DEMO 2011/1

ΠΟΛΥΩΝΥΜΙΚΕΣ ΑΝΑΠΑΡΑΣΤΑΣΕΙΣ ΤΩΝ ΑΚΕΡΑΙΩΝ ΚΑΙ ΠΡΩΤΟΓΕΝΗΣ ΑΥΤΟ-ΟΜΟΙΟΤΗΤΑ ΣΤΗΝ ΙΕΡΑΡΧΙΑ ΤΩΝ ΚΑΘΟΛΙΚΩΝ ΛΕΞΙΚΩΝ.

Θ. Ε. ΡΑΠΤΗΣ

POLYNOMIAL REPRESENTATIONS OF THE INTEGERS AND PRIMORDIAL SELF-SIMILARITY IN THE HIERARCHY OF UNIVERSAL LEXICONS.

T. E. RAPTIS

ΕΚΕΦΕ ΔΗΜΟΚΡΙΤΟΣ

Τ.Θ. 60228, 153 10, ΑΓ. ΠΑΡΑΣΚΕΥΗ (ΑΘΗΝΑ) ΕΛΛΑΣ

"DEMOKRITOS"

National Centre for Scientific Research

P.O. BOX 60228, 153 10, AGIA PARASKEVI (Athens) GREECE

ΠΟΛΥΩΝΥΜΙΚΙΕΣ ΑΝΑΠΑΡΑΣΤΑΣΕΙΣ ΤΩΝ ΑΚΕΡΑΙΩΝ ΚΑΙ ΠΡΩΤΟΓΕΝΗΣ ΑΥΤΟ-ΟΜΟΙΟΤΗΤΑ ΣΤΗΝ ΙΕΡΑΡΧΙΑ ΤΩΝ ΚΑΘΟΛΙΚΩΝ ΛΕΞΙΚΩΝ

Θεοφάνης Ε. Ράπτης

ΕΚΕΦΕ «ΔΗΜΟΚΡΙΤΟΣ»

Διεύθυνση Τεχνολογικών Εφαρμογών

rtheo@dat.demokritos.gr

POLYNOMIAL REPRESENTATIONS OF THE INTEGERS AND PRIMORDIAL SELF-SIMILARITY IN THE HIERARCHY OF UNIVERSAL LEXICONS

Theophanes E. Raptis

NCSR "DEMOKRITOS"

Division of Applied Technologies

rtheo@dat.demokritos.gr

Αθήνα, 17 Μαρτίου, 2011

Athens, 17 March, 2011

Polynomial Representations and Primordial Self-Similarity in the Hierarchy of Universal Lexicons

T. E. Raptis†

†Division of Applied Technologies

National Center for Science and Research "Demokritos"

Patriarchou Gregoriou & Neapoleos, Ag. Paraskevi, Athens, Greece, 60228

Email: rtheo@dat.demokritos.gr

A set of fundamental objects is presented that facilitates derivation of some new results with special interest in a variety of topics including Chu spaces, dynamical systems, symbolic dynamics and the theory of polynomials. Three alternative representations of the power set of binary patterns in their associated exponential intervals are presented in terms of polynomials and a natural conjecture on their fractal structure is deduced. Practical applications in Automata theory and Digital Signal Processing are proposed based on special functions defined on the new representation.

1. Introduction

Almost a century after the collapse of the initial proposal of Hilbert for a complete mechanization of mathematics, people turn their attention back to automated theorem proofs and high accuracy numerical calculations for inspiration and new, possibly counterintuitive formulas.

Visualization as a tool for trivialization of proofs guided by intuition becomes attractive nowadays due to the enormous computational power accumulated and modern parallelization techniques [1, 2, 3].

Despite that, the basis of all of our modern computing power is still based on the notion of a Turing machine and Allonzo and Church's λ-calculus [4, 5]. The question is then whether an alternative formulation that could possibly help to overcome limitations of the present paradigm can be revealed using experimental mathematics.

In the present article, a new way of treating computations is established that combines at once the power of induction together with the power of visualization. It is based on the fact that powersets of n-ary symbolic alphabets are closely linked with n-ary trees which can then be deployed in the form of special matrices that form a recursively enumerable set which will here be known as the *"Hierarchy of Universal Lexicons"*.

Moreover, a similar construct if endowed with special memebership functions automatically becomes a Chu Space [6, 7, 8], with the set of symbols (or n-ary cyclic states) being isomorphic to points and the set of integer representations being the set of states. A fundamental fact that implies a deeper connection with other more abstract and general theories that may affect the whole of mathematics in the same way that *Topos* theory already attempