 259–270 (2000)*

# STOCHASTIC METHODS: IMPLEMENTATION OF IMPROVED GENETIC ALGORITHM FOR SHAPE RECOGNITION PURPOSES

Imre KÓNYA

Department of Control Engineering and Information Technology
Budapest University of Technology and Economics
H–1521 Budapest, Hungary
Tel.: 36-1-463-2214
e-mail: konya@seeger.iit.bme.hu

Received: April 14, 2000

## Abstract

Considering parameter optimization tasks, a fundamental advantage of the genetic algorithm lies in its robustness, i.e. that it is capable of providing a solution of good quality even when the error surface is unknown or of extreme shape. Its disadvantage is its demand for large system memory and computational capacity, as well as the fact that because of its stochastic basis it is not to be used in most real-time applications. However, if the adjustment or training is rarely to be made, or the comparison of different parameter vectors can be performed rapidly, it is without doubt advantageous to apply.

The author of this paper has examined and modified the standard methods to improve their performance. One of these was the implementation of the Special Linear Combination crossover method. Another one was the Maximum Fitness Guided Migration method. Both methods will be discussed in detail in this paper. The author has implemented the advanced methods in a computer program, which was then tested on a shape recognition task.

*Keywords:* genetic, algorithm, crossover, migration, SLC, MFGM.

## 1. Introduction

A great part of tasks in practical engineering (*including problems in the field of image processing and shape recognition*) can be discussed mathematically i.e. as a transformation of inputs to outputs. The transformation can be characterized by numbers or rather by a parameter vector (*whose components are these numbers*), which has to be determined in a way that the transformation applied to each input combination should give correct output as a result.

This is the classical task of parameter identification, for which many numeric methods offer some kind of solution. These methods can be interpreted as guided search algorithms, which conduct a search in the parameters' space. In the course of this search, the goal is to find the optimal parameter vector. The transformation defined by this vector is the one that, when applied to different input combinations, results in a response which is the closest possible one to the expected response. In other words: the ‘distance’ (*e.g. Euclidean*) between the expected and the resulted responses, and the sum of these distances, thus the error must be minimal [1].

Conventional methods [2] tend to be applied less often, while the applications of stochastic search algorithms and among them of the classical genetic algorithm and its advanced forms gain ground. The genetic algorithm works on the model of the biological evolution. Unlike conventional methods that create the new solution on the basis of the previous one, it operates on a set of solutions, from which the best ones will be chosen, and combined with each other to a new, hopefully better solution. The classical genetic algorithm has a quality that might be surprising at first, notably that it does not assume any special condition of the search-space, thus it is able to approximate a function even when the error surface is not differentiable or the error-surface is ruptured. This is so, because the algorithm does not need the gradient of the error surface (*only the rate of the errors when comparing the errors that belong to different solutions*). Some enhancements of the classical genetic algorithm result in solutions faster and more robust than the original one without limiting the area of applicability, others, although introducing limitations, provide an extra fast and higher quality convergence for some problem types.

## 2. The Biological Evolution and the Genetic Algorithm

The basic assumptions of the biological genetic model are as follows: The individuals of a species are living in a reproductional community, called population. In the course of the struggle for life the more viable individual survives, reaches its maturity and will be capable of self-reproduction. The described process is referred to as natural selection. In the process of reproduction, the parental gene sets combine and form an offspring that has properties similar to those of its parents. During this process some error might occur (*spontaneous mutation*), that can result in an individual having merely new properties.

The genetic algorithm’s term ‘parameter vector’ corresponds to the term ‘individual’, thus a ‘set of vectors’ refers to a ‘population’. Generation is referred to as a set of vectors at a given moment. The biological ‘viability’ is the genetic term of ‘fitness’, that is the complementer function of error. (*It is minimal when the error is maximal, and vice versa, the transformation is monoton.*) The genetic algorithm uses three operators while it creates the next generation of individuals. These are: selection, crossover and mutation. In the course of the classical selection, the best individuals of a population will be chosen for reproduction. The classical way of reproduction is the one-point bitwise crossover.

