INTELLIGENT SOFT COMPUTING IN NUCLEAR ENGINEERING IN BRAZIL

R. Schirru¹, A. S. Martínez¹, C. M. N. A. Pereira², R. P. Domingos¹, M. D. Machado¹ and L. Machado¹

¹Universidade Federal do Rio de Janeiro, Programa de Engenharia Nuclear Ilha do Fundão, 21945-970, P.O. Box 68509, Rio de Janeiro, RJ - Brazil
²Comissão Nacional de Energia Nuclear, Instituto de Engenharia Nuclear, CREA Ilha do Fundão, 21945-970, P.O. Box 68550, Rio de Janeiro, RJ - Brazil

ABSTRACT

Nuclear reactor design and operation often involve important human cognition and decisions. Design optimization, transient diagnosis and core reload optimization, are examples of complex tasks faced during a nuclear reactor design or operation. In order to handle such kind of tasks expert knowledge is required. Due to the complexity involved in the cognition and decisions to be taken, computerized systems have been intensely explored in order to aid design and operation. Following hardware advances, soft computing has been improved and, nowadays, intelligent technologies, such as evolutionary programming, neural networks, expert systems and fuzzy systems are being used to support design and operation. This work presents applications of Intelligent Soft Computing (ISC) to three important cognition problems which are: the nuclear reactor design, the core reload optimization and transient diagnosis. © 1999 Published by Elsevier Science Ltd.

INTRODUCTION

Following the hardware technology advances, Intelligent Soft Computing (ISC) techniques have been intensely studied and improved in the last years, and nowadays practical applications become a reality. Such techniques present several advantages when compared to traditional ones, such as: (i) acquisition of better results in the optimization processes when no prior knowledge is available, (ii) possibility of application to problems to which the conventional methods are not suitable, (iii) simulation of human cognition processes, instead of trying to solve deterministically what is not deterministic.

Practical applications of ISC can be found in many fields of engineering, including nuclear engineering that requires safe and robust systems. The ISC can be applied to nuclear reactor design and operation aiding the nuclear engineer or the operator in difficult tasks that requires expert knowledge or cognition, and also automate optimization processes. Three important applications of ISC in the nuclear engineering are design optimization, core reload optimization and transient diagnosis.
AN OVERVIEW ON INTELLIGENT SOFT COMPUTING TECHNIQUES

Neural networks

Artificial neural networks (Kosko, 1992), many times called only neural networks (NN), are computer algorithms that try to simulate the biological neural networks. As a consequence of this simulation, intelligent features arise. The most popular type of neural net, is the multi-layer feed-forward (see Figure 1) with supervised learning. Such kind of NN have the ability of learning nonlinear mapping between input and output parameters, being suitable to pattern classification or function interpolation.

![Figure 1. The artificial neuron and the multi-layer neural network models.](image)

Each neuron has as its output a nonlinear function of its inputs. The most used are the logistic and the hyperbolic tangent.

The learning process is made by the synapses weight (w) adjusting in order to minimize the error between the output given by the net and the output that should be learned.

Genetic algorithms

Inspired on Darwin's species evolution theory (Darwin 1859), Holland's genetic algorithms (Holland, 1975) manipulate a population of symbolic structures, in order to evolve them to its best adaptation. In GA (Goldberg, 1989, Davis, 1991 and Michalewicz, 1994), the optimization parameters are encoded into symbolic structures, metaphorically called genotypes. The genotypes are formed by a set of genes that carry intrinsic characteristics of the symbolic individual. Such characteristics dictate the adaptability of the individual in the environment, in which it may survive or die. The selection and evolution are made in such a way that stronger individuals have more chance to be selected, transferring their characteristics to their offspring. From generation to generation, the tendency is that strong individuals become stronger and more numerous and the weak individuals tend to be extinct.

The adaptation process starts from a set of candidates to the problem solution, a random generated population of individuals. It is assigned to each individual a fitness that predicts its resistance and adaptability. Then a natural selection process is simulated. In this process, the selection probability of a given individual is function of the fitness.

Choosing a random cross point over the binary string that represents the genotype, followed by the change of parts between the two parents simulates the crossover in a classic GA. The mutation is simulated by the inversion of one of the bits of the genotype according to the mutation probability that is a genetic parameter. Taking into account the reproduction, crossover and mutation, it can be statistically proved, by the schemata theory (Goldberg 1989), that strong individuals may be more numerous in subsequent generations, as the population becomes more adapted, generally concentrating to near-optimum regions.
The parallel search together with the genetic operators above mentioned provides a global exploitation of the search space, requiring no prior knowledge about it.

Genetic Programming

Genetic Programming (Koza, 1992) is an evolutionary programming technique that evolve programs in order to find the computer code that better solves a proposed problem. While in the classical GA the individuals are fixed length bit-strings, in the GP they are data structures representing a computer program. The fitness for each individual is obtained by measuring the performance of the program in realizing a specific task that is the objective of the problem.

As in the GA, genetic operators, such as natural selection, crossover and mutation are used. From generation to generation, programs, which are able to solve the proposed problem, increase in the population.

The early GP systems were developed using List Processing languages. The reason is the way LISP languages manipulate functions and data. In the LISP language there are two entities: lists and atoms. Lists are collections of items and atoms are the variables or constants. Both are called symbolic expressions, s-expressions. No distinction is made between programs and data. An example of individual, representing the expression (* (+ 2 2) (* a c)) is given by the tree in Figure 2.

![Figure 2 - Example of individual](image)

Figures 3 shows an example of crossover in the GP. In the crossover, a node is randomly chosen in each one of the graphs that represent the parents. Then, the sub-trees are exchanged between the parents, forming the offspring.
Figure 3 - Example of crossover

Figure 4 illustrates the mutation. In the mutation, one or more nodes are randomly chosen and the symbols in those nodes are, as well, randomly changed. After mutation, the resulting individual can be completely different from the original one.
The Population-Based Incremental Learning (PBIL) algorithm (Baluja, 1995) is a kind of combination between evolutionary optimization and a hill climbing technique. This algorithm is an extension of Equilibrium Genetic algorithm.

