OPTIMISATION OF ELECTRICAL ENERGY CONSUMPTION USING GENETIC ALGORITHMS

F. Garzia, G.M. Veca
Università di Roma "La Sapienza"

ABSTRACT

An optimisation technique for electrical energy consumption based on genetic algorithms is presented. The proposed technique is able of ensuring a consistent energy waste reduction by means of a high flexibility and efficiency in energy management.
This kind of system can be hosted on a quite simple electronic microcontroller whose performance, such as the number of operations per second or the memory occupation have been studied in this paper.

1. INTRODUCTION

The optimisation of electrical energy consumption can be achieved using mixed hardware [1,2] and software techniques [3,4].
In the most of situations, to program the system, a certain number of data must be collected for a long time to know in advance the exigencies of the final users together with their energy consumption time table.
These kind of systems are characterized by a great limitation in adapting to any variations, that must be properly implemented in the software to be correctly managed.
This problem can be avoided using evolutionary strategies such as the one offered by the genetic algorithms [5,6].
Genetic algorithms (GAs) offer the great advantage of evolving their behaviour to match with the behaviour of the final users, using a mechanism that is very similar to the one used by nature.
The input data can be represented, for example, by the presence of people inside the room, the outside temperature, the inside temperature, the time, the date and other data that are useful to characterise the desired application and so on.
The output data are represented by the desired energy management strategies as a function of the input data installations that act directly on the electrical loads and the air conditioner.
Different genetic algorithm can be used to achieve the desired purpose, each characterised by peculiar features: as the number of inputs and their relations with output data varies, a genetic algorithm is more indicated with respect to the other algorithm. In fact every other management strategy would need a certain artificial intelligence, such as, for example, the one provided by neural networks, whose complexity and the consequent implementation duty grow with the number of input and output variables.
Since the computation resources of the electronic controller that hosts the genetic system is unavoidably limited, it is necessary to reduce as more as possible the number of input data and the complexity of the energy management strategy, that is a genetic algorithm which has to optimise electrical energy consumptions and that can be hosted on a commercial microcontroller, that is the purpose of this paper.

2. GENETIC ALGORITHMS PRINCIPLES

Genetic algorithms represent wide range numerical optimisation methods, that use the natural processes of evolution and genetic recombination.
They can be used in different application fields, thanks to their versatility.
GAs are very useful when the target is to find an approximate global minimum in a high-dimension, multi-modal function domain, in a near-optimal manner.
They can easily handle discontinuous and non-differentiable functions, on the contrary of the most optimisation methods.
The GAs encode each parameters of the problem that must be optimised into a proper sequence (where the alphabet used is generally binary) called a gene, and combine the different genes to constitute a chromosome. A proper group of chromosomes, called population, experience a Darwinian processes of natural selection, mating and mutation, creating new generations, until it reaches the final optimal solution driven by a desired fitness function.
The GA optimisers, therefore, operate according to the following nine points:
1) encoding the solution parameters as genes;
2) creation of chromosomes as strings of genes;
3) initialisation of a starting population;
4) evaluation and assignment of fitness values to the individuals of the population;
5) reproduction by means of fitness-weighted selection of individuals belonging to the population;
6) recombination to produce recombined members;
7) mutation on the recombined members to produce the members of the next generation;
8) evaluation and assignment of fitness values to the individuals of the next generation;
9) convergence check.

The coding is a mapping from the parameter space to the chromosome space and it transforms the set of parameters, which is generally composed by real numbers, in a string characterized by a finite length. The parameters are coded into genes of the chromosome that allow the GA to evolve independently of the parameters themselves and therefore of the solution space.