This is carried out in the following way: Couples will be formed from the chosen individuals (*parent vectors*), and the offspring vector is created by copying a randomly chosen number of bits from the first parent vector (*father*), the rest from the second parent vector (*mother*). After this, the operation of the classical mutation is performed at a small probability. This form of mutation means that a randomly chosen bit of the offspring is inverted. The next generation will be obtained from the previous one, in the way that the offspring vectors overwrite the memory area of the worst individuals. Repeating the steps described above, the individuals of each next generation will be better and better (see *Fig. 1*), thus the obtained parameter vectors give a solution of smaller error [3].

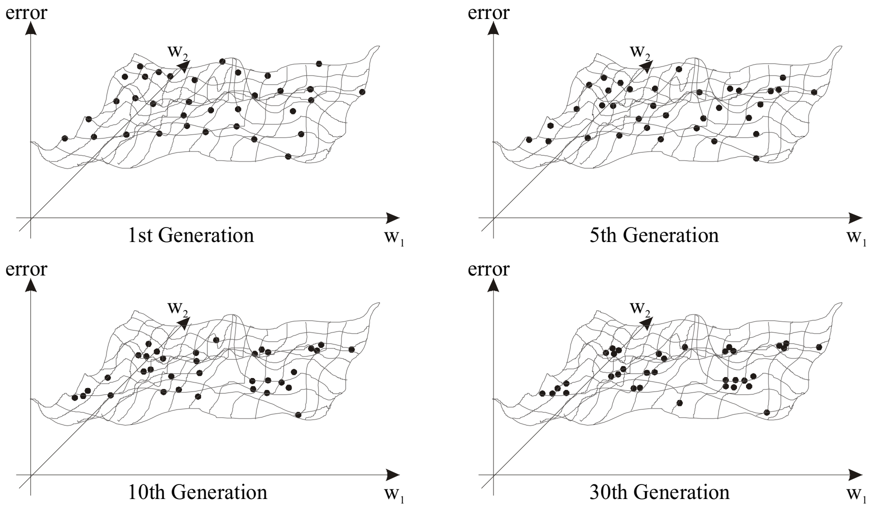


*Fig. 1.* The convergence of the parameter vectors (*2D case: there are only two components of* '$w$'). The vectors gather around the pits of the error surface.

## 3. The Implementation of Selection

The use of the classical selection results in a population of individuals too similar to each other. This similarity could go that far, that they would be the exact replication of the same individual, which means that the algorithm is stuck at a local minimum. Therefore, it is well- advised to give some chance to reproduce even for the individuals that do not perform well at the moment, but evolve in a promising direction [3]. So, the Fitness Proportional selection (see *Fig. 2*) can be used instead, which means that an individual will be chosen as a parent at a probability (see (1)) that is proportional to its fitness value.

$$P(i) = \text{fit}(i)/(\text{fit}(1) + \text{fit}(2) + \ldots + \text{fit}(n)), \tag{1}$$

where '$n$'=the number of the individuals of the population.

However, if there are high deviations to be found among the fitnesses of the individuals in a population, the chance of survival and to reproduce will be practically zero for the inferior individuals. So the next modification of the proportionate selection is expedient: The fitness of an individual is only used to establish a row of order among them. The larger the fitness value, the higher the position (noted by $\mathrm{pos}(i)$ in (2)) of the individual in the sequence. The probability of becoming a parent is proportional to the vector's position in the sequence (see *Fig.3*):

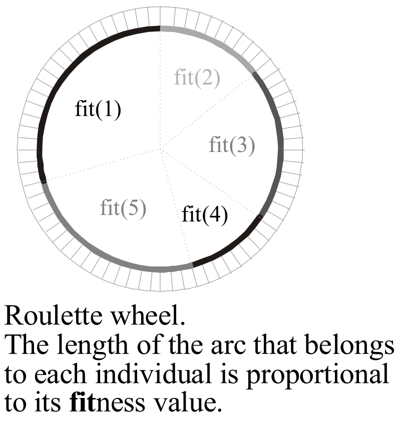

fit(1)
fit(2)
fit(3)
fit(4)
fit(5)
Roulette wheel.
The length of the arc that belongs to each individual is proportional to its **fit**ness value.

