FRACTALS, NEURAL NETWORKS, CELLULAR AUTOMATA, FORMAL LANGUAGE AND CODING THEORY

Ying Liu
Dept. of Mathematics and Computer Science
Savannah State College
Savannah, Georgia 31404

ABSTRACT

Fractal theory for learning, computer graphics and data compression; Hopfield neural networks for pattern recognition; cellular automata for parallel computation; formal language theory; and coding theory for fault tolerance, data compression, and cryptography ---- what are the common features of these different fields ? In this paper, we will develop a theory to unify all these fields and many other fields. We will show that each of these fields is a piece of a big pie. This unified view will make the understanding of each field deeper and help further development of these fields. This unified view will also open doors for proposing numerous models in various fields.

Key word: Dynamical systems, Fractal, Iterated Function Systems, Neural Network, Cellular Automata, Learning, Cryptography, Data compression, Finite automata.

1. INTRODUCTION

In this paper, we introduce a big "pie", featuring storing information in stable configurations of dynamical systems (DS). The big "pie" has many pieces, including learning, pattern recognition, neural network, finite automata, cellular automata, fractal and iterated function systems, computer graphics, data and image compression, fault tolerance, cryptography, and coding theory.

In section 2, we first introduce three classifications of discrete-time dynamical systems, based on attractor topology [9], the space complexity of parameter space of mapping rules [15], and information capacity [5,15]. The classification based on attractor topology [9] divides dynamical systems into 8 classes. Study of dynamical systems has a long history. It deals with a space, where the object of interest is described by this space, and a law, which describes how the object changes with time or a parameter. Physics is the first science to study dynamical systems. Recently, many types of dynamical systems have been developed for many applications, including pattern recognition [1,2,3,5,14], data compression[4], image analysis[4], computer graphics[4], and many other applications. The membership problem of each classifications will be discussed.

Then in section 3, we will show that Barnsley's iterated function systems [4] is a special case of class 2 dynamical systems based on the attractor classification. IFS has broad applications in computer graphics [4] and image compression [4]. The idea is to store information in a unique stable configuration of dynamical systems. IFS can also be applied to pattern recognition [11]. Any family of class 2 dynamical systems can be used for computer graphics, image encoding and data compression, and pattern recognition. Whether a family of class 2 dynamical systems is practically useful for a specific application depends on the other classifications of the family.

In section 4, we will show that Hopfield neural networks [1,2,3] are class 3 dynamical systems, based on attractor classification. Further more, Hopfield neural networks represents a negligibly small number of class 3 dynamical systems [15]. Hopfield model can be used for pattern recognition[1,2,3]. It is not very successful because of its low information capacity. However, the idea of storing information in stable configurations of dynamical systems used in the Hopfield model is profound and can be used for many other systems. Any family of class 3 dynamical systems can be used for pattern recognition and learning. They can also be used for many other applications, like computer graphics [4], and image analysis, and sometimes, data compression [4,8]. Whether a family of class 3 dynamical systems is practically useful for an application depends on the other classifications of the family.

Wolfram's cellular automata [13] is also a dynamical system. It has been used to simulate complex phenomena with local order. Examples are airplane wings, stock market, and plant growing. Cellular automata defined by Wolfram has very low information capacity and hence is less attractive for many applications. However, their information capacity can be expanded easily and can be used for many applications, including learning and image encoding.

In section 5, we will show that formal language theory [6] can be used as control devices in various dynamical systems. There are systematic ways to introduce finite control to dynamical systems [10]. Introducing finite control systems will increase capacity of a class of dynamical systems [8].

In section 6, we discuss how DS can be used in coding theory. The application of class 2 DS generates a code, macrocode [16]. The application of class 3 DS in coding theory generates another code, attractor code [17]. Macrocode can be used for fault tolerance encoding and data storage. Attractor code can be used for fault tolerance and cryptography.

In section 7, image compression using dynamical systems are discussed [7]. Fractal transformation and fractal equations are introduced. A brief discussion of how to solve a Fractal equation will be discussed. Three different image compression algorithms will also be introduced.

When a dynamical systems is used, different aspects may be focused. Some discussions emphasize the orbits, some discussions emphasize the law of motion, while others emphasize stable configurations. Throughout this paper, we use dynamical systems in only one way, that is, we use stable configurations of dynamical systems to store and process information. We will also only discuss the discrete-time dynamical systems.