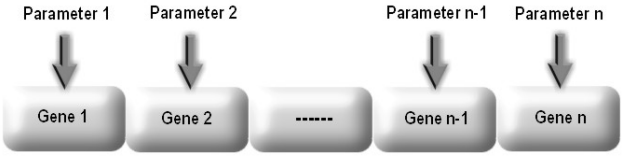

Fig. 1 Encoding of the solution parameters as genes of a chromosome

Once created the chromosomes it is necessary choose the number of them which composes the initial population. This number strongly influences the efficiency of the algorithm in finding the optimal solution: a high number provides a better sampling of the solution space but slows the convergence. A good compromise consists in choosing a number of chromosomes varying between 5 and 10 times the number of bits in a chromosomes, even if in the most of situations, it is sufficient to use a population of 40-100 chromosomes and that does not depend of the length of the chromosome itself. The initial population can be chosen at random or it can be properly biased according to specific features of the considered problem.

Fitness function, or cost function, or object function provides a measure of the goodness of a given chromosome and therefore the goodness of an individual within a population. Since the fitness function acts on the parameters themselves, it is necessary to decode the genes composing a given chromosome to calculate the fitness function of a certain individual of the population.

The reproduction takes place utilising a proper selection strategy which uses the fitness function to choose a certain number of good candidates. The individuals are assigned a space of a roulette wheel that is proportional to they fitness: the higher the fitness, the larger is the space assigned on the wheel and the higher is the probability to be selected at every wheel tournament. The tournament process is repeated until a reproduced population of N individuals is formed.

The recombination process selects at random two individuals of the reproduced population, called parents, crossing them to generate two new individuals called children. The crossover is useful to rearrange genes to produce better combinations of them and therefore more fit individuals. The recombination process has shown to be very important and it has been found that it should be applied with a probability varying between 0.6 and 0.8 to obtain the best results.

The mutation is used to survey parts of the solution space that are not represented by the current population. If the mutation probability overcomes a fixed threshold, an element in the string composing the chromosome is chosen at random and it is changed from 1 to 0 or vice versa, depending of its initial value. To obtain good results, it has been shown that mutations must occur with a low probability varying between 0.01 and 0.1.

The converge check can use different criteria such as the absence of further improvements, the reaching of the desired goal or the reaching of a fixed maximum number of generations.

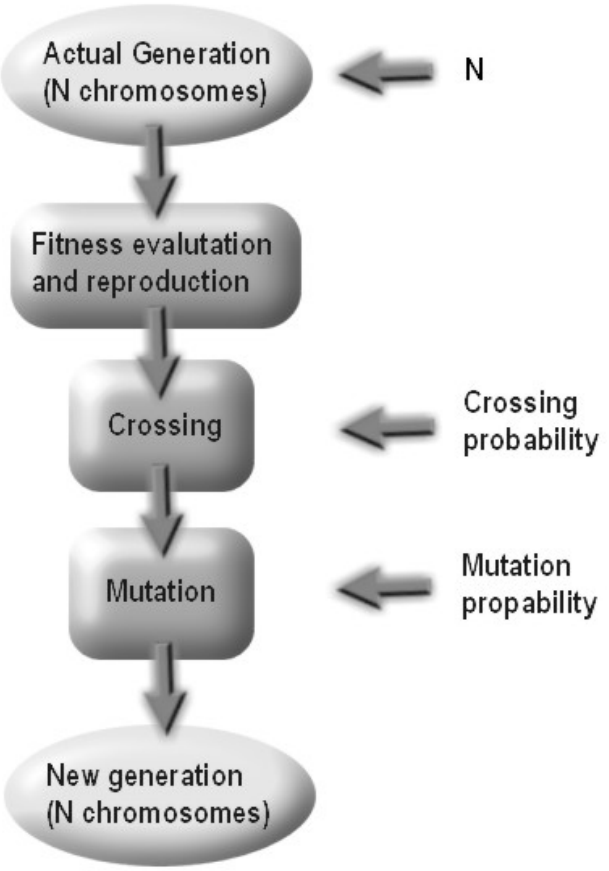

Figure 2: Operative scheme of GA iteration

3. THE GENETIC CLASSIFIER SYSTEM

A classifier system is a machine learning system that learns syntactically simple string rules to guide its performance in an arbitrary environment. A classifier system is composed by three sub systems:

1) rules and messages system
2) apportionment of credit system
3) genetic algorithm.

