Evolutionary Computation And Genetic Algorithms For Energy Management And Conservation

Fabio Garzia, Giuseppe M. Veca, IEEE Senior Member
Dipartimento di Ingegneria Elettrica, Università degli Studi "La Sapienza"
Roma, 00184 ITALY

fabio.garzia@uniroma1.it

Abstract:
A genetic algorithm based strategy that can be implemented in a low cost electronic microcontroller for energy management and conservation application is presented.

It is capable of guaranteeing high performance, adapting and evolving its behavior according to the environmental controlled parameters change.

1.0 Introduction

The manage and the control of the energy flows inside a building can be made using different system architectures[1,2]. Their differences lies on the features and performance and obviously on the overall cost.

Once chosen a particular hardware platform[1,2] it is necessary to realize a proper software that implements and executes the desired energy management policy[3,4].

The choice of the energy policy needs to know 'a priori' the exigencies of the final users together with their energy consumption time table, that is a certain number of data must be collected, in the most of the cases, for a long time.

This problem can be avoided using evolutionary strategies such as the one offered by the genetic algorithms[5].

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 others. 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 therefore whose necessary implementation charge, grow with the number of input and output variables.

It is therefore possible to use very simple genetic algorithms to perform quite simple operations or it is possible to use advanced genetic algorithm to perform complex operation.

Since the computation resources of the electronic module that controls the input data and the output devices are 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 find the most suitable genetic algorithm for this kind of application that has to perform, anyway, an advanced energy management program, which is the purpose of this paper.

2.0 Genetic Algorithms

Genetic algorithms are search algorithms based on the mechanism of natural selection and natural genetics. They combine survival of the fittest among string structures with a structured yet randomized information exchange to form a search algorithm with some of the innovative flair of human search. In every generation, a new set of artificial creatures, corresponding to the strings, is created using bits and pieces of the fittest of the old and an occasional new part is tried for good measure.

The system presented in this paper uses evolutionary computation technology as a basic working mechanism but it operates according to the so-called genetic classifier system principle, that is a system which interacts with the environment where it operates, according to proper desired parameters as inputs and outputs, and which is capable of generating new operative rules learning directly from the reaction of the environment on its action.

We briefly explain, in the following, the principles of GAs and of genetic classifiers.

2.1 An overview of Genetic Algorithms

GAs require the natural parameter set of the optimization problem to be coded as a finite-length string over some finite alphabet. One of the simplest alphabet that can be used is the binary one. Data are therefore organized into strings, that represent the chromosomes from the natural point of view.

A genetic algorithm starts with a population of strings and thereafter generates successive populations of strings that better satisfies the payoff information.

A simple genetic algorithm that yields good results in many practical problems is composed of three operators:
1) reproduction;
2) crossover;
3) mutation.

Reproduction is a process in which individuals strings are copied according to their objective function values f, that is what the biologist call the fitness function, that represents a sort of measurement of profit or goodness that we want maximize. Copying strings according to their fitness values means that strings with higher value have a higher probability of contributing one or more offspring in the next generation.

To reproduce it is only necessary to spin the weighted roulette wheel a number of times that is equal to the number of string of the initial population, to obtain a new generation of strings. It possible to see from fig.2 that the strings characterized by a higher fitness value show a higher percentage of being reproduced in the successive generations. Once a string has been selected for reproduction, an exact replica of the string is made in the new generation group.

Once reproduction has taken place, the algorithm proceeds with crossover. First of all the position of the strings of the new population is altered at random, and then, for each pair of strings, an integer position p along the string is selected uniformly at random between 1 and the length L of the string less one. Two new strings are created by swapping all characters between position p+1 and 1 inclusive.

The third important operation is the mutation, that is very important since even if reproduction and crossover effectively search and recombine existing notions, occasionally they may become overzealous and lose some potential important genetic material. In this sense the mutation operation protects against such an irrecoverable loss, changing, in the considered situation, a 1 to 0 and vice versa, representing a random walk through the string space. It represents an insurance against premature loss of important notions when used carefully with reproduction and crossover operators.

The empirical studied about GAs have demonstrated that the frequency of mutation to obtain good results is on the order of one mutation per thousand of bits position transfers, that are values very similar to the natural ones, demonstrating that mutation is an essential but secondary operator with respect to the one already mentioned.