The idea is to create an auxiliary vector indicating the probability of having a “1” in each position in the genotype (individual). The process starts with all probabilities set to 0.5. Randomizing each position of each individual, according to the specified probability, the population is generated. After initialization, each individual in the population is evaluated by the objective function, receiving a fitness value.

A simple approach to update the probabilities vector is the use of the best (higher fitness) and the worst (lower fitness) individuals. Such update is similar to that used in Learning Vector Quantization (Haykin, 1994):

\[ P(i,t) = P(i,t-1), (1-Lr) + x(i).Lr \] (1)

Where \( P(i,t) \) is the i-th element of the probabilities vector on time t and \( Lr \) is the learning rate.

The probabilities are reinforced according to the learning rate (the step by which the probability is increased or decreased). If the individual used for correction has a “1” in some position, the probability is increased, otherwise, it is decreased (remember that the probabilities vector refers to the “1”). The process is repeated until the convergence criteria is reached.

In this work it is used a variation of the method previously described to update the probabilities vector. This variation consists on replacing the use of both best and worst vectors in the population by the use of the N best individuals, with variable learning rate, calculated according to the individual fitness, as follow:

\[ Lr_n = Lr \cdot \frac{fit_n}{fit_{best}} \] (2)
Where $L_{rn}$ is the effective learning rate for individual $n$, $L_r$ the initial learning rate (given by operator), $f_{in}$ the fitness value of the individual $n$ and $fit_{best}$ is the best fitness.

The development of this new computational method named PBIL-N could successfully deal with both numerical and combinatorial optimization problems.

**Expert systems**

Expert systems (ES) (Gonzalez, 1993) are computational tools that use expert knowledge to solve problems of specific domains. In the ES, symbols are manipulated in order to obtain the problem solution. Figure 5 shows a simplified ES diagram represented by its main components: Knowledge Base (KB), Inference Engine (IE), Fact Base (FB) and Man-Machine Interface (MMI).

![Figure 5. Expert System](image)

An important aspect of the expert systems is the separation between knowledge and how this knowledge is manipulated to get the problem solution. This fact allows the use of the same IE for different KB. In other words, different applications can be made using the same shell (IE+MMI).

The KB is the most important part of an expert system once it contains all the information needed to solve a specific problem. The knowledge included in the KB is mapped into an adequate representation (Gonzalez, 1993) and (Israel, 1983). There are several ways to represent the knowledge; the most common is the use of rules (If antecessor Then consequent).

In the IE the symbolic structures into the KB are manipulated so that the knowledge can obtained by inference rules and conflict solution. Common inference methods are the forward chain and the backward chain. In the forward chain (Gonzalez, 1993) the rules contained in the KB are used upon the showing of a fact in the FB that satisfies the antecessor of this rule while in the backward chain the inference is done in the sense of getting antecessor for a given consequent.

The verification and validation process (V&V) of ES (NUREG/CR-6316, 1995) is, however, quite complex. A great difficulty related to V&V for a V&V process of ES arises from the fact that these systems bring certain subjectivity, once the KB is built by a human specialist that input his own view of the problem to be solved.

Verify and validate an ES means to verify and validate all its components. The IE and the MMI use conventional encoding, hence they go through a conventional V&V process. The KB must go through V&V processes that guarantee the absence of both semantic and syntactic errors in the rules. However, an important point that must be emphasized in the V&V process is that even if a KB is verified and validated it does not mean that the solutions presented are correct. It happens because the conflict resolution criteria used may not be adequate to the rules used in the KB. Thus, in the ES V&V, not only the syntax and semantic of rules must be verified and validated but one must also worry about the kind of inference that is being done.
NUCLEAR REACTOR DESIGN OPTIMIZATION

During the reactor core design, a set of parameters must be adjusted and a lot of constraints must be satisfied in order to obtain a safe and economical reactor. The main objective is to find the best configuration of the design parameters. Hence, a good representation of the phenomena related to the neutron interactions in the reactor core as well as an efficient optimization technique are required.

Due to their simplicity and low computational cost, gradient search has been used as the optimization method. Most of them, such as the one described in Rozon (1992), use linear programming techniques. However, because of the nature of exploitation of these hill-climbing-like methods, their application to a multimodal search space can lead to a local optimum.

Nuclear reactor core design optimization process includes non-linearity, discontinuities and multimodality, becoming a complex problem with large number of state parameters to be optimized subject to a large number of constraints.

Early studies (Pereira, 1997) have pointed to GA and NN as a promising tool to be used in core optimizations. Recent investigations (Pereira, 1999) revealed some advantages when compared to a classical nonlinear optimization method based on linear programming.

A nuclear reactor core design optimization problem

Here it is described an optimization problem which has characteristics that are common to most of the nuclear reactor core optimization problems. The objective is to minimize the average peak factor $f_p$ for the three-enrichment zone reactor shown in Figure 6. The constraints are: average flux, $\phi = 8.0 \times 10^{-5} \pm 1\%$ (source normalized), neutron multiplication factor, $k_{eff} = 1.0 \pm 0.01$, and submoderation. In order to reach the objective, the fuel radius, as well as cladding thickness, equivalent radius, the enrichment of the three zones, cladding material and fuel material, are allowed to vary according to the ranges specified in Table 1. The region dimensions, $R_1$, $R_2$ and $R_3$ remain constant and the values are 86 cm, 38 cm and 18 cm respectively.