*Fig. 2.* Fitness proportional selection

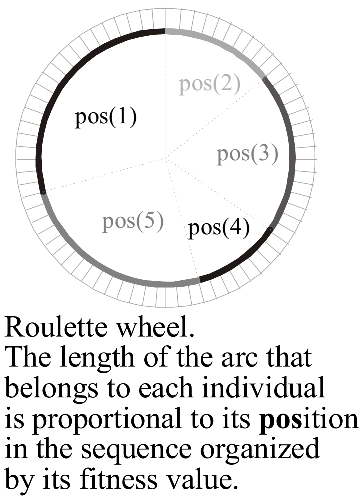

pos(1)
pos(2)
pos(3)
pos(4)
pos(5)
Roulette wheel.
The length of the arc that belongs to each individual is proportional to its **pos**ition in the sequence organized by its fitness value.

*Fig. 3.* Fitness organized sequence position proportional selection

$$P(i) = \mathrm{pos}(i)/((n+1)(n/2)). \tag{2}$$

It is a useful byproduct of this method that its application does not require the exact calculation of error for a parameter vector. It is sufficient if their comparison criterion is fulfilled. This means that the speed of the algorithm can be dramatically increased, because (*if we find a fast method for the comparison of the vectors*) it will not be necessary to calculate the results (*and the error*) to each of the input combinations of the training set for the two vectors to compare.

The selection methods discussed above cannot prevent the accidental disappearance of a population's best individual at 100%. This is because the offspring is not a copy of a parent; it is the combination of two parents. Therefore the best one (*or the best few but maximum a few percent*) of the individuals will be preserved as they are in the next generation, while of course they can be chosen as parents as well. This is the elitist strategy, which ensures that with each generation the fitness of the best individual increases, or does not change in the worst case.

## 4. The Implementation of Crossover

To perform the classical one-point bitwise crossover, we generate a $k(i)$ random number for each parent vector couple, in the range of $1 \ldots$ '$b-1$', where '$b$' is the bitlength of a parameter vector, and '$i$' is the index number of the couple. The 1st, 2nd, 3rd, $\ldots k(i)$th bit of the first parent will be copied to the offspring. The remaining bits of the offspring will be copied from the second parent. (*If some characteristics of the individual, represented by a vector, are defined mostly in the first part of the vector, it could be feared, that these characteristics will be overwhelmingly defined by parent A, since during any crossover the first few bits are always copied from parent A. This undesired effect can be neutralized, if the selection of parent A and parent B is random. In such a way any particular vector can be parent A as well as parent B*). The $k(i)=0$ and the $k(i)=b$ cases are excluded because it would not mean the testing of a new combination, and the survival of the best parameter vectors can be ensured by the elitist strategy.

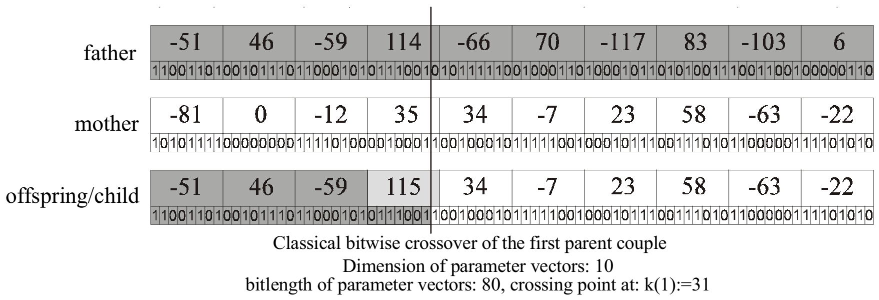


*Fig. 4.* One-point bitwise crossover, two-complement code 8-bit parameter vector components