3.0 Genetic Classifier

Since the genetic controller used for our purposes utilizes as working algorithm a genetic classifier, we explain, in the following some operative concepts necessary to better understand the proposed system.

A classifier system is a machine learning system that learns syntactically simple string rules, called classifiers, to guide its performance in an arbitrary environment. A classifier system consists of three main components:
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. A production system is a computational scheme that uses rules as its only algorithmic device. 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 increased 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 by permitting parallel activation of rules during a given matching cycle. Thanks to this feature, classifier systems allow multiple activities to be coordinated simultaneously.

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.

Apportionment of credit via competition and rule discovery using genetic algorithm form a reasonable basis for constructing a machine learning that is computationally convenient and efficient.

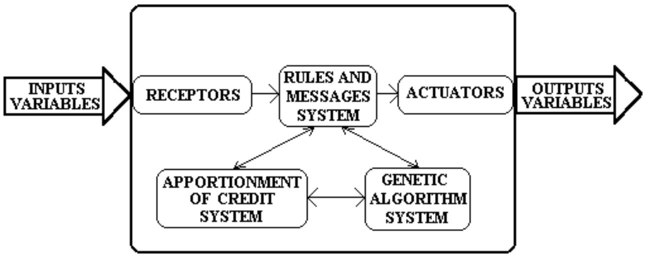

Fig.1 Scheme of a classifier system

3.1 Rule and Message System

The information flows from the environment through the detector where they are decoded to one or more finite length messages that are posted to a finite-length message list where the messages may activate string rules called classifiers. When a classifier is activated, it posts a message to the message list. These messages invoke other classifiers or they may cause an action to be taken by means of the actuators, that represent the system's action triggers. Following these procedures, a classifier combines environmental cues and internal thoughts to determine what the system has to do actually or later, coordinating the information's flow from detectors, to the process zone (message list and classifier store) and to the actuators.

A message of the classifier system is defined as a product, or a concatenation, of M 0's or 1's, that can be written as: <message>::=$\{0,1\}^M$. Messages are the basic tokens of information exchange in a classifier system. The messages sent to the message list may match one or more classifiers or string rules.

A classifier is a production rule characterized by a very simple syntax of the kind: <classifier>::=<condition>:<message>, where the condition is a pattern recognition system whose syntax is of the kind: <condition>::=$\{0,1,\#\}^M$, being # a wild card character.

A condition is matched by a message if at every position a 0 in the condition matches a 0 in the message, a 1 matches a 1, or a # matches a 0 or a 1.

Once a classifier's condition is matched, that classifier becomes a candidate to post its message to the message list on the next time step. The classifier can be posted or not to the message list depending of an activation of auction that depends on the evaluation of a classifier's value or weighting.

3.2 Apportionment of Credit Algorithm

Classifier systems rate individual classifiers according to the classifier's role in achieving reward from the environment. There exists a certain numbers to do it but the most common method is the so called bucket brigade that implements an information economy procedure where the right to trade information is bought and sold by classifier, that form a chain of middlemen between information manufacturer, represented by the environment, to information consumer, represented by the consumer.

3.3 Genetic Algorithm in Classifier Systems

The apportionment of credit by means of bucket brigade algorithm ensures a clean procedure to evaluate rules and to decide between competing alternatives, but it is still necessary to find a way to inject new, and possibly better, rules into the system, that is the reason according which GAs are introduced. These new rules are placed in the population where they are processed using the auction, payment and reinforcement mechanism to properly evaluating their role in the system.

The mutation process must be modified since classifier systems use a ternary alphabet: even if we use the same concept of mutation probability, when a mutation is invoked, the mutating character is substituted by one of the other two with equal probability, that is $0 \rightarrow [1,\#]$ or $1 \rightarrow [0,\#]$ or $\# \rightarrow [0,1]$.

4.0 Description of the System

The considered system is composed by an electronic microcontroller, controlled by a genetic algorithm, a certain number of input sensors and a certain number of outputs, that control the energy actuators.

The inputs can be represented by different parameters that characterise the desired application while the outputs 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.