2. DYNAMICAL SYSTEMS

In this section, we introduce dynamical systems and related theories. First of all, a dynamical system is described by a state $x \in X$ of the system. Such a state x is also called a configuration of the

0-7803-0720-8/92 $3.00

system [13]. All such configurations together form a space H(X), called a configuration space [13]. Some dynamical systems only use a subspace of H(X). Secondly, an evolution of a system in its configuration space is specified by a set of production rules.

A specification of a dynamical system includes a definition of a configuration space and a set of rules for motions in the configuration space [4,13]. Most of the dynamical system evolutions are irreversible. An orbit of a system is the trajectory in its configuration space. Orbits merge with time, and after many time steps, orbits starting all possible initial configurations become concentrated onto "attractors" [4,13]. These attractors typically contain only a very small fraction of possible configurations. Evaluation of a dynamical system to an attractor from an arbitrary initial configuration realizes data encoding.

Definition Let X be a complete metric space. Then the set of all subsets of X form a space denoted as H(X). [4]

Definition A (discrete-time) dynamical system consists of a configuration space $G(X) \subset H(X)$ together with a mapping $F: H(X) \rightarrow H(X)$. G(X) is closed under F. Formally,

$$D = (G(X), F).$$

When G(X) = H(X), sometimes we simply write

$$D = (X, F).$$

2.1 Dynamical Systems Space [15] and Dynamical System Graph[9]

Theorem Let $|X| = N$, that is, X has N element: $X = \{ 0, 1, 2, \ldots, N-1 \}$, then there exists

$$\Pi = 2^{N2^N}$$

dynamical systems.

The proof can be found in [15]. Let $|X| = N$. There is a systematic way to define all dynamical systems. This will take the space complexity of $O(2^N)$. All dynamical systems can be ordered by a systematic definition, which gives each dynamical system an order.

Definition The canonical order [9] for dynamical systems with $|X|=N$ is defined in the following way. Each dynamical systems is represented by an integer in radix 2^N and the length of the integer is 2^N. These integers can be, in turn, mapped to decimal integers: 0, 1, 2, ..., $2^m - 1$, where $m = N2^N$.

All the dynamical systems together form a space, D. There are $|D| = 2^m$ points in the space D, where $m = N2^N$. The geometric picture of the space depends on the distance between the points in D. A dynamical system can be represented by a graph. For example, let $X = \{ 0, 1, 2\}$, then the dynamical system 0000 0007 is a graph where there is a directed edge from nodes 001, 010, 011, 100, 101, 110, 111 go to node 000, and from node 000 to node 111.

Definition A dynamical system graph [9] is a directed graph defined as follows: Let $|X| = N$, then there are 2^N nodes in the digraph. If state i goes to state j, there will be an edge from node i to node j.

Note that there are

$$2^{|H(X)|^2}$$

digraphs, but there are only

$$|H(X)|^{|H(X)|}$$

dynamical system digraphs. The following results about DS graphs [9] are useful for understanding many applications of this paper.

Theorem The graph of a class 2 dynamical system is a tree where the root has a looped edge connecting to itself. The root is the attractor of the system.

Theorem The graph of a class 3 dynamical system is a forest where each of the roots has a looped edge connecting to itself. The roots are the attractors of the system.

Theorem Let $|X| = N$, there are at least $2^N!$ class 2 dynamical systems and there are at least $2^N!$ class 3 dynamical systems with two point attractors.

2.2 Classification of Dynamical Systems by Attractors

Definition In the attractor classification, the classes and the attractor topology are related as follows:

Class	Feature of Systems
1	Null attractor systems.
2	Point attractor systems.
3	N point attractors, $N < \infty$.
4	N point attractors, $N \rightarrow \infty$.
5	N periodic attractors, $N < \infty$.
6	N periodic attractors, $N \rightarrow \infty$.
7	regular infinite attractors.
8	Strange attractors.

2.3 Classification of Dynamical Systems by Parameter Space

Dynamical systems are specified by { X, F }. F defines the mapping rule. F is specified by a set of numbers. These numbers span a space, the parameter space. Dynamical systems can be classified by their parameter space. Let X have N point. A dynamical system is called O(1) dynamical system if the dynamical system is specified by a constant number of parameters. A dynamical system is called O(N) dynamical system if the dynamical system is specified by O(N) number of parameters.