![Characteristics Cell Reactor](image)

**Figure 6. The reactor and characteristic cell**
Then, the optimization problem can be written as:

Minimize:

$$f_m(R_f, \Delta r, \Delta m, E_1, E_2, E_3, m_f, m_c)$$

Subject to:

$$7.92 \times 10^{-5} \leq \phi(R_f, \Delta r, \Delta m, E_1, E_2, E_3, m_f, m_c) \leq 8.08 \times 10^{-5};$$ (3)

$$0.99 \leq k_{\text{eff}}(R_f, \Delta r, \Delta m, E_1, E_2, E_3, m_f, m_c) \leq 1.01;$$ (4)

$$\frac{dk_{\text{eff}}}{dV_m} > 0;$$ (5)

$$R_{f\min} < R_f < R_{f\max};$$ (6)

$$\Delta r_{\min} \leq \Delta r \leq \Delta r_{\max};$$ (7)

$$\Delta m_{\min} \leq \Delta m \leq \Delta m_{\max};$$ (8)

$$E_{1\min} \leq E_1 \leq E_{1\max};$$ (9)

$$E_{2\min} \leq E_2 \leq E_{2\max};$$ (10)

$$E_{3\min} \leq E_3 \leq E_{3\max};$$ (11)

Fuel material = \(m_f = \{\text{UO}_2, \text{U-metal}\};\) (12)

Cladding material = \(m_c = \{\text{Zircaloy, Al or Stainless-304}\};\) (13)

where \(V_m\) is the moderator volume and the min and max subscript refers to the minimum and maximum values given in Table 1.

Modeling an optimization problem using genetic algorithm consists in two basic steps: choosing the appropriate genotype structure and designing a suitable objective function that must evaluate the fitness of each genotype.
The genotype represents the set of parameters that can vary in the nuclear core design. The parameters not included in the genotype representation cannot be changed during the optimization process. The genes in the proposed model are the binary representation of the parameters values, coded into binary fixed length strings. The length of the strings dictate the precision of the codification. The lower limit of the gene range correspond to all bits set to zero as well as the upper limit correspond to all bits set to one. In other words, the range of each variable is discretized into $2^n$ values, where $n$ is the genotype length.

An example of a genotype for this problem is presented in Figure 7. For the continuous parameter, the phenotype uses a digital to analog (D/A) decodification of the genotype, and for the material, enumerated decodification is used, such as 0 to UO$_2$ and 1 to U metal.

<table>
<thead>
<tr>
<th>Rf</th>
<th>$\Delta r$</th>
<th>$\Delta m$</th>
<th>E1</th>
<th>E2</th>
<th>E3</th>
<th>mf</th>
<th>mc</th>
</tr>
</thead>
<tbody>
<tr>
<td>1000000</td>
<td>11111111</td>
<td>0000001</td>
<td>1000000</td>
<td>1000101</td>
<td>1001101</td>
<td>0</td>
<td>10</td>
</tr>
<tr>
<td>0.889</td>
<td>0.254</td>
<td>0.0272</td>
<td>3.50</td>
<td>3.56</td>
<td>3.65</td>
<td>UO$_2$</td>
<td>Al</td>
</tr>
</tbody>
</table>

Figure 7 - Example of genotype and phenotype

To each genotype is assigned a fitness that is a function of the variables to be maximized or minimized and of the constraints of the problem, calculated by the reactor physics simulators with the parameters decoded from the genotype. The generalized fitness can be written as:

$$ f = y + \sum_{i=1}^{\infty} F(x_i) $$

$$ F(x_i) = \begin{cases} 0, & x_i^{\text{min}} \leq x_i \leq x_i^{\text{max}} \\ r_i |\Delta x_i|, & \text{otherwise} \end{cases} $$

Where $y$ is the objective variable that must be maximized or minimized, $\Delta x_i$ means how far variable $x_i$ is from the middle of the constraint range, $x_i^{\text{min}}$ and $x_i^{\text{max}}$ are the lower and upper limits of the constraint range, and $r_i$ is the penalty constant.

The specific objective function for this problem is:
\( f = \begin{cases} 
\ell, & \text{if } \Delta k_{\text{eff}} \leq 0.01; \Delta \phi \leq 0.01 \phi_0; \frac{\Delta k_{\text{eff}}}{\Delta V m} > 0 \\
\ell + r_1 \Delta k_{\text{eff}}, & \text{if } \Delta k_{\text{eff}} > 0.01; \Delta \phi \leq 0.01 \phi_0; \frac{\Delta k_{\text{eff}}}{\Delta V m} > 0 \\
\ell + r_2 \Delta \phi, & \text{if } \Delta k_{\text{eff}} \leq 0.01; \Delta \phi > 0.01 \phi_0; \frac{\Delta k_{\text{eff}}}{\Delta V m} > 0 \\
\ell + r_3 \frac{\Delta' k_{\text{eff}}}{\Delta V m}, & \text{if } \Delta k_{\text{eff}} \leq 0.01; \Delta \phi \leq 0.01 \phi_0; \frac{\Delta' k_{\text{eff}}}{\Delta V m} < 0 \\
\ell + r_1 \Delta k_{\text{eff}} + r_2 \Delta \phi, & \text{if } \Delta k_{\text{eff}} > 0.01; \Delta \phi > 0.01 \phi_0; \frac{\Delta' k_{\text{eff}}}{\Delta V m} > 0 \\
\ell + r_1 \Delta k_{\text{eff}} + r_2 \Delta \phi + r_3 \frac{\Delta' k_{\text{eff}}}{\Delta V m}, & \text{if } \Delta k_{\text{eff}} > 0.01; \Delta \phi > 0.01 \phi_0; \frac{\Delta' k_{\text{eff}}}{\Delta V m} < 0 \\
\end{cases} \) (16)

Where

\[
\Delta k_{\text{eff}} = |1.0 - k_{\text{eff}}| \quad (17)
\]

is the penalty for the configurations where \( k_{\text{eff}} \) is out of constraints.

\[
\Delta \phi = |\phi - \phi_0| \quad (18)
\]

the penalty for the configurations where \( \phi \) is out of constraints.

and

\[
\frac{\Delta' k_{\text{eff}}}{\Delta V m} = \frac{k_{\text{eff}} - k'_{\text{eff}}}{0.03 V m} \quad (19)
\]

the penalty for the configurations where the reactor is submoderated, where \( k'_{\text{eff}} \) is the multiplication factor obtained for 3% of variation in \( V m \) (as the fuel volume - \( V f \) - remains constant, the rate \( V m/Vf \) is altered. The values used for \( r_1, r_2 \) and \( r_3 \) were 10.