The rule and message system of a classifier system is a special kind of production system, that is a computational scheme that uses rules as its only learning method. Although there is a wide variation in syntax between production systems, the rules are generally of the form ‘if <condition> then <action>’. The meaning of a production rule is that the action may be taken when the condition is satisfied. Even if this simple device for representing knowledge can seem to be too constraining, it has been shown that production system are computationally complete and also convenient, since a single rule or a small set of rules can represent a complex set of thoughts compactly. Classifier systems restrict a rule to a fixed-length representation. This restriction has two benefits: all strings under the permissible alphabet are syntactically meaningful and fixed string representation permits string operators of the genetic kind, letting possible a genetic algorithm search of permissible rules.

Classifier system use parallel activation whereas traditional expert systems use serial rule activation. During each matching cycle, a traditional expert system activates a single rule. This rule-by-rule procedure is a

bottleneck to increase productivity, and much of the difference between competing expert system architectures concerns the selection of the better single rule activation strategies for this or that type of problem. Classifier systems overcome this bottleneck, allowing parallel activation of rules during a given matching cycle. Thanks to this feature, classifier systems allow multiple activities to be coordinated simultaneously.

When choices must be made between mutually exclusive environmental actions or when the size of the matched rule set must be pruned to accommodate the fixed length message list, these choices are postponed to the last possible moment, and the arbitration is then performed competitively.

In traditional expert systems, the value or rating of rule relative to the other rules is fixed by the programmer in conjunction with the expert group of experts being emulated. In a rule learning system, the relative value of different rules is one of the key pieces of information that must be learned. To facilitate this kind of learning, classifier systems force classifier to coexist in an information-based service economy. A competition is held between classifiers where the right to answer relevant messages goes to the highest bidders, with the subsequent payment of bids serving as a source of income to previously successful message senders. In this way a chain of rules is formed from the input of the system, represented by the detectors, to the output of the system, represented by the actuators. The competitive nature of the economy ensures that good rules, that are the more profitable, survive and bad rules, that are unprofitable, die off.

The apportionment of credit is very important in a classifier system. It uses a sort of internal currency that is exchanged and accumulated to provide a natural figure of merit. Using a classifier's bank balance as a fitness function, classifier may be reproduced, crossed, and mutated, according to the criteria illustrated in the previous paragraphs. Thus, not only can the system learn by ranking extant rules, but it can also discover new possibly better rules as innovative combinations of its old rules.

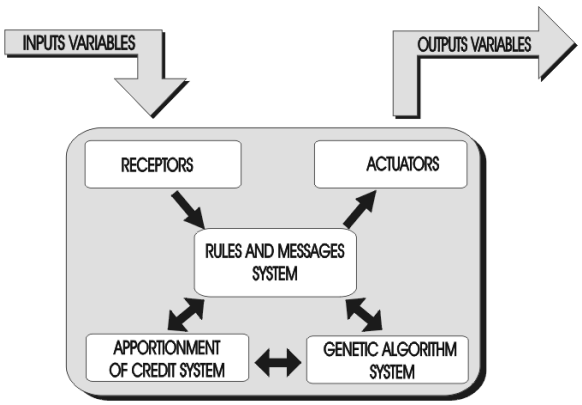

Fig.3 Scheme of a classifier system

4. THE DESIGNED GENETIC SYSTEM

The system is composed by an electronic microcontroller that controls one or more than one room. It is equipped with a certain number of input sensors and a certain number of output actuators, which manage the electric loads and the other energy sources.

The input can be represented, for example, by the presence of people inside a room, the outside temperature, the inside temperature, the time, the date and other data that are useful to characterise the desired application and so on. The output data are represented by the desired energy management strategies as a function of the input data that act directly on the electrical and the air conditioner installations.