Definition In the parameter classification, the classes and the space complexity of specifying a parameter space are related as follows:

Class	Number of Parameters
O(1)	O(1)
O(N)	O(N)
$O(N^2)$	$O(N^2)$
.....	
$O(2^N)$	$O(2^N)$

Actually, there are constrains on these parameters. The parameter space is a manifold.

2.4 Classification of Dynamical Systems by Information Capacity

Let X have N elements, then there are

$$n = 2^{N2^N}$$

different dynamical systems. Given a family of dynamical systems, their information capacities are fixed. The information capacity is in turn dependent on the number of different dynamical systems which can be produced by this type.

Definition In information capacity classification, a family can be classified by the number of different systems in the family:

1. $O(2^N)$
2. $O(2^{N^2})$
3. $O(2^{N^3})$

... ...

Systems like O(1), O(N), and $O(N^2)$ are preferred for obvious reasons: they are simple. However, these systems often suffer in their information capacities. There is a trade off between information capacity and the space complexity of the parameter space. For example, Hopfield model has very low information capacity [5].

2.5 Examples

Example IFS [4] and ω-OFA [8]:
In attractor classification, they are in class 2 dynamical systems.
In parameter classification, they are in class O(1).

open question The membership problem of IFS with k parameters in the information capacity classification.

Example Hopfield neural networks[1,2,3]:
In attractor classification, they are in class 3.
In parameter classification, they are in class $O(N^2)$.
In information capacity classification, Abu-Mostafa and Jaques [5] proved that the Hopfield neural network can produce no more than

$$O(2^{N^3})$$

different dynamical systems.

Example Wolfram's cellular automata[4]
In attractor classification, they are in class 5.
In parameter classification, they are in class O(N).
In information capacity classification, they are in class $O(2^N)$.

3. FRACTALS AND CLASS 2 DYNAMICAL SYSTEMS

The basic idea of using class 2 dynamical systems is to store information in the unique attractor of dynamical systems. The Iterated Function System family [4] is the best example. The ω-Orbit Finite Automata family is another example [8].

3.1 A Mapping Between Parameter Space and H(X)

Two spaces are used here, the parameter space T and the data space H(X). Each point t in the space T specifies a dynamical system. A class 2 dynamical system can define data by its attractor. Given a dynamical system, i.e., given a point $t \in T$, we can retrieve the data defined by the dynamical system : $p \in H(X)$. This defines a mapping

$$F: \ T \dashrightarrow H(X)$$

or

$$p = F(t), \quad p \in H(X), \ t \in T.$$

3.2 The Condition for the Second Class Dynamical Systems

Barnsley has shown that if a metric space X, together with a set of contractive mapping is chosen, a single point attractor system will be produced.

Let g be a mapping X ---> X. It can be shown [4] that if g(x) is a contractive mapping in [a,b], g has a unique fixed point in [a,b]

$$g(\alpha) = \alpha.$$

Let x_0 be any value in [a,b], the sequence

$$x_{n+1} = g(x_n)$$

converges to the unique fixed point. Let

$$A = \{ X; \ \Sigma \},$$
$$\Sigma = \{w_1, \ w_2, \ . \ . \ ., w_k \}$$

be an IFS. The transformation

$$W: \quad H(X) \dashrightarrow H(X)$$

is defined by

$$W(B) = \bigcup_{n=1}^{k} w_n(B), \qquad B \in H(X).$$

Barnsley has shown that W(B) is a contractive mapping in H(X) if w_i, i = 0, 1, ... k-1 is a contractive mapping in X. Here contractivity in H(X) means the Hausdorff distances decrease among points in H(X) after a transformation. Therefore, using the above argument of space X in the space H(X), W has a unique fixed point

$$W(A_\infty) = A_\infty$$

Let A_0 be any point in H(X), the sequence

$$A_{n+1} = W(A_n)$$

converges to the unique fixed point. A fixed point is also called a point attractor.

3.3 An Example: Learning and Pattern recognition

Fractal, or class 2 DS in general, can be applied to learning [12]. In this scheme, input vectors are processed in three stages: encoding, internal change, and internal quantization. The first stage is the data encoding, meaning a large input vector is encoded into a relatively small vector which catches

the characteristics of the input vector. The idea of encoding is to store an input vector in an attractor of a dynamical system. The internal space is the parameter space of dynamical systems. The parameter classification of DS, given in the previous section, determines how complex the internal space is for a given family of systems. The second stage involves a system change. The system gradually encodes the input data. Then the output data of this stage is a specification of a dynamical system. In the last step, the internal data organization of the system changes through " quantization". Sample data quantize the internal space. This quantization stores the information which has been learned by a system.