**Remark**: Multiple-point bitwise crossover methods differ only from the method above in that the offspring's bits will consist of the concatenation of not just two bit-sequences (*one from the first parent vector (father), another from the second one (mother)*) but more than two alternatively originned sequences follow each other. The number of these sequences is one less than the number of the crossing-points. E.g.: when applying 3-point bitwise crossover on 50 bit long vectors, 3 numbers $(p, q, r)$ have to be chosen at random between 1 and 50, so that $p+q+r < 50$. The offspring's $1 \ldots p$ bits will be transferred bit by bit from the father's $1 \ldots p$ bits, the $p+1 \ldots p+q$ bits from the mother's $p+1 \ldots p+q$, then the $p+q+1 \ldots p+q+r$ bits again from the father's bits, and the rest of the bits of the offspring from the mother's $p+q+r+1 \ldots 50$ bits.

There are two possibilities: each component of a child vector will spring undividedly from the first or from the second vector, or there will be a component whose bits come partially from the first, partially from the second parent vector (see *Fig. 4*).

It can be proven that even when the component in question comes from two two-complement coded numbers of different sign or the two components are floating point numbers whose exponent is of different sign, the result will be likely far outside of the interval defined by the parent vector components. (*For example, the* $-1$ *and* $+1$ *two-byte two-complement code integer numbers can result in* $-32767$ *or in* $+32767$ *depending on where the first bit is taken from. The* $1 * 10e - 1$ *and the* $1 * 10e + 1$ *real type numbers when cut after the first bit of the exponent can even result in a value as high as* $10e + 38$).

So the problem of this method is, that it cannot fulfill an important criterion of the biological evolution model: 'the offspring is similar to its parents'. The problem was caused by the splitting of a parameter vector component, so in many genetic applications the position of cutting is restricted to the component-bounds. (*This method in this article will be referred to as Component Frame Restricted crossover method*). This means however, that not only the interval of the search space, but its resolution is dramatically restricted as well.

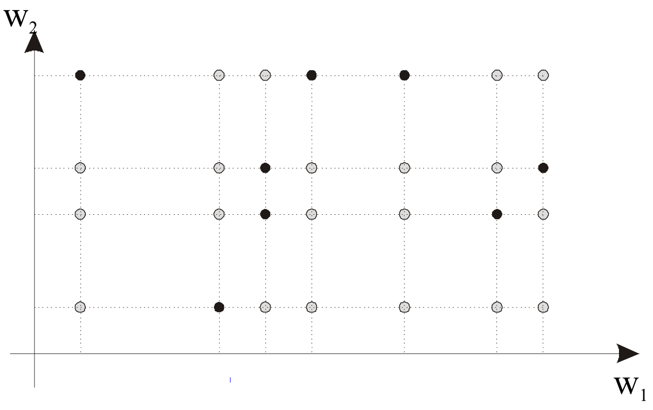


*Fig. 5.* $V$ is the number of all the dots in the figure
● represent the parameter of the first population. ○ represent all the other vectors that can be tried out when using only the selection and the Component Frame Restricted crossover method. 2-dimensional case, number of vectors: 8, $M(1) = 7$, $M(2) = 4$, $V = 28$

Let '$V$' (see (3)) be the number of different vectors that can be tried out when using only the selection and the crossover operator.

$$V = M(1) * M(2) * \ldots * M(s) \tag{3}$$

$M(j)$ represents the different numbers of all the parameter vectors' $j$-th component $j = 1 \ldots s$, where '$s$' is the dimension of the parameter vector.