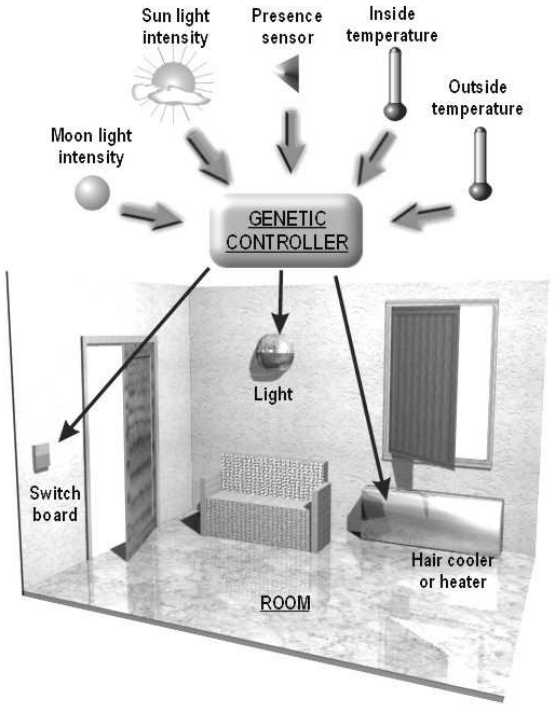

Fig.4 Scheme of the considered genetic system

An entry level system concentrates only on the presence information, switching on and off the electrical loads as a function of the occupation state of the controlled room, that is the system is capable of learning the occupation state and of switching in advance the electrical loads to let the people find a comfortable setting from the environmental point of view. It is also capable of learning when the room has been definitely left, so that it disconnects all the electrical loads. From this point of view the genetic system is capable of greatly reducing the energy consumption, ensuring an optimal comfort inside the controlled room.

The payoff information is represented from the efficiency in energy waste reduction: the higher this number the more the relative controlling rule is enforced, the lower this number the more the relative controlling rule tends to extinct.

The system must learn to predict when the room will be occupied basing on the previous occupancy state and on other parameters, that is to distinguish when it has momentarily left the controlled room from when it has definitely left the same room. This kind of system has already been studied [5], showing high potentialities in energy management.

To extend the potentiality of this kind of system it is possible to introduce a further development represented by the capability of controlling the air conditioning system from the user's comfort point of view, that is to switch the air conditioner in advance or later with respect to a certain fixed time, as a function of the user occupation state of the room, of the desired inside temperature, of the outside temperature and of other environmental parameters. This potentiality allows the system not only to learn the room occupant behaviour, but also the room behaviour from the thermal point of view.

The considered input variables are the room occupation information, the time of the day, the day of the week, the day of the month, the month, the inside temperature, the outside temperature, and the outside light intensity. All the considered variables are thought to be essential in the determination of the occupation state of the room for the most of the use of the considered building (home, office, school, university, factory, museum, hospital, etc.).

The range of these variables together with their binary codification are shown in table 1.

The choice of these variables does not need further explications, with the exception of the room occupation information, that is composed by 144 binary samples corresponding to one sample every ten minutes for the previous 24 hours. This information is coded into the environmental message string as a binary information, where a digit 1 means that the room is occupied and a 0 means that the room is unoccupied.

Considered variable	Variability range	Variable type	Number of bits
Room occupation information	0÷1 (*144)	Binary	144
Time (hour+minute)	0÷24 + 0÷59	Integer + Integer	5+6
Day of the week	1÷7	Integer	3
Day of the month	1÷31	Integer	5
Month	1÷12	Integer	4
Inside temperature	0 ÷+31	Integer	5
Outside temperature	-40 ÷+60	Integer	7
Outside light intensity	10^{-3}÷10^{6}	M*10^{N} (M integer, N integer)	4+4

TABLE 1 Variables composing the message string and relative codification.

It is now necessary to define the command generated by the classifier to execute the switching on or off of the electrical loads and the air conditioners. Since the system must learn to predict at what time it is necessary to activate or deactivate the mentioned devices, the command has the following syntax:

command::=<time>:<load identification>:<load condition>

where the time is coded in the same way of the time coded in the condition message, that is 5 bits for the hour information and 6 bits for the minute information, the load identification is used to act correctly on the desired device (a bit, that allows to represent two values, if it is necessary to operate on an electrical load and an air conditioner device) while the load condition is coded using only 1 bit, that is a 1 when the load is switched on and a 0 when the load is switched off.