The encoding process chooses a family of class 2 dynamical systems, i.e., choose a parameter space T. The criteria to choose T is such that all vectors in the training set P can be encoded:

$$t_i = f^{-1}(p_i), \quad \forall p_i \in P, \qquad t_i \in T.$$

Encoding Rule: A stimulus is an attractor of a class 2 dynamical system. The internal space of a learning machine is the parameter space of class 2 dynamical systems.

Given an input vector, a system randomly makes a guess about how to encode the input vector, then the system gradually adjusts itself. Eventually, the input vector is encoded into the system.

Changing Rule: The systems use local minima approach to encode stimuli.

The quantization procedure divides the parameter space T into M+1 parts, where M is the number of classes in a training set P. The extra class covers the undefined patterns. After the quantization,

$$T = \bigcup_{i=1}^{M} T_i \cup T'.$$

where T' represents the undefined class of stimuli.

Given a stimulus p, it can be classified by

$$t = F^{-1}(p) \in T.$$

If $t \in T_a$, the pattern p is in class a. If t is in T', or t can not be found, p is in the undefined class.

Quantization Rule: The parameter space T is quantized by the set Q.

4. NEURAL NETWORK AND CLASS 3 DYNAMICAL SYSTEMS

The basic idea of using class 3 dynamical systems is also to store information in the unique attractor of dynamical systems. Hopfield model and BAM are examples [1].

4.1 A Mapping Between Parameter Space and Quantization of H(X)

Two spaces are used here, the parameter space T and the data space H(X). Each point t in the space T specifies a dynamical system. A class 3 dynamical system can define a quantization of H(X). Given a dynamical system, i.e., given a point $t \in T$, we can retrieve the quantization of H(X) defined by the dynamical system. This defines a mapping

$$F: \quad T \dashrightarrow H(X) = \bigcup_{i=1}^{M} H(X)_i.$$

4.2 The condition for the second class dynamical systems

There are two types of conditions to determine the membership problems. For the neural network types, the energy function can be used [1]. For the iterative types, H(X) can be divided into regions and in each region, the condition for class 2 DS can be checked [12].

4.3 An Example: Learning and Pattern recognition

The neural network of the Hopfield model only represents a small fraction of possible systems. However, the idea can be generated to any family of class 3 dynamical systems. In this section, we introduce another learning model, learning using class 3 dynamical systems. When an unknown stimulus is imposed on a system, the system iterates in discrete time steps using a given formula. The system is considered to have converged when output no longer changes on successive iterations. This fixed point attractor determines which class the unknown stimuli belongs.

Encoding Rule: An input vector is an initial configuration of a class 3 dynamical system. The internal data space is the parameter space of class 3 dynamical systems.

Given a set of input vectors, a system changes its internal data starting from the current configuration. The system gradually adjusts itself. Eventually, the stimuli-response relations are encoded into the system.

Changing Rule: The systems use a local minima approach to encode stimulus.

The quantization procedure divides the input vector space H(X) into K parts, where K is the number of attractors of the system. After the quantization,

$$H(X) = \bigcup_{i=1}^{M} H(X)_i.$$

Given a pattern p, it can be classified by If $p \in$

$$p_\infty = F^\infty(p) \in H(X).$$

$H(X)_a$, the pattern p is in class a.
Quantization Rule: The stimuli space H(X) is quantized by the set of attractors of the system.

5. DYNAMICAL SYSTEMS WITH INTERNAL CONTROL

Formal language theory [6] can be used as a control device in dynamical systems [8]. Given a dynamical system, one can systematically add a control device to the system. The details are described in [10]. BAM [1] is such an example.

Definition Let D be a dynamical system

$$D = \{ X; \Sigma \}, \qquad \Sigma = w_1.$$

The one-state automata dynamical system generated

from D is a new family of dynamical systems

$$D = \{X, \Sigma\},$$

where Σ is a set of mapping rules

$$\Sigma = \{w_1, w_2, \ . \ . \ . \ , w_n\}.$$

The dynamical systems evolve by iteration of the mapping

$$\Sigma(B^{(t)}) = \bigcup_{i=1}^{n} w_i(B^{(t-1)}), \quad B \in H(X).$$

Definition The automata dynamical system generated from D is a new family of dynamical systems

$$A = \{ R; \ \Sigma; \ M; \ I; \ F \}$$

where(1) R is a finite set of states and X_i's are metric spaces. (2) Σ is a set of mapping alphabet, which is similar to the case of an LM1 system of D. (3) M is an (nxn) transition matrix. Each element of this matrix, M_{ij}, is a subset of Σ. The transition rule is

$$(X_j, \ w) \rightarrow X_i, \quad if \quad w \in M_{ij}.$$

The set M_{ij} can be empty. (4) The initial state is I = $\{X_I\}$ which takes input vectors. (5) The set of final states F is a subset of R which determines the attractors.