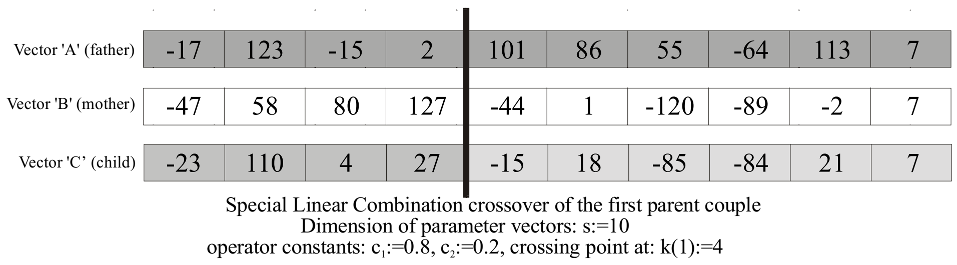
The value of '$V$' cannot increase without mutation, because only the mutation can adjust a vector component to a different value than the original set of values (see *Fig.* 5).

This restriction can be eliminated using another method, where the $j$-th component of the offspring vector comes from the Special Linear Combination (S.L.C.) of the $j$-th component of the parent vectors (see *Fig.* 6). According to this, $C(j)$ can be calculated by (4).

$$C(j) = c_1 * A(j) + c_2 * B(j) \tag{4}$$

In (4) $C(j)$ is the $j$-th component of the offspring vector, $A$ and $B$ mark the parent vectors, $c_1$ is a real number from the interval $0.5 \ldots 1$, and $c_2 = 1 - c_1$. When one-point crossover is applied, a $k(i)$ random number has to be drawn separately for each parent couple. $C(j)$ can be obtained by (4) when $j \leq k(i)$. When $j > k(i)$ the roles of $c_1$ and $c_2$ in (4) have to be exchanged.

**Remark**: Because the selection of parent $A$ and parent $B$ is random, any particular vector can be parent $A$ and parent $B$ as well, so the $0.5 \ldots 1$ interval for $c_1$ is sufficient, the use of the whole $0 \ldots 1$ interval is not necessary.



| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Vector 'A' (father) | -17 | 123 | -15 | 2 | 101 | 86 | 55 | -64 | 113 | 7 |
| Vector 'B' (mother) | -47 | 58 | 80 | 127 | -44 | 1 | -120 | -89 | -2 | 7 |
| Vector 'C' (child) | -23 | 110 | 4 | 27 | -15 | 18 | -85 | -84 | 21 | 7 |

Special Linear Combination crossover of the first parent couple
Dimension of parameter vectors: s:=10
operator constants: $c_1$:=0.8, $c_2$:=0.2, crossing point at: k(1):=4

*Fig. 6.* One-point Special Linear Combination crossover, two-complement code 8-bit parameter vector components

As a result of the S.L.C. crossover method, the offsprings $1 \ldots k(i)$ components will be more *similar to* (and not – if the Component Frame Restricted crossover method were used – *exactly the same as*) the components of parent $A$ and the $k(i) + 1 \ldots s$ components more like the ones of parent $B$. This way the S.L.C. crossover method can scan the search space much more flexibly, and it is much more capable of fine tuning the parameter vectors. However, it unjustifiedly prefers solutions that are located in the geometrical center of the '$s$'-dimensional rectangle defined by the '$s$'-component original parameter vectors, instead of their neigbourhood. Therefore, it might be expedient to take the value of $c_1$ (*or after the exchange the value of* $c_2$) from the $[1 - q, 1 + q]$ real interval, where $0 < q < 0.5$. As the genetic algorithm is usually started from the random population of original vectors, the crossover method mentioned above gives a rough scan of the parameter vectors' space at the beginning. Later, as more and more parameter vectors find a specific optimum pit of the error surface becoming this way similar to each other, the child vectors (*that spring from the S.L.C. crossover of the parent vectors*) will be geometrically close to the parents as well. This means that the search will be increasingly concentrated at the optimum pit, and the algorithm converges.

In general, it is also possible to determine each component of the offspring vector from the corresponding component of the parent vectors. The use of a more complicated function than the linear function increases the time consumed, so it is only recommended if it is suggested by special a priori information on the search space.

The one-point crossover methods mentioned above also have their multi-point versions.