The Genesis (Grefenstette, 1990) was used as a base code to implement the GA. The reactor simulations were made using the Hammer (Suich, 1967) system.

The results obtained by the application of the proposed GA to the above described problem are shown in Table 2.
Table 2 - Results obtained by the application of the proposed GA to the optimization problem.

<table>
<thead>
<tr>
<th>Objectives</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minimum average peak factor</td>
<td>1.295</td>
</tr>
<tr>
<td>Average Flux</td>
<td>8.023 x 10^{-5}</td>
</tr>
<tr>
<td>(k_{\text{eff}})</td>
<td>0.9948</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rf (cm)</td>
<td>0.6280</td>
</tr>
<tr>
<td>(\Delta r) (cm)</td>
<td>0.1604</td>
</tr>
<tr>
<td>(\Delta m) (cm)</td>
<td>0.6808</td>
</tr>
<tr>
<td>E1 (%)</td>
<td>2.7087</td>
</tr>
<tr>
<td>E2 (%)</td>
<td>3.0394</td>
</tr>
<tr>
<td>E3 (%)</td>
<td>4.7638</td>
</tr>
<tr>
<td>mf (%)</td>
<td>U-metal</td>
</tr>
<tr>
<td>mc</td>
<td>Stainless-304</td>
</tr>
</tbody>
</table>

The genetic parameters used were population size of 300 individuals, crossover and mutation rate of 60% and 1% respectively, ranked selection and elitism. Possibly because of the difficulty found in this problem, the GA took more generations to reach the stop criteria.

As can be seen in Figure 8, the convergence of the fitness has occurred, and the GA found a good configuration that has generated small peak factor (with all the constraints satisfied). The expert knowledge could not indicate a better configuration.

During the genetic algorithm evolution, many other valid configurations (configurations that satisfy the constraints) could be found, however, higher peak factors were associated with them. For example, it could be found valid configurations with peak factor greater than 20. On the other hand, peak factors less than 1.5
could be found for different materials. These discontinuities and multimodalities can often lead an expert to find a local optimum.

The applicability of the method to mixed continuous/discrete optimization was demonstrated by the application of the GA to the complex problem. It is important to outline that the GA contrast with other methods in the sense that it does not need any prior knowledge about the search space, fact that allows the optimization of problems without concerning about the existence of derivatives or its inverse, continuity, limits, and so on. Due to the fact that the GA does not need prior knowledge about the search space, this model can be adapted and applied to any kind of problem for which it is possible to represent the points in the search space by a binary string and evaluate these points by an objective function.

THE NUCLEAR CORE RELOAD OPTIMIZATION PROBLEM

Nuclear fuel management is a difficult task that involves complex combinatorial optimization problems. For many years fuel management have been solved by expert knowledge. The main objective is to find the best loading pattern (LP) of fuel in the nuclear reactor core that leads to a maximization of the operating time, minimizing the operation cost.

In the nuclear core reload, fuel assemblies must be replaced in the core according to its isotopic enrichment, related to the cycles in the core. Besides, fresh fuel must substitute for some old assemblies. In other words, fuel assemblies with different ranges of isotopic enrichment must be positioned in the reactor core so that the operating time of that load is maximized.

In order to evaluate the loading pattern, the reactor must be simulated by a computer code calculating the fuel cycle length or related parameters such as the peaking factor. During the optimization process, the simulation must be done many times, to evaluate each loading pattern (LP).

The nuclear reactor core reload problem (CRP) consists in combining fuel assemblies with different enrichment with the possible positions in the nuclear core. The CRP is, then, a combinatorial problem that can be translated to the classical Traveling Salesman Problem (TSP) which has special interest for the optimization researchers.

In the TSP, the objective is to find the minimum distance that completes a tour for a set of cities passing only once in each city and coming back to the starting one. Mathematically, the problem is to find the minimum Hamiltonian cycle in a given graph.

The fuel assemblies in the CRP correspond to the cities in the TSP and the position in the core is related to the order of the cities to be visited. Hence, the solution of the CRP is the solution of the TSP.

Genetic Algorithms and the Traveling Salesman Problem

Due to the complexity imposed by the TSP, classical optimization methods have been giving place to alternative solutions, including specific heuristics, simulated annealing (SA), genetic algorithms (GA) and others evolutionary algorithms (EA). The results obtained by the use of GA in solving the TSP have encouraged researchers to the application of such technique.

In the genetic modeling of a TSP, the genes represent the cities, and the genotype represents the tour where the order of the genes in the genotype is the order of the cities to be visited. If the classical crossover is applied to such genotype representation, it may generate invalid offspring, with some cities twice visited and others not present in the tour, as can be seen in Figure 9.
Intelligent soft computing

Figure 9 – Classical crossover in the TSP genotype

Note that both tours are invalid candidates to solution. To face such problem, modifications in the crossover were proposed in order to generate valid offspring, such as Partially Mapped Crossover (PMX), Order Crossover (OX), Cycle Crossover (CX) and others described in Holland (1975). Bean (1994) proposed the Random Keys (RK) to decode the genotype instead of modify the crossover. Schirru (1997) proposed the List Model (LM) as an alternative method that do not need the modifications in crossover. The last one has been successfully applied in practical CRP (Chapot, 1999).