Since the learning time of the net depends on the variability of the input data, that is similar input patterns need a low number of rules to be properly recognized while very different input patterns (owed, for example, to great weather variability that produces a great variability of the occupation state of the room) need a quite high number rules to be properly managed, it has been introduced a parameter called "daily presence variability" (DPV) that represents the variation degree between two subsequent days in term of room occupation. It consists, for all the 144 samples points used by the system for our example, in the calculation of the absolute value of the difference between the desired output $O_D(i)$ of the system on the actual day and the desired output $O_{D-1}(i)$ of the system on the previous day, both taken at the same sample time i:

$$DPV = \frac{\sum_{i=1}^{144} \left| O_D(i) - O_{D-1}(i) \right|}{144} . \qquad (11)$$

From the given definition it is evident that if a considered day is characterized by a DPV equal to 1 it is totally different from the previous day (the system must switch on whereas in the previous day it had to switch off and vice versa) while if a considered day is characterized by a DPV equal to zero it is exactly equal to the previous day. The DPV parameter is very useful in characterizing the variability of the input data that strongly influences the learning time and the performance of the net.

The switching error, that is the error made by the system in switching on when it have to switch off and vice versa has already been studied [5] and it will not be repeated here for brevity, where only the results are shown in fig.5.

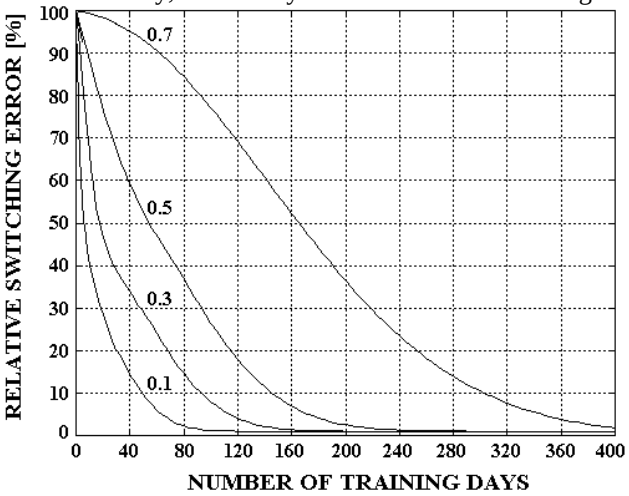

Fig.5 Relative switching error as a function of training days for different values of DPV.

To study the system from the thermal switching point of view it is possible to define a proper temperature error that is used to check the efficiency of the system in switching the air conditioner on exactly in time to let find a temperature that differs by ΔT from the desired temperature when the room occupants arrive. For this reason it is possible to consider a temperature error with margin ΔT defined as the number of times (related to 100 general events) that the temperature inside the room differs more than ΔT with respect to the desired temperature when the room occupants arrive.
The temperature error as a function of the number of training days for different values of ΔT, considering DPV constant and equal to 0.7, has already been studied [6] and it will not be repeated here for brevity, where only the results are shown in the following fig.6. It is possible to see that when the ΔT parameter decreases (higher toleration of temperature error), the number of training days decreases too.

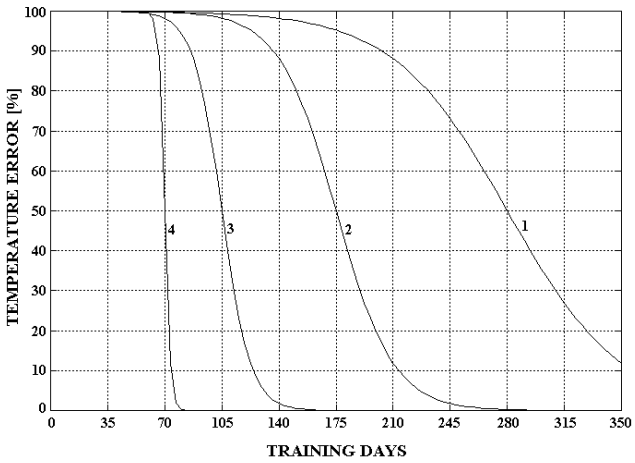

Fig.6 Temperature error as a function of the number of training days for different values of ΔT, for a constant value of DPV equal to 0.7.