## 5. The Realization of Mutation

In the case of the classical bitwise mutation this means the inversion of one bit, which can result in a dramatic change of the offspring vector. (*E.g. the change of one bit in the exponent of a real number*.) More gentle versions are the value oriented mutations, which means that the component of a vector should be slightly increased or decreased by adding to or subtracting from it a small positive number, or by multiplying it by a number close to $+1$. It is also possible to change the value of a component by applying a function to it. All of these can be done by using fixed or random numbers for the modification. When more drastic changes are needed to be made, it is possible to overwrite some components of the offspring vector by a random number. There could be a different frequency applied for mutations causing smaller changes and for others causing greater changes [3].

## 6. The Scanning of the Error Surface – Convergence

When there is one population of the individuals and the used genetic algorithm applies the Fitness Organized Sequence Position Proportional selection, the Special Linear Combination crossover and some kind of rarely occurring mutation, it can be seen that the parameter vectors concentrate more and more in the neighbourhood of the error surface's optimum pits (see *Fig. 1*). When in addition to this, elitist strategy is used, it could be hoped that the vectors of the elite will represent different local minima of the error surface.

However, this could only be expected if the shape and size of the pits on the error surface were similar, which is most often not the case. Thus, the vectors that somehow got close to a pit of steeper slope could quickly displace the representatives of the other local minima from the elite, and so significantly lower their chance of survival. Finally, every individual of the population will be focused around this local optimum. The biological analogy is well known, and the solution of the nature for the problem is the refreshing of blood (*i.e. the set of genes*) by an immigrant individual of another population. This idea led to the application of the multi-population genetic algorithms.

## 7. Multi-Population Genetic Algorithms

These algorithms work on the same basis as in the case of one population. The difference is that at start the parameter vectors are divided into disjoint sets, i.e. populations. The three basic genetic operators (*selection, crossover and mutation*) are separately executed on the individuals of each population. There is a separate elite in each population. In the course of the generations each population evolves in its own way, thus the individuals of different populations will have different good and bad qualities. Our goal is to unify all the good qualities in one individual.

To achieve this, sometimes (*at a small probability*) we move an individual from one population to another (*migration*). The good qualities of the immigrant will be hopefully passed toward the individuals of the acceptor population making them even more perfect [3]. The migration is often realized by the simple exchange of two individuals of different populations. This way, it is not to be feared, that the individuals completely disappear from a population that is chosen most of the time as a donor.

This method however involves hidden dangers. To explain this let us assume that there is a population called ‘*P*’, chosen as donor at the moment, and a population called ‘*Q*’, that is now the acceptor. Let us also assume that the fitness of the best individual in ‘*P*’ is better than the fitness of the best one in ‘*Q*’. In this case there is a chance of choosing an individual from ‘*P*’ whose fitness is better than the fitness of anyone in ‘*Q*’. This individual will most likely displace (*by its offsprings*) the earlier members of the elite in ‘*Q*’, so the population ‘*Q*’ will be transferred to the local minimum already scanned by population ‘*P*’ (see *Fig.7.a*). The phenomenon has its biological analogy as well: the mammals brought to Australia by man have rapidly gained living-space at the cost of the native species without changing genes with theirs. Because in general between two populations one is always better than the other, it is not recommended to realize the migration by a simple change of individuals.

The problem of rescanning mentioned above cannot be avoided at a safety factor of 100%, however, its chance can be significantly decreased by the use of the Maximum Fitness Guided Migration method. The M.F.G.M. method makes an individual migrate only, if the maximal fitness in the donor population is less than the maximal fitness in the acceptor population. This means that the immigrant cannot overwrite the acceptor’s elite, but it has a chance to pass its good features onto the members of the acceptor population (see *Fig.7.b*).

## 8. A Practical Application of the Genetic Algorithm

The advanced genetic algorithm was implemented in a computer program. To test its performance the software simulation of a real-life image recognition problem was used. The exact position of a rotating disc counter had to be detected.