The decodification in the LM occurs as follow: let G={G1,...,Gn} be the genotype formed and its respective genes. G1,...,Gn can be any integer between 1 and the number of cities. Let C={C1,...,Cn} be a base city list. To decode G in a valid tour T, each Gi\textsuperscript{th} city element must be removed from C and pushed into T until G is empty.

A practical application of genetic algorithms in the core reload problem

Here, a practical application, made by Chapot (1999) will be described and commented. The reload optimization system, called ALGER (Genetic Algorithm Applied to Reload Studies) is a GA modeled based on the Genetic Search Implementation System, Genesis (Grefenstette, 1990) and the Advanced Nodal Code, ANC (Liu, 1985), using the LM approach to the TSP.

The result of the simulation of the ANC program is a value corresponding to the cycle length, or peaking factor, which makes part of the objective function. This value is used as fitness for each individual. Then the GA operates over the population. The process is repeated until the convergence criteria is reached.

The ALGER system has been applied to optimization problems for a Brazilian 626 MW Westinghouse PWR, called Angra 1 nuclear power plant. Angra 1 core is comprised of 121 fuel assemblies, 80 of them belonging to 8-fold symmetric type, 40 having a 4-fold symmetry and one fuel assembly located in the middle of the core.

Applying the ALGER system to the minimization of the radial peaking factor, F\textsubscript{XY}, using eight-core symmetry and two dimensional geometry modeled in ANC, it was obtained, at the end of 120 generations (stop criterion) the value of 1.304 for the radial peaking factor. Figure 10 illustrate the convergence of the F\textsubscript{XY}. In the third generation, the GA found F\textsubscript{XY}=1.327 which is much lower than the value obtained by the manual optimization F\textsubscript{XY}=1.344. As a result, the cycle length for this LP was 3 ppm (about one Effective Full Power Day - EFPD).

Another test with objective of cycle length maximization was made. In this case, the results obtained by manual method, using the ANC 3D quarter-core full-cycle depletion, was Nuclear Enthalpy Power Peaking Factor (FAH)=1.366 and Cycle Length = 228 FFPD. Maximizing cycle length implies maximization of end of cycle boron concentration. However, to minimize the computational costs it was used the equilibrium of Xenon C\textsubscript{B} as the parameter to be maximized. At the end of 100 generations, the maximum C\textsubscript{B} found was 1198 ppm, with F\textsubscript{XY}=1.365, as shown in Figure 11.
Extrapolating the critical boron concentration to 0 ppm, it was found a cycle length of 277 EFPD, 49 EFPD greater than the value obtained by manual optimization. The results show that the CRP is a very difficult task for the expert, but the ALGER system could reach better configurations automatically.

**The PBIL applied to the TSP**

The PBIL is another optimization tool that can be applied to the nuclear core reload problem. The models that can be used are the same approaches described in the last section. In order to test comparatively the PBIL with different approaches, TSP benchmarks were selected: Oliver 30 (Oliver, 1987) and Rykel 48
Intelligent soft computing

(Bibix, 1995). In the problem encoding/decoding the Random Key (RK) and the List Model (LM) were used.

Oliver 30 is a TSP benchmark containing 30 cities to which the least distance is 423.7. The parameters used by the algorithms are shown in Table 3 and the obtained results using both algorithms (PBIL and PBIL-N) are presented in Table 4, for both types of encoding used.

<table>
<thead>
<tr>
<th>Table 3 – Configuration parameters for Oliver 30</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter</td>
</tr>
<tr>
<td>----------------------------------</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Population Size</td>
</tr>
<tr>
<td>Learning Rate</td>
</tr>
<tr>
<td>Negative Learning Rate</td>
</tr>
<tr>
<td>Mutation Rate</td>
</tr>
</tbody>
</table>

Table 4 – Results obtained for Oliver 30

<table>
<thead>
<tr>
<th>Generations</th>
<th>PBIL</th>
<th>PBIL-N</th>
</tr>
</thead>
<tbody>
<tr>
<td>RK</td>
<td>ML</td>
<td>RK</td>
</tr>
<tr>
<td>250</td>
<td>921.9</td>
<td>910.4</td>
</tr>
<tr>
<td>500</td>
<td>723.2</td>
<td>749.3</td>
</tr>
<tr>
<td>750</td>
<td>723.2</td>
<td>502.9</td>
</tr>
<tr>
<td>1000</td>
<td>581.4</td>
<td>425.5</td>
</tr>
<tr>
<td>1250</td>
<td>445.4</td>
<td>423.7</td>
</tr>
<tr>
<td>1500</td>
<td>423.7</td>
<td>423.7</td>
</tr>
</tbody>
</table>

Figure 12 shows the convergence of the algorithms in the solution for Oliver 30. It is clearly seen that for such a simple problem like this one, both algorithms were capable of finding the smallest distance, being PBIL-N faster in locating this point for both types of encoding used.