5. OPERATIVE PARAMETERS TO DIMENSION THE HOST CONTROLLER OF THE SYSTEM

In this paper we are interested in studying the global performances of the system, that is to define some operative parameters, that allow to choose correctly the microcontroller that host the genetic system from the operative point of view.
The performances of the proposed system have been studied in a simulated-real contest, obtaining interesting and useful results.
Since the system has to perform a certain number of operations, even if they are quite simple due to the structure of the algorithm, it is necessary to define a parameter that considers this factor, to avoid of choosing a too slow microcontroller which is not capable of performing in real time the requested operations, provoking malfunctioning of the whole genetic system. For this reason it has been considered the number of simple operations per second (NSOS) as parameter to check this functionality.
It is obvious that the greater the variability of the presence inside the room (greater values of the DPV variable) and the higher is the number of generated rules at the end of the learning process, which is characterized by a faster generation rate for little variations of DPV and a slower generation rate for large variations of DPV variable.
This means that the system has to perform anyway a greater number of operations at the begin of the learning process and a little number of operations at the end of the learning process.
In the following figure the results obtained for the NSOS parameter are shown.

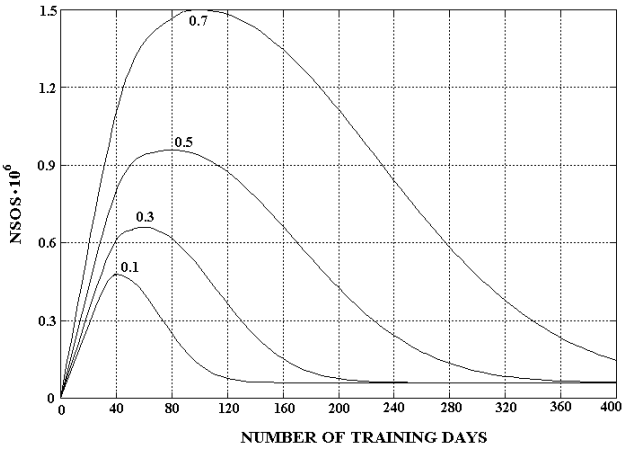

Fig.7 NSOS parameter as a function of the training days for different values of DPV parameter.

It is possible to see that the system executes a higher number of operations in the initial period, when the learning process is more intense and therefore the number of generated rules is greater. Since this number increases with the DPV parameter, to correctly choose the microcontroller which has to host the system it is necessary to consider the worst situation that is represented by the NSOS parameter in the initial days of the learning process for great values of DPV parameter. In this way we are sure that the controller is capable of performing its duty calculations even in the presence of the typical computation overloads that takes place in the initial period.
The obtained number can be used to correctly choose the calculation performance of the microncontroller.
Another important parameter to consider in the choice of microcontroller is the internal memory. In fact the system generates a certain number of rules to characterize and learn the energy optimisation strategy, and this number varies obviously with the DPV parameter: the greater its value and the greater is the number of generated rules, the more little this number and the more little is the number of generated rules.
For this reason it has been considered the number of rules (NOR) as parameter to control the memory occupation and the obtained results are shown in the following figure 8.

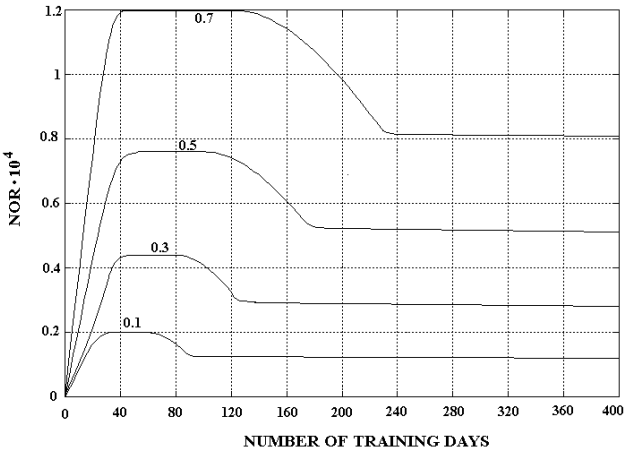

Fig.8 NOR parameter as a function of the training days for different values of DPV parameter.