To obtain the accuracy required the identification of the least significant digit

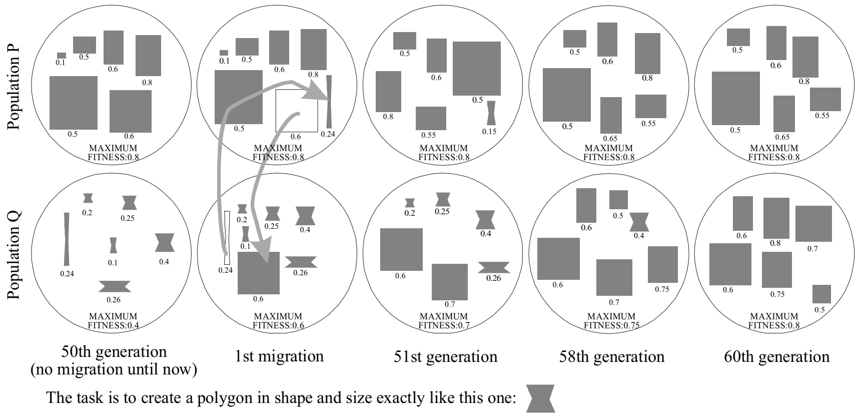


*Fig. 7.a.* The possible undesired result of the classical ‘migration by exchange’ method: The concave hexagonal shape of the original population $Q$ cannot be passed on to the individuals possessing the greater size quality of the original population $P$. This way, the suboptimal error-minimum pit already scanned by population $P$ will be rescanned by population $Q$ as well. Remark: The fitness of the exact match is one. If the target object cannot be fully covered, or if it is overcovered the fitness is smaller. The minimum fitness value is zero.

was not sufficient; four positions between had to be recognized as well (see *Fig.*8). There was a great deal of noise to be expected in the real images. The original $32 \times 64$ pixel 8-bit image was reduced to $8 \times 8$ pixel 16 bit one. This was the input of a neural network of 256 inputs, 2 layers ($500 + 50$ neurons) and 50 outputs [4]. This NN could be separated into 50 subnets of 256 inputs, 2 layers ($10 + 1$ neurons) and 1 output (see *Fig.* 9). There was a full connection between the 2 layers of a subnet, but there were no cross-subnet weights.

The 50 subnets were trained in 50 separate sessions. The training process was realized by a self-developed genetic-based computer program. One training session meant the adjustment of 2570 parameters. The number of learning samples was 650. This was different for each subnet, and consisted of only noiseless images of the rotating disc counter. The testing set was created based on the idealised images. These images were then distorted by a high amplitude random noise.

Testing results have shown that while the rate of noisy and unchanged pixels did not exceed 30 percent, the rate of correct identifications was above 99%. However, if the noise rate exceeded 30%, the rate of correct recognitions dropped rapidly. The training of one subnet took approximately 2–4 hours on a 300 MHz PC.

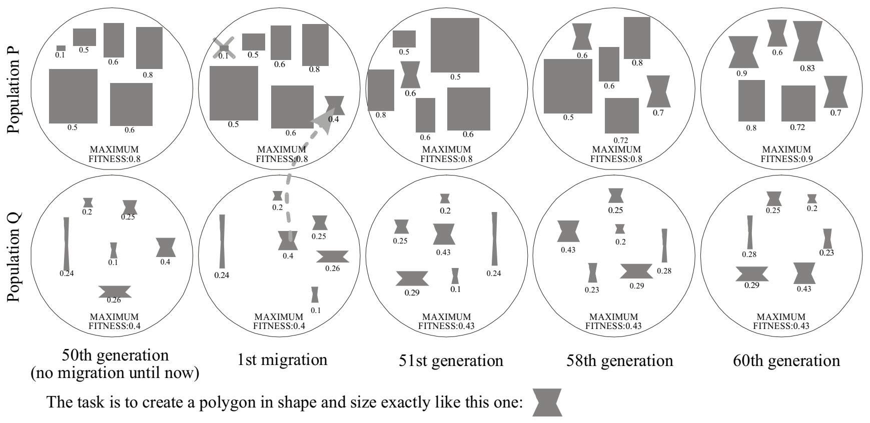


*Fig. 7.b.* The advantage of the Maximum Fitness Guided Migration method: A higher possibility for the combination of the individuals' good qualities. The concave hexagonal shape of the original population $Q$ can be combined with the greater size quality of the original population $P$, and so a deeper or even an absolute optimal error-minimum pit can be found. Remark: The fitness of the exact match is one. If the target object cannot be fully covered, or if it is overcovered the fitness is smaller. The minimum fitness value is zero.