Rykel 48 is a TSP benchmark containing 48 cities to which the least distance is 14422. The parameters used by the algorithms are shown in Table 5 and the obtained results using both algorithms (PBIL and PBIL-N) are presented in Table 6, for both types of encoding used.
Table 5 – Configuration parameters for Ryke148

<table>
<thead>
<tr>
<th>Parameter</th>
<th>PBIL</th>
<th>PBIL-N</th>
</tr>
</thead>
<tbody>
<tr>
<td>Population Size</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>Learning Rate</td>
<td>0.08</td>
<td>0.03</td>
</tr>
<tr>
<td>Negative Learning Rate</td>
<td>0.04</td>
<td>0.01</td>
</tr>
<tr>
<td>Mutation Rate</td>
<td>0.0</td>
<td>0.0</td>
</tr>
</tbody>
</table>

Table 6 – Results obtained for Ryke148

<table>
<thead>
<tr>
<th>Gerações</th>
<th>PBIL</th>
<th>PBIL-N</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>RK</td>
<td>ML</td>
</tr>
<tr>
<td>500</td>
<td>39817</td>
<td>40362</td>
</tr>
<tr>
<td>1000</td>
<td>35055</td>
<td>37820</td>
</tr>
<tr>
<td>1500</td>
<td>29701</td>
<td>36133</td>
</tr>
<tr>
<td>2000</td>
<td>26110</td>
<td>32697</td>
</tr>
<tr>
<td>2500</td>
<td>23472</td>
<td>29769</td>
</tr>
<tr>
<td>3000</td>
<td>19130</td>
<td>28742</td>
</tr>
<tr>
<td>3500</td>
<td>17839</td>
<td>28512</td>
</tr>
<tr>
<td>4000</td>
<td>16979</td>
<td>27961</td>
</tr>
<tr>
<td>4500</td>
<td>16022</td>
<td>26073</td>
</tr>
<tr>
<td>5000</td>
<td>15568</td>
<td>21132</td>
</tr>
<tr>
<td>5500</td>
<td>15467</td>
<td>20692</td>
</tr>
</tbody>
</table>

Figure 13 shows the convergence of the algorithms in the solution for Ryke148. It is clearly seen that this problem is a bit more complex than Oliver 30 is. Both algorithms were not capable of finding the exact point for the least distance; nevertheless the PBIL-N algorithm presents smaller error than the PBIL algorithm in both types of encoding used.

![Figure 13 – Convergence for PBIL and PBIL-N algorithms in Ryke148 problem](image)

The PBIL-N algorithm using the new probability vector updating presented better results for the problems analyzed in this research either referring to precision terms or referring to the computational processing time compared to the traditional solution of the PBIL algorithm.
The implementation of the new probability vector updating resulted in the delivery of a new evolutionary tool that shows good applicability in the solution of complex combinatorial problems, analogue to those of PWR nuclear reactor reload with great potential to be used in its optimization.

INTELLIGENT MONITORING SYSTEMS AND TRANSIENT CLASSIFICATION

Pattern recognition has been of special interest for the ISC community. Many techniques such as classifier systems (Goldberg 1989), fuzzy systems (Kosko 1992), and neural networks (Kosko1992), have been explored.

The Minimum Centroids Set method based on GA

In order to introduce the Minimum Centroids Set (MCS) method (Pereira, 1998), the simplest classification method based on centroids, which we call the Simple Centroid (SC) method will be presented at first. The idea is to create one cluster per class, and represent each class by the cluster centroid. A given sample is said to belong to the class which the centroid is the closest one. Each component of the centroid is given by the arithmetic average of the components of the cluster. The distance from one point to the centroid is given by the Euclidean distance between two points. An example of clustering and classification, based on the SC for two classes represented by a set of points in a 2-dimensional space, can be seen in Figure 14.

\[ V_2 \]
\[ V_1 \]

Figure 14. Example of two clusters and their respective centroids $C_1$ e $C_2$.

In the example shown in Figure 14, the patterns $p_1$ e $p_2$ are easily distinguished by their centroids. However, in the time series, the patterns are spread along the time axis, and they can often interlace themselves creating confusion areas, that may lead to misclassifications, as shown in Figure 15.

\[ V \]
\[ t \]

Figure 15. Example of two time series $V_1(t)$ e $V_2(t)$ and their respective centroids $C_1$ e $C_2$ where pattern $p_1$ is classified as belonging to class $C_2$ and $p_2$ as belonging to class $C_1$. 
Consider now the hypothetical example of two other time series in Figure 16. Making use of the SC method it will be found that the classes centroids will be located at the middle of the time interval that may lead to misclassification for all patterns to be classified.

![Figure 16. Two hypothetical time series where the centroids are coincident](image)

In this case, the simple algorithm described in Figure 18 may classify perfectly a given pattern P.

```plaintext
MinimumDistance = d(P, C(1,1));
For c = 1 to NumberOfClasses do begin
    For i = 1 to NumberOfSubclassesPerClass do begin
        If d(P, C(c,i)) < MinimumDistance Then begin
            MinimumDistance = d(P, C(c,i));
            Class = c;
        End If;
    End For;
End For;
```

![Figure 18. The algorithm of classification on a multiple-centroid scenario](image)

In Figure 18, C(c,i) is the ith centroid of class c and d(P, C(c,i)) is the distance between point P and the cluster C(c,i).
The Minimum Centroid Set (MCS) method consists in finding the best set of centroids that better distinguishes the classes from each other, considering one or more centroids per classes. In other words, the MCS method intends to find the minimum number of time partitions (and the positions) of the classes in which the subclass centroid better classifies a set of test patterns, using the simple algorithm in Figure 18, that we will call the MCS algorithm.

The goal of the proposed method is to apply the well-known and simplest mathematical metric in the regions of the domain where they work well. But to find this sub-domains in which the application of the simple method is well succeeded may be a hard task. Because of the nature of the optimization to be done and the poor prior knowledge about the search space, it is proposed the use of Genetic Algorithm (GA) (Goldberg 1989) as the optimization technique. The use of the genetic algorithm in the search for the best centroid set is the main subject of this approach.

The genetic algorithm generate candidates for the best partitions in which the SC calculates the centroids of each subclass. These centroids are then used in the MCS algorithm. The MCS algorithm tries to classify the pattern test set, sending the number of well succeeded classifications (performance) back to the GA that, in its turn, uses it to guide the optimization search. The process is repeated until either the convergence criteria is achieved or the performance is satisfactory, when the loop may be interrupted.

