Like FL has been introduced as extension of classical boolean logic, allowing to cope with ill-defined systems, incomplete knowledge, noised environment and so on, the peculiarities of FCS are intuitively view as an extension of Cellular Neural Networks and Cellular Automata in solving real world applications.

3. CELLULAR FUZZY SYSTEMS AS UNIVERSAL COMPUTING MACHINES

The power of such systems can be proved, theoretically, in a quite trivial way.
In fact, as it is well-known in the world of logic computing, the Turing's Universal Machine (TUM) theorem states that: "it doesn't exist computation that any computer can do that cannot be done by Turing machines".
Another important result has been obtained in 1982 by Berlekamp et al. [3] concerning with the "game of life". He gave a proof that this simple "game" is equivalent to a Turing machine: then it can be stated that any architecture able to implement the game of life can satisfy the TUM theorem.
This result has been recently used by L.O. Chua et al. in [4] and [5] showing that a Cellular Neural Network (CNN) and, in particular, its CNN Universal Machine (CNNUM) are able to implement Life and therefore they are as Universal as the Turing machine.
The "game of life" algorithm is a first order cellular automata, ideally played on a infinite 2D cell array: it starts from a random distribution of cells with two possible status (dead or alive) that are evolved, during discrete time steps, based on the following rules:

1) if the generic *C(i,j)* cell is alive at time *k* and its *1*-neighborhood contains more than 3 living cells, then *C(i,j)* will be dead at time *k+1 (death by overcrowding);*
2) if *C(i,j)* is alive at time *k* and its *1*-neighborhood contains less than 2 living cells, then *C(i,j)* will be dead at time *k+1 (death by exposure);*
3) if *C(i,j)* is dead at time *k* and its *1*-neighborhood contains exactly 3 living cells, then *C(i,j)* will become alive at time *k+1 (birth);*
4) if *C(i,j)* is alive at time *k* and its *1*-neighborhood contains exactly 2 or 3 living cells, then *C(i,j)* will remain alive at time *k+1 (survival).*

It has been demonstrated that, by evolving from random starting conditions the game of life results in surprisingly complex behavior. After a suitable transient time, up to 15 different patterns can be generated. Each pattern represents a particular cell configuration which can become static, or oscillating with a stable center position, or can move on the whole cell space until it does not interact with another pattern (e.g. the "glider gun" pattern).
The equivalence with Turing machines imply that the predictability of the future state for the cell array, given its initial state, is similar to the halting problem of Turing machines.
In order to establish an extension of the results given by L.O. Chua concerning with CNNs to the field of FCS, some background definition and preliminary assumption, related with the architecture defined in definitions 1-4, are needed

Definition 5
The basic element of a Fuzzy Cellular System implementing a "fuzzy life" algorithm is constituted by a fuzzy cell with continuous state variable $x_{ij}(t)$ representing the health status of the generic individual in the i-th row and j-th column of the whole cell array. The value +1 corresponds to a alive cell with optimal health status, while the value 0.5 corresponds to an alive cell with normal health status and the value 0 corresponds to a death cell; all intermediate values are possible for alive cells with different health state.

Assumption 1 (set *R1*)
Let us assume that the health state of a generic cell is inferred by the state of the cells of the *1*-neighborhood: in particular, a predominance of neighborhood living cells with good health state will produce a worsening of the considered cell state caused by overcrowding, the same result, caused by exposure, will be produced in case of predominance of neighborhood living cells with bad health state, while a generally normal health state of the neighborhood will produce a betterment of the cell state and an increasing of the "survival probability" of the cell. This set of qualitative results are written in the form of fuzzy rules constituting the set *R1*, and it can be formulated by the following equation set:

$$
\begin{aligned}
Nr_{ij}(t) &= \sum_{(k,l)=i-1,j-1}^{i+1,j+1} x_{kl}(t) \\
\mu_k &= \left(x_{ij}(t) \quad is \quad A_k(x_{ij}(t))\right) \quad and \quad \left(Nr_{ij}(t) \quad is \quad B_k(Nr_{ij}(t))\right) \\
R1\_out &= \frac{\sum_{k=1}^{nrules} \mu_k \cdot FS_k}{\sum_{k=1}^{nrules} \mu_k}
\end{aligned}
\tag{1}
$$

being A_k and B_k the fuzzy sets of the cell state and of the neighborhood respectively, while μ_k represents the activation degree of the k-th fuzzy rule in the set *R1* and *R1_out* is the computed crisp-value which can be interpreted as a "survival probability" for the cell being considered.

Assumption 2 (set *R2*)
Concerning with the time evolution of a single cell state, it will be assumed that each cell will dead whenever its "survival probability", or its new health state computed by *R1*, go down a fixed "minimum threshold". On the contrary, when the new state value computed by *R1* go up a fixed "maximum threshold", it is interpreted as a birth of a new cell. Other cases will produce a modification of the cell state by the following equation:

$$x_{ij}(t+1) = x_{ij}(t) + \Delta x \cdot (R1\_out - 0.5) \tag{2}$$

Using such assumption the whole fuzzy processing system is characterized by the fuzzy sets and the rules outlined in Fig.2 and Fig.3, while Fig.4 reports some simulation results.
The main difference with the traditional "game of life" algorithm is concerning with the introduction of a "not binary" health state and consequently on the minimum-maximum thresholds and change-in-state Δx parameters. The value of those parameters, chosen on an appropriate range, influence the dynamic behavior of the whole FCS, not the final patterns, nor their computing capabilities.

Moreover, using the theoretical results outlined in [4] and [5], it is definitely proved that FCS are as universal as the Turing machine. From a theoretical point of view, this means that FCS can implement any computing algorithm, independently of its complexity.
Finally, as an advantage of the approach, to program FCS requires an human operator specifying the rules which govern the evolution and interactions of each elementary computing element (the cell), which, in turn, is a simplified approach with respect to conventional parallel architecture programming.

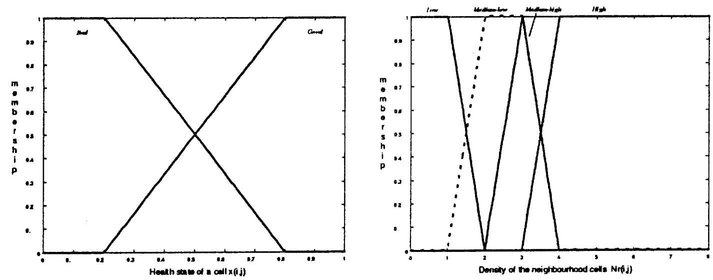

Fig. 2 Membership functions in the IF-part of the R1 set

R1 set:

rule nr.1 (dead by overcrowding):
IF Xij(t) is Good AND Nr(i,j) is High Then FS_1 is 0.0

rule nr.2 (dead by exposure):
IF Xij(t) is Good AND Nr(i,j) is Low Then FS_2 is 0.0

rule nr.3 (birth):
IF Xij(t) is Bad AND Nr(i,j) is Medium-High Then FS_3 is 1.0

rule nr.4 (survival):
IF Xij(t) is Whatever AND Nr(i,j) is Medium-Low Then FS_4 is 0.5

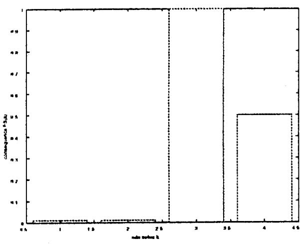

Fig. 3 Rules and parameters in the THEN-part of the R1 set

(a) (b) (c)

Fig. 4 Some simulation results of the "fuzzy life" algorithm:
(a) after 3 time steps;
(b) after 100 time steps;
(c) after 200 time steps.