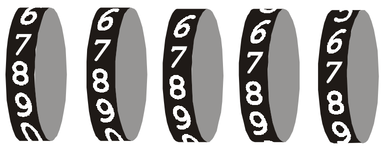


*Fig. 8.* The five sub-digit positions on the least significant digit disc of the rotating disc counter that had to be identified

## 9. Conclusion

The results show that the improved genetic algorithm (*when it is applied for parameter optimization tasks of large dimension numbers*) can provide a good quality solution even if the parameter-space is unknown, if sufficient computer power is at hand. The use of the method is not recommended in applications, where training is to be carried out often and the result has to be obtained in a relatively short time interval.

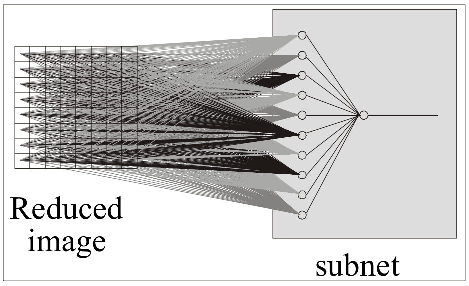


*Fig. 9.* The structure and the connections of a subnet

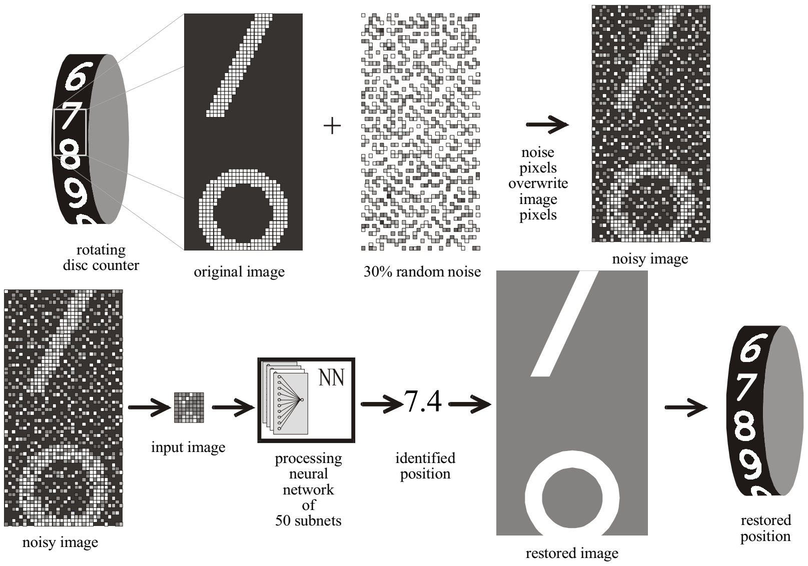


*Fig. 10.* The process of position reconstruction

## References

[1] DIEDERICH, J., *Artificial Neural Networks: Concept Learning*, Los Alamitos, California, IEEE Computer Society Press, 1990, pp. 11–47.

[2] NELSON, M. M. – ILLINGWORTH, W. T., *A Practical Guide to Neural Nets*, Massachusetts, Addison-Wesley Publishing Company, 1991, pp. 128–151.

[3] GOLDBERG, D. E., *Genetic Algorithms in Search, Optimization, and Machine Learning*, Massachusetts, Addison-Wesley Publishing Company, 1991.

[4] VEMURI, V., *Artificial Neural Networks: Theoretical Concepts*, Los Alamitos, California, IEEE Computer Society Press, 1990, pp. 136–144.