This approach of generating a dynamical system from a dynamical system has been used for IFS [8], neural network[10], and cellular automata[10]. It can be used in any dynamical systems.

6. CODING THEORY

In this section, we briefly describe how *DS* can be used in coding theory. Two different types of codes, macrocode and attractor code, are introduced. Macrocode can be used for error detection. Attractorcode can be used for fault tolerance and cryptography.

6.1 Macrocode[16]

There are many reasons for wanting to change the form of digital data. Entropy is defined as average information per message [18]. Given N bits, there are 2^N messages. If various messages are equally probable, the entropy is N, and the message is represented by no less than N bits[18]. However, in general, this rarely happens. For a given application, only 2^M messages out of 2^N are most likely to appear. This reduces the number of bits per message from N to M.

For a specific application, if most of all the 2^M messages share some common features and if these features can be caught by a class 2 dynamical system, then a class 2 dynamical system can be used to encode the information. Macrocode theory[16] is developed for this reason.

Macrocode can detect and thereafter possibly correct errors efficiently in mass memories. The macrocode features encoding a large amount of data into a separate code and detecting, sometimes even correcting multiple and symmetric errors in mass memories. The macrocode detectability of multiple and symmetric errors is upper bounded to a block size of mass memories (usually 512 - 4K bytes).

The macrocode is represented by Mnd, where M is the information bytes, N is the specification of a dynamical system, and d = { d1, d2}. d1 and d2 are the Hausdorff and the Hamming distances between M and the attractor of N, respectively. This code has error detectability. If d = (0,0), this code also has error correction ability. If d, obtained in the encoding stage, is not (0,0), faults with any number of bits involved can be detected only if the fault changes either the Hausdorff or Hamming distance. The ability of fault detection will be increased if we allow more distances to be included.

6.2 New Cryptosystems [19]

Dynamical cryptosystem [19] features storing information in stable configurations of dynamical systems but representing the information in arbitrary configurations of dynamical systems. The development of this scheme is based on the theory of class 3 dynamical systems. A string p in a plaintext can be considered as a stable configuration of a dynamical system. The encoded string of p in a ciphertext is an arbitrary configuration which will lead the dynamical system to the attractor defined by the original string p. The encoding procedure treats a string in a plaintext as an attractor of a class 3 dynamical system, and generates a configuration randomly, as long as the configuration leads to a correct attractor. The process of decryption is to generate attractors of a dynamical system from the configurations contained in the ciphertext.

Example Neural Cryptosystem

Neural cryptosystems feature storing information in stable configurations of neural networks but representing the information in arbitrary configurations of neural networks. A message, m, in a plaintext can be considered as a stable configuration of a neural network. The encoded string of m in a ciphertext is an arbitrary configuration which will lead the neural network to the attractor defined by the original message m. The encryption procedure, C = E(m), treats a message m in a plaintext as an attractor of a neural network, and generates a configuration C randomly, as long as the configuration, C, leads to a correct attractor, m. The decryption procedure regenerates the attractor, m = D(C) = D(E(m)), of the neural network from the configuration, C, contained in the ciphertext.

If the key cryptosystems are called first level systems, meaning operations are carried out in the space X, the new system described above are called the second level systems, meaning operations car carried out in the space H(X). The third level systems will live in the space H(H(X)).

6.3 Attractorcode Theory For Fault Tolerance [17]

Attractorcode is implemented by storing data in attractors of dynamical systems for fault tolerance. In this coding theory, there is no obvious error detection and correction procedures. The error detection and correction are implemented by additional hardware or software at both ends of a channel. The development of the attractorcode is based on the theory of class 3 dynamical systems. The attractorcode uses the attractors of dynamical systems as encoding devices. If a code p is changed to a code q because of some errors, the code q represents exactly the same data as the original code p, as long as the erroneous code q stays in the same attractor basin centered at the attractor p. This means that only errors with large Hamming or Hausdorff distances between p and q can result in the original code p to leave the attractor basin and hence alter the data contained. Errors with small

distances between a code and its erroneous code are automatically tolerated in the system.