The clusters are coded into binary strings that have the length equal to the number of training patterns per accident. The groups of 1's (one) or 0's (zeros) together form a cluster or subclass. An example of the genotype that encodes the clusters of two hypothetical classes in Figure 19 is shown in Figure 20.

![Figure 19. Two hypothetical time series representing two classes.](image)

![Figure 20. Example of genotype](image)

The fitness reward the high performance of correct classification and low number of centroids, as described below.
\[ f = K_d \cdot P - (K_c + K_o) \cdot C \]  

(20)

where P is the number of correct diagnosis, C is number of clusters, \( K_d \) is the weighting factor for the performance, and \( K_c \) is the weighting factor for the number clusters and \( K_o \) is the offset factor.

The fitness suggested presents three constants to be adjusted according to the relative importance of the variables to be optimized.

Once defined the genotype shape and the fitness function, the last thing to do is to adjust the genetic parameters with the aim of optimizing the convergence process of the genetic algorithm.

In order to validate the MCS method for pattern recognition, three simple reference cases, which results (best centroids) are well known were created. Figure 21 shows them.

As the GA works only with the time axis, to increase the number of variable time series that represents each class not necessarily increase the complexity for the GA.

The Genesis code (Grefenstette, 1990) was used to implement the system, and the results obtained for the centroids optimization can be seen in Table 7.

<table>
<thead>
<tr>
<th>Test</th>
<th>Time of Centroid 0</th>
<th>Time of Centroid 1</th>
<th>Time of Centroid 2</th>
<th>Performance Correct / trials</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test-1</td>
<td>28.0</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Test-2</td>
<td>14.0</td>
<td>42.0</td>
<td>42.5</td>
<td>-</td>
</tr>
<tr>
<td>Test-3</td>
<td>1.5</td>
<td>3.0</td>
<td>28.0</td>
<td>54.5</td>
</tr>
</tbody>
</table>

It is observed that the optimum values for the centroids were reached with small errors, presenting very good diagnosis results (zero or very low percentage of misclassifications).

After validation, the method was applied to a real case involving three transients in a nuclear power plant (NPP). Such kind of pattern recognition has been treated using artificial neural networks, e.g. Bartlett (1992) and Bartal (1995). It was chosen three typical transients in nuclear power plants represented by 15 variables each, and it is assumed by hypothesis that all variables are required to the transient recognition.
The transients considered were blackout, lost of coolant accident (LOCA) and the steam line break. These transients were represented by tables with 60 points per variable, generated by simulation with time-step of one second (by hypothesis the transients can be characterized by time series of 60 seconds). The variables that considered in the nuclear transients are the Primary flux, Nuclear power, Thermal power, Cold Leg temperature, Hot leg temperature, Average temperature, Subcooling margin, Pressurizer pressure, Steam generator wide range, Steam generator narrow range, Steam pressure, Feed water flow, Break flow, Pressurizer level and Steam flow.

The comparisons of the obtained results with those obtained by the SCM, and an Adaptive Vector Quantization (AVQ) Neural Network (NN) (Alvarenga 1998) are shown in Table 8.

<table>
<thead>
<tr>
<th></th>
<th>Correct diagnosis / Total trials</th>
<th>Correct diagnosis (%)</th>
<th>Partitions</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCM</td>
<td>144 / 180</td>
<td>82%</td>
<td>1</td>
</tr>
<tr>
<td>MCS</td>
<td>171 / 180</td>
<td>95%</td>
<td>3</td>
</tr>
<tr>
<td>AVQ NN</td>
<td>155/180          86%</td>
<td>86%</td>
<td>-</td>
</tr>
</tbody>
</table>

The GA was very precise to optimize the centroids for the reference scenarios, converging to a configuration that is very near to the known best solution. Table 8 shows the ability to optimize the relationship between performance and number of centroids for the real nuclear transient identification, improving considerably the efficiency of the SCM. Besides, the results were better than the ones obtained by the NN, indicating that MCS based on GA is a competitive tool. In order to improve this tool, a “don’t know” classification approach is under investigations.

Learning optimized classification rules by means of GP

Recently, some applications of GP to the nuclear engineering can be seen (Domingos, 1997). Here, results of the application of GP to nuclear transient classification are shown. The sample problem is the same described in the last section. The resulting program is a rule base constituted by the rules in the form:

\[
\text{IF} \leq \text{"Variable" } \text{"Constant" THEN "Action 1" ELSE "Action 2"}
\]

The term “Constant” is a value identified by the system. “Action 1” or “Action 2” can be any actions such as the transient classification or another rule IF<= or IF>.

The representation model adopted presents several advantages. The crossover allows the genetic material exchange in many levels, such as rules, constants or variables. Besides, the traditional crossover could generate rules with no sense, such as IF Blackout THEN Temp, or (In The Cold Leg > X) ELSE “Action 2”. In this case the rule’s syntax is correct but the semantic is not, wasting computational time in the rule’s evaluation. To deal with such situation, restrictions are imposed. The possible structures that represent the individuals are fixed, identifying what are the arguments of each specific function.

After several experiments, some different classification programs where found, including different variables. Such behavior emphasizes the probabilistic characteristic of the evolutionary computation. To each solution, it was presented a test set constituted by the learned patterns with 5% of noise. All of them presented a good generalization capacity. Some examples of the solutions found are:

```lisp
(defun Class (V1 V2 V3 V4 V5 V6 V7 V8 V9 V10 V11 V12 V13 V14 V15 Time)
  (IF< V8 204.79842 'Blackout
  (IF>= V10 192.09966 'LOCA 'STEAM)))
```
(defun Class (V1 V2 V3 V4 V5 V6 V7 V8 V9 V10 V11 V12 V13 V14 V15 Time)
   (if< V10 208.20222
        (if< V15 36.50531 'Blackout 'STEW ZOCA))

Note that, in spite of the large amount of data to be learned, the GP could mine the data finding a small set of classification rules.