It is possible to see that the NOR assumes an asymptotic behaviour that is reached after a number of training days that grows with the DVP parameter. It is also possible to see that the asymptotic values of NOR grows with the DPV parameter.
A message, in the considered system, is composed by 187 bits, as it is possible to calculate from table 1, while a command is composed by 13 bits, as it is possible to calculate from its definition given before. Since a rules has the following syntax 'if <message>then<commands>', this means that a rules is composed by 200 bits, equal to 25 bytes.
To calculate the memory occupation it is only necessary to multiply the NOR for the memory occupation of a single command, that is 25 bytes. The obtained number can be used to correctly choose the memory capacity of the microcontroller.
It is interesting to relate the obtained results for the NSOS and the NOR parameters with the results obtained for the relative switching error (RSE) and the temperature switching error (TSE).
First of all it is possible to see that the RSE decreases just immediately after the first training days whereas the TSE decreases after a discrete number of training days. This is due to the fact that the switching on and off of the light is a quite simple operation to be managed with respect to the switching on and off of the air conditioning, where the system has to codify in rules the thermal behaviour of the controlled room, forecasting properly at the same time the presence of occupants inside it. This implies that the NSOS and therefore the NOR continue to grow rapidly even after the first training days, as effectively obtained and shown in the relative figures.

6. CONCLUSION

An optimisation technique for electrical energy consumption based on genetic algorithms has been presented. This kind of system can be hosted on a quite simple electronic microcontroller whose performance, such as the number of operations per second or the memory occupation have been studied in this paper. The proposed system is able of ensuring a consistent energy waste reduction by means of a high flexibility and efficiency in energy management.

6. REFERENCES

[1] F. Garzia, G.M. Veca, "Management and Control of Energy Flows in Building", TELESCON 97, Budapest (Hungary).
[2] F. Garzia, G.M. Veca, "Smart Automatic Control of Energy Flows in Building", DUE 2000, Cape Town (South Africa), 2000.
[3] F. Garzia, G.M. Veca, "Neural Techniques for Energy Management in Building", DUE 2001, Cape Town (South Africa), 2001.
[4] F. Garzia, G.M. Veca, "Advanced Neural Techniques for Energy Management", DUE 2002, Cape Town (South Africa), 2002.
[5] F. Garzia, G.M. Veca, "Evolutionary Computation and Genetic Algorithms for Energy Management and Conservation", Telescon 2002, Montreal (Canada), 2002.
[6] F. Garzia, G.M. Veca, "Energy Management using Genetic Algorithms", Energy and the Environment 2003, Halkidiky (Greece), 2003.
[7] D.E. Goldberg, "Genetic Algorithms in Search, Optimisation and Machine Learning", Addison Wesley, 1989.

7. AUTHORS

Principal Author: Fabio Garzia holds a degree in Electronics Engineering and a PhD in Applied Electromagnetism from the University of Rome "La Sapienza", Italy. He is appointed professor at the degree in Safety and Security Engineering of the same university. He is also responsible for the Safety and Security Engineering Laboratory. His mail address is:

Dipartimento di Ingegneria Elettrica
Università degli Studi di Roma "La Sapienza"
Via Eudossiana, 18
00184 Roma, Italy

Co-author: Giuseppe M. Veca is full professor of Electrical Engineering at the University of Rome "La Sapienza", Italy. He is Senior Member of IEEE and Dean of the PhD in Electrical Engineering. His fields of research are EMC/EMI, Magnetic Devices and Power Apparatus and Systems. His mail address is:

Dipartimento di Ingegneria Elettrica
Università degli Studi di Roma "La Sapienza"
Via Eudossiana, 18
00184 Roma, Italy

Presenter: The paper is presented by Fabio Garzia