When a dynamical system has a set of point attractors, it can be used to quantize the code space [11]. Given a set of strings, we try to find a dynamical system such that the attractors of the system is the given set of strings, or can be used to represent the given set of strings. The dynamical system inference problem is to infer a dynamical system whose attractor basin structure fits the given set of strings. The encoding process for a given dynamical system is simply to replace the original strings by attractors of a chosen system. The attractors assigned to a given set of strings are called the attractorcodes of the strings. The decoding process evolves the chosen dynamical system into its attractors with the given attractorcode as initial conditions, then decodes the attractor. The degree of fault tolerance measured in the Hamming distance or the Hausdorff distance depends upon the structure of attractor basins of a chosen system.

6.4 Digraph Cryptosystems

A direct generalization of the above cryptosystem is "digraph cryptosystem". As we have discussed in section 2.1, the dynamical system graphs is a subset of the set of all digraphs. A digraph cryptosystem is a digraph with many sinks. Each sink is a clear-text. Any node leading to it may be its cryptogram. The whole picture of a dynamical system is a forest.

7. IMAGE COMPRESSION

In this section, we will introduce fractal transformation and fractal equation. We will also show how learning can be applied to image compression.

7.1 Fractal Transformation

IFS, or class 2 dynamical systems in general, defines the fractal transformation:

F: T ---> H(X)

or

p = F(t), p ∈ H(X), t ∈ T.

The inverse of the mapping defines Fractal data compression.

7.2 Fractal Equations

Given an image p ∈ H(X), one can find a dynamical system

$$A = \{ X; \Sigma \}$$

such that the attractor of this dynamical system is the given image p, therefore, the image, p, is compressed into a specification of a dynamical system. This defines the first Fractal equation

$$h_d (p, F(t)) = 0.$$

where h_d is the Hausdorff distance [4]. It turns out that it is time consuming to solve this equation. However, this equation can be converted to another form. Since the unique point attractor is stable under dynamical system transformation, the second fractal equation is

$$h_d (p , \Sigma_t(p)) = 0.$$

where $\Sigma_t(p)$ is the mapping of p defined by the class 2 dynamical system. Solving the second equation is equivalent to solving the first one, but it is much easier to solve the second one than the first.

Solving the fractal equation is an optimal problem. Several algorithms for solving the fractal equations are listed in [11]. Commercial software can be purchased from Barnsley's company.

7.3 Image Compression

There are several image compression methods related to dynamical systems:

(1) Solve a fractal equation for class 2 dynamical systems[7];
(2) Solve a fractal equation for class 3 dynamical systems[7]; and
(3) Vector quantization.

A. Fractal Equation for Class 2 Dynamical Systems.
The discussion in section 7.2 is this type.

B. Fractal Equation for Class 3 Dynamical Systems.
The class 3 dynamical systems have more than one attractor. To store an image in an attractor, both the specification of a system and an initial configuration are required to specify an image. The initial configuration is usually a simple pattern which can be specified by an integer. Given an image p ∈ H(X), one can find a dynamical system and an initial configuration q such that the attractor of this dynamical system, with q as the initial configuration, is the given image p. Therefore, the image, p, is compressed into a specification of a dynamical system, together with an integer.

This defines a mapping

F: T x N ---> H(X)

or

p = F(t,i), p ∈ H(X), i ∈ N, t ∈ T.

The Fractal equation is

$$h_d (p, F(t,i)) = 0.$$

C. "Vector Quantization"

Class 3 dynamical systems quantize the space

$$H(X) = \bigcup_{i=1}^{K} H(X)_i$$

In a subspace space $H(X)_i$, there are configurations which can be specified easily and there are configurations which can not be specified easily. Let all images in $H(X)_i$ be represented by the attractor of this region. This attractor can be specified by a simple image used as initial configuration.

Given an image, p, it can be used as an initial configuration. The evolution of the dynamical system with the given image, p, as initial configuration will lead to an attractor, say q. There is a simple image, q0, associated with the attractor, q. q0 also leads the dynamical system to the attractor q. Therefore, the given image, p, is compressed to the simple image, q0, in the encoding phase. In the decoding phase, q0 will be used to retrieve the attractor q. The difference between p and q defines the compression torture ratio. The average compression ratio is obtained by comparing p and q0. This is another version of vector quantization, which has been broadly used in commercial packages.

8. CONCLUSION

There are several generalizations of the above theory. (1) The idea of point attractor discussion can be generated to cyclic attractors easily, therefore, class 5 dynamical systems can also be used for many applications specified in this paper. (2) Dynamical system graphs can be generated to digraphs. A such example is given in section 6.4.

Unification theory of everything is always attractive. Greeks invented the word "atom" 2000 years ago in seeking a unified view to understand nature. This paper attempts to unify many different approaches in many different fields. In time where the number of theories is growing everyday, converging the number of theories is an important *method to put information under control*. For this reason, a theory of theories is developed in this paper.

I would like to thank my wife Gina Liu for proof reading of the paper and correcting numerous grammar mistakes.

REFERENCES

1. Bart Kosko, Neural Networks and Fuzzy Systems, Prentice hall, 1991.

2. T. Kohonen, "An introduction to neural computing," Neural Networks, Vol.1, pp.3-16, 1988.

R. Lippmann, "An introduction to computing with neural nets," IEEE ASSP MAGAZINE, APRIL, 1987, 4-22.

3. J. J. Hopfield, "Neural networks and physical systems with emergent collective computation abilities," Proc. of Nat. Academy Sci., USA, vol. 81, pp.3088-92, 1984.

4. M.F. Barnsley, Fractals Everywhere, Academic Press, 1988.

M.F. Barnsley, The Desktop Fractal Design Handbook, Academic Press, 1989.

M.F. Barnsley, et. al., Construction Approximation, V5, p1, 1989.

5. Y. S. Abu-Mostafa and J. St. Jaques, " Information capacity of the Hopfield model," IEEE Trans. Inform. Theory vol. IT-31, pp.461-464, 1985.

R. J. McEliece, et al., "The capacity of the Hopfield associative memory," IEEE Trans. Inform. Theory vol. IT33, pp.461-482, 1987.

6. J.E. Hopcroft and J.D. Ullman, Introduction to automata theory, language, and computation complexity, Addison-Wesley, 1979.

L. Boasson and M. Nivat, Adherence of languages, Technical Preprint, Universite Paris 7, Uer De Mathematilques, 1979.

7. Ying Liu and Hede Ma, " Comparison between image analysis using class 2 and class 3 dynamical systems," To appear Proc SPIE, Vol. 1657, 1992.

8. Ying Liu and Hede Ma, "ω-Orbit finite for data compression," Proceeding of IEEE Data Compression Conference '91, pp. 165 - 174, 1991.

9. Ying Liu, "Classification of dynamical systems and its application in computer vision," To appear Proc. SPIE on Vision Geometry, Boston, 1992.

10. Ying Liu and Hede Ma, "A parallel pattern recognition approach", Proceedings of Fourth ISMM International Conference on Parallel and Distributed Computing and Systems, Washington, D.C., pp 255 - 261, October 8 - 11, 1991.

11. Ying Liu and Hede Ma, "Pattern recognition using ω-Orbit Finite Automata," Proc SPIE 1606, pp. 226 - 240, November, 1991.

12. Ying Liu and Hede Ma, "A comparison of two learning philosophies," Proceedings IEEE Southerncon 1992, pp. 247 - 254.

13. S. Wolfram, "Universality and complexity in cellular automata", Physica 10D, (1984) 1-35.

14. S. J. Perantonis and P. J. G. Lisboa, " Translation, rotation and scale invariant pattern recognition by high-order neural network of moment classifiers," IEEE Tran. Neural Networks, Vo. 3, pp 241 - 251, Mar, 1992.

15. Ying Liu, "Two pattern recognition algorithms using dynamical systems," To appear IEEE international conf. on Intelligent robots and systems ' 1992.

16. Ying Liu and Hede Ma, " A simulation of Macrocode for error detection and correction in mass memory," To appear in Sim Tec' 92, Houston.

17. Ying Liu, "Attractorcode for fault tolerance in data communication systems," To appear ISATED Int'l Conf. on Reliability, quality control and risk assessment, Washington D.C., 1992.

18. See for example, M. S. Roden, "Digital and data communication systems," Prentice-Hall, 1982.

19. Ying Liu and Hede Ma, " A simulation of a new cryptosystem using finite dynamical systems," To appear in Sim Tec '92, Houston.