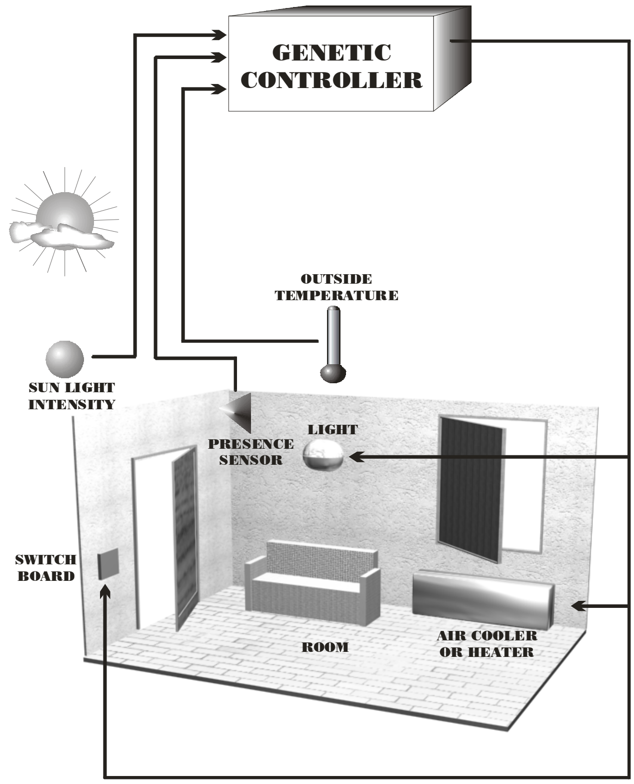

Fig.2 General scheme of the considered genetic system

A first level system concentrates only on the presence information, switching on and off the electrical charges 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 charges 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.

To ensure a high degree of precision it is necessary to extend as more as possible the number of past input presence variables, finding a good compromise between the precision of the system and the data length: good results are obtained if a 24 hour period is used as input that is a total of 144 input samples (24 hour multiplied 6 samples per hour, that is 1 sample every 10 minutes). This information is coded into the environmental message string as a binary information, where a 1 or a 0 in the i-th position means that the room was occupied or unoccupied [(144-i)*10] minutes ago, constituting a sub string of 144 bits.

Other useful information to be coded into the environmental string are represented by the time of the day, the day of the week, the day of the month, the month, 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 time of the day is composed by the hour information, varying between 0 and 24, and by the minute information varying between 0 and 59. The hour information is coded with 5 bits, that can be used to represent integer numbers between 0 and 31 (32 numbers), while the minute information is coded with 6 bits, that can be used to represent integer numbers between 0 and 63 (64 numbers). It is obvious that the system is programmed to ignore the values generated from the genetic algorithm which exceed the allowed real value (24 for the hour information and 59 for the minute information), even if the binary codification allows to represent over this limits. The same controlling operation is made on the values of the other coded information that composes the environmental message string.

The day of the week can be represented using a number varying between 1 (related to Monday) and 7 (related to Sunday). The correspondent values can be represented using 3 bits (8 numbers), excluding the binary 0 that does not correspond to any real day of the week.

The day of the month can be represented using a number varying between 1 to 31. The correspondent values can be represented using 4 bits (32 numbers), excluding the binary 0 that does not correspond to any real day of the month.

The month can be represented using a number varying between 1 to 12. The correspondent values can be represented using 4 bits (16 numbers), excluding properly the binary number that does not correspond to any real month.

The outside temperature can be represented using a number varying between –40 and +60: this range allows to consider the most of climatic situations. The correspondent values can be represented using 7 bits (128 numbers), excluding properly the binary number that does not correspond to any real value of the temperature.

The outside light intensity is a quite different information to be coded since it can vary from 10^{-3} lux in a dark night without moon to 10^{6} lux in a sunny day during in summer, when the sun is at the zenith, that is a range that spans for 9 order of magnitudes. In this case, since it is not necessary a high resolution for this kind of variable, a representation of the kind $M*10^{N}$ has been chosen, where M is an integer varying between 0 and 10, and N is relative number varying between –3 and 6. The M number is coded with 4 bits, that can be used to represent 16 numbers (excluding the unnecessary numbers), while the N number is coded with 4 bits, that can be used to represent 16 integer numbers (properly scaled between –3 and 6, excluding the unnecessary numbers).

Considered variable	Variability range	Variable type	Number of bits
Presence 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
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.