An expert monitoring system

Monitoring nuclear power plants systems face difficult problems to be solved throughout their development such as: parameter variation calculation, system maintenance and verification and validation process. Here is presented a model of monitoring system for large plants (Machado, 1997) using artificial intelligence techniques that solves the problems above mentioned and also contributes to increase its reliability.

This model features an expert system (Gonzalez, 1993) that uses object-oriented concepts (Leving, 1990) and it was used in SICA’s (Angra-I Computer Integrated System) implementation with good results.

A basic point of this approach is the knowledge representation adopted in the construction of the knowledge base. Such representation allows a better performance of the system when compared to conventional ones.

The knowledge domain is the collection of all variables that one wants to monitor. This domain can be mapped using parent-offspring structures that form a net, as shown in Figure 22. A node represents a variable and all nodes are expressed using object-oriented formalism, where each node is an object. This net can be understood as being the collection of all rules contained in the expert system knowledge base (KB).

The connection among generating nodes (generating variables) and the resulting nodes represents the calculation made to obtain this offspring from the parents. This connection, called operator, is an attribute of the class that represents the variables. The value associated with this attribute dictate the operation to be done.

Once the main interest is monitoring, the rules of knowledge base must present dynamic features. One example of rule that is contained in KB would be:

\[\text{IF there was change in variable 6 or in variable 7 \ THEN apply operator and calculate variable 10 with new variables 6 and 7}\]

With this rule it can be seen that the dependency of variables gets explicit, once a generating variable - in this example variables 6 and 7 - is altered, the offspring variable (variable 10) will be calculated once again. Thus, this knowledge representation becomes a natural expression for variable dependency, eliminating any necessity of turning it explicit.
Building the knowledge base is always made by an expert through a man-machine interface (MMI) developed to be familiar to the user. This base is then manipulated in the inference engine during the monitoring process.

In the forward chain rules, an initial fact from the base satisfies the antecessor of a rule making it to be triggered. When a rule is triggered it generates a new fact that must satisfy the antecessor of another rule successively till the last rule have been triggered, thus generating the solution of the problem.

The developed conflict resolution strategy is based in the knowledge representation adopted. Thus, the rules which take lower level variables have priority in being triggered before the rules which depend on upper level variables. In case of rules that have same priority, the first generated in the Fact Base will be triggered.

The inference that is being presented using this conflict solving contains synchronizing and contesting concepts used in the construction of real time systems, due to its structure. Hence, the optimized indifference can be considered.

In conventional expert systems the data synchronization is forced by a managing program that uses commands from the operational system, restricting their usage to the equipment to which the system was designed. In the proposed expert system this restriction is no longer valid because no commands from the operational system was used. Therefore it can be processed in any computational environment. The real time processing depends on the equipment. If it is able to process in real time, so a real time response will be given. Otherwise, the response will be invalidated and one should increase the hardware processing capacity so that the real time processing can be available.

Another advantage of this conflict resolution is its capacity of acquiring new knowledge, that is, new rules, keeping the inference correct, once it is completely integrated to the representation of knowledge adopted.

The maintenance of this expert system is much easier then the conventional ones, due the fact that most of the conventional code becomes knowledge base and, so, it does not belong to the expert system conventional codification, decreasing considerably the number of code lines. Besides, the used object-oriented approach also brings a contribution to ease the maintenance of the prototype.

Another advantage comes from the fact that the KB is not fixed computer code. Any change performed in the knowledge base is made using the MMI and thus, no program code must be rewritten.

As the main objective is monitoring nuclear power plants, the recommendation followed was NUREG-6316 (1995) for verifying and validating the knowledge base. The NUREG-6316 suggests that V&V process of knowledge base gathers the following items: a) analysis of syntax errors, b) analysis of semantic errors and c) analysis of errors related to the object oriented programming.

The adopted representation of knowledge in a parent offspring tree and its perfect integration with the conflict solving strategy barrels rules with syntax errors from the knowledge base. Meanwhile, the probability of semantic errors was diminished once the responsibility of building knowledge was transferred from a system analyst to an engineer who is an expert in the subject.

Another important fact to reliability is that even when the knowledge base has been verified and validated and so far is correct, it can point out solution that might not be correct. This is possible due to an incorrect inference that may be performed. In this model it can be assured that when a knowledge base is correct it will point out correct responses once the inference done is perfectly integrated with the adapted knowledge representation. So, in order to verify and validate the system it is enough to verify and validate the knowledge base, which, in this case, means performing tests.

The application consists on implementing a substantial part of SICA (Martinez, 1986 and 1988), safety parameters display from Angra I nuclear power plant. This system has been developed to act either in critical situation and in regular operation conditions.
Using Artificial Intelligence, through the development of an Expert System the presented encoding have been simplified. The system maintenance was simplified and the time and cost of development have been decreased.

This way, through a natural knowledge representation and an adequate inference structure, many of the problems related to safety monitoring systems in complex plants have been eliminated and the developed system could reach high degree of reliability with an easy V&V process.

**CONCLUSIONS**

Computational tools have important to the nuclear engineering. Together with the hardware technology evolution, the software technology has also been improved. For many years, the efforts were concentrated on the improvement of complex physical simulations in order to aid design and plant operation. Nowadays, intelligent soft computing (ISC) can be useful to help nuclear plant design and operation. Such modern approach, in which the software simulates human cognition, intends to improve design optimization and aid difficult decisions, considering the human factors.

**REFERENCES**


Suich, J. E. and Honec, H. C. (1967) The HAMMER System Heterogeneous Analysis by Multigroup Methods of Exponentials and Reactors, Savannah River Laboratory, Aiken South Carolina.