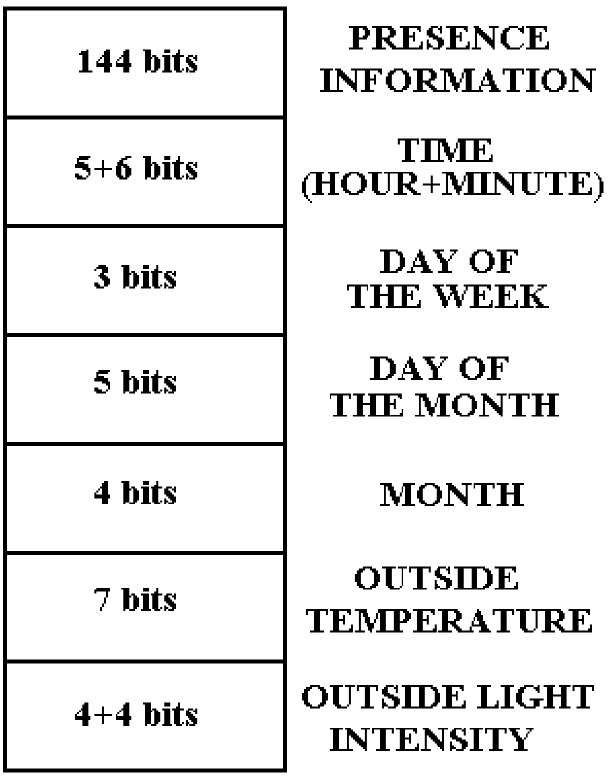

Fig.3a Scheme of the message string.

It is now necessary to define the message generated by the classifier to execute the switching on or off of the electrical loads. Since the system must learn to predict at what time it is necessary to activate or deactivate the mentioned loads, the message has the following syntax: message::=<time>:<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, while the load condition is coded using only 1 bit, that is 1 when the load is switched on and 0 when the load is switched off.

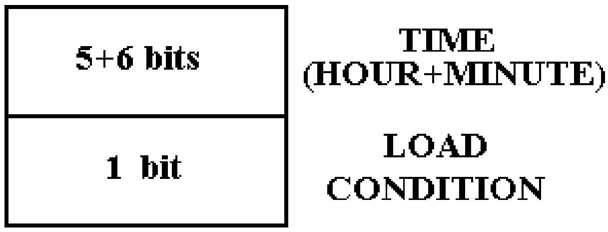

Fig.3b Scheme of the action string.

Since the standard temporal length of the presence variable has been chosen to be equal to 10 minutes, there must be a criterion that decides the binary value of the presence variable in the case of continuously changing of this variable inside the 10 minutes interval which means that the occupant enters and exits continuously. For this reason a proper threshold unit has been used. This unit acquires the instantaneous values of the presence variable and decides, at the end of the period ΔT, if the occupant has remained inside the room more than 50% of time, setting the input variable to the net equal to 1, or less than 50%, setting the input variable to net equal to zero.

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 "day variability" (DV) that represents the variation degree between two subsequent days. It consists, for all the 144 samples points used by the system, 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:

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

From the given definition it is evident that if a considered day is characterized by a DV 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 DV equal to zero it is exactly equal to the previous day. The DV parameter is very useful in characterizing the variability of the input data that strongly influences the learning time and the performance of the net.

4.1 Results

The system has been simulated and tested in a real time modality to study its behavior and its performance. It has been considered, as performance variable, the relative switching error that is the error generated when the system switches on instead of switching off as requested from the occupation state of the controlled room and vice versa.

In fig.5 the relative switching error as a function of the number of training days for different values of day variability (DV) parameter is shown.

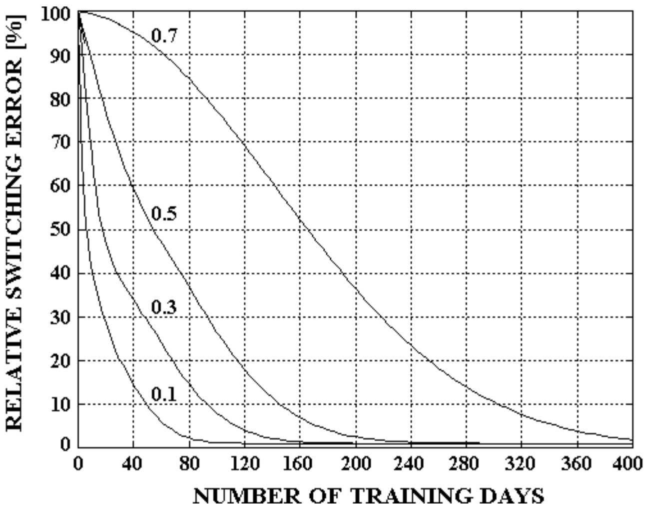

Fig.4 Relative switching error as a function of the number of training days for different values of DV (day variability) parameter. The mutation probability has been chosen to be equal to the optimal value of one mutation per kbit of string information.

It is possible to see that the more the DV parameter increases (that is the variability of the input data increases) and the more the system needs a longer period to be trained since it has to generate a higher number of rules to correctly switch the electrical loads.

In figs.6 the relative switching error as a function of the DV parameter for different values of mutation probability is shown. It is possible to see that the relative switching error increases modestly with the DV parameter when the mutation probability is of the order of 1 mutation per kbit of string information, since this helps in not loosing precious information that reproduction and the crossover could have neglected or not discovered at all. Higher or lower values of this parameter, provoke a not optimal evolution process, not allowing the generation of efficient rules and increasing the relative switching error.

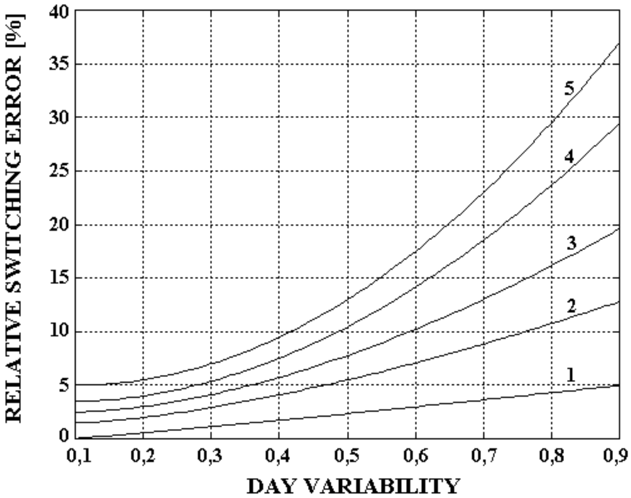

Fig.5a Relative switching error as a function of DV (day variability) parameter for different values (greater than one) of mutation probability, expressed in number of mutations per kbit of string information.

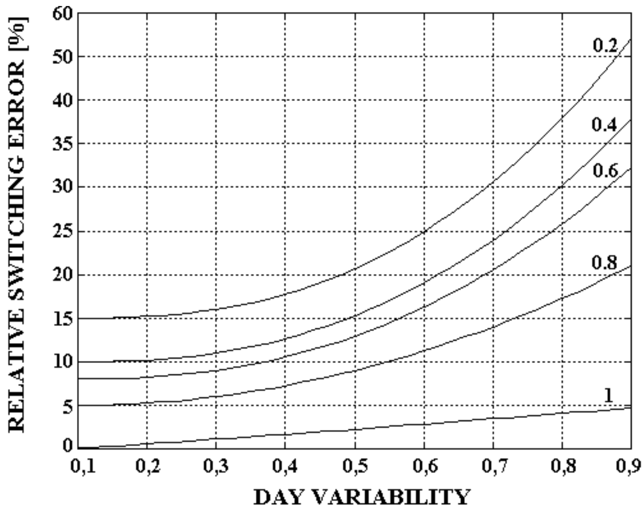

Fig.5b Relative switching error as a function of DV (day variability) parameter for different values (lesser than one) of mutation probability, expressed in number of mutations per kbit of string information.

5.0 Comparison with a traditional system

A traditional system is only capable of switching on or off the electrical loads using information such as date, time or temperature. This means that if the occupants leave the controlled room for a certain time in a regular way, the system does not switch off the electrical loads if it has not been properly programmed, wasting a lot of energy.

The proposed system, on the contrary, is capable of learning in real time, generating continuously new strings that codify the room occupants profile and adapting its energy management strategy to every change of the behavior of the users.

The genetic system is also capable of adapting its management strategy to the variation of the external environment, that is to switch the air conditioner devices as a function of the external system. In fact, basing on the external temperature and on the external light intensity, it is capable of predicting if it is necessary to anticipate or delay the switching on time of the air conditioner with respect to the predicted arrival time of the room occupants to let them find an optimal temperature without wasting energy.

The genetic system is also capable of codifying the insulation condition of each controlled room relating the light intensity and the outside temperature with the internal temperature, the time, the date, the air conditioner working situation: in this way the system generates strings that consider the heat flow generated from the sun irradiation on the room in a hot day or the heat flow from the internal towards the external environment in a cold day.

A traditional system, on the contrary, switches on the air conditioner at a fixed time, without considering variation of external temperature and of the sun condition and without considering the insulation condition of the controlled room, that is the incoming or outgoing heat flows.

A typical microcontroller used for a traditional system is capable of performing simple programming commands such as if <condition> then <action>, or simple arithmetic operations. It is also generally equipped with a quite large on-board memory necessary to store the strings generated by the genetic program to codify all the information. This means that it is not necessary to use dedicated devices, since the most common commercial controllers can support a genetic algorithm without any problem.

We want now give more details about the sensors used by the genetic system, the installation and the cost, comparing them with a traditional system.

The presence sensor used is the same used for intrusion detection in alarm system. The recommended kind of sensor to be used in this situation is the double technology that uses both microwave and infrared to reveal the presence of people. Alternatively it is possible to use the passive infrared sensor. This kind of sensor can be use from both the genetic and the traditional system and its cost is of about 10-20 US$.

The temperature sensor in constituted by a simple termoresistance or a termocouple, that is capable of converting a temperature value into a resistance or a voltage respectively. This output variable is read by the analogic input of the controller. This kind of sensor can be use from both the genetic and the traditional system and its cost is of about 1-2 $.

The light intensity sensor in constituted by a simple photoresistance or a photodiode, that is capable of converting a temperature value into a resistance or a voltage respectively. This output variable is read by the analogic input of the controller. This kind of sensor can be use from both the genetic and the traditional system and its cost is of about 1-2 $, that is just the same of the temperature sensors.

These three sensors are common to the genetic controller and to the traditional controller. Since the genetic controller has been designed to provide more advanced functionalities, such us to predict the occupants behavior using as information the outside light intensity and the outside temperature or to predict if it is necessary to anticipate or delay the turning on or off of the air conditioner basing on the outside or inside environment conditions, it is necessary to install also an external temperature and light intensity sensor. Since these sensors work externally, they are exposed to severe environmental conditions, and therefore it is necessary to use waterproof sensor, whose cost is a bit higher of the normal sensor of about 50%.

We want now give more details about the microntroller. A traditional controller does not need any advanced logic but it just need to perform simple operations basing on an internal timer. These kind of controller are eventually equipped with a proper telecommunication interface to exchange data with the other controllers. Different products are available on the market an their cost is of about 25-50 $.

A controller that can be used as genetic controller needs to perform a bit complex operations, as we already said before, and the market actually offers good products at the cost of about 40-80 $ US, that is about 80% more with respect to the controller used in the traditional scheme.

Considering all the components the cost of the traditional system is of about 37-74 $ while the cost of a genetic system, considering the extra outside sensors, is of about 65-130 $. These cost must be increased with the installation cost and the program cost. The installation cost is quotable around 30% of the components cost, including the cost of the cables, that is 11.1-22.2 $ for the traditional scheme and 19.5-39 $ for the genetic scheme. The program cost is quotable around 10% of the components cost, that is 3.7-7.4 $ for the traditional scheme and 6.5-13 $ for the genetic scheme.

The total costs are of about 51.8-103.6 $ for the traditional scheme and of about 91-182 $.

The extra cost necessary to install a genetic system is totally justified by the great energy saving operated by the system itself.

The installation complexity of the systems is just the same, with the only difference that the genetic system needs two more external sensors (temperature and light intensity) to better predict the behavior of the room occupants and of the environment.

6.0 Conclusion

A genetic algorithm based system for energy management purposes has been presented. It can be implemented in a low cost microntroller and it is able to ensure high performances, learning the behavior of the users and adapting, correspondently, its electrical loads control strategy.

7.0 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] D.E. Goldberg, “Genetic Algorithms in Search, Optimization and Machine Learning”, Addison Wesley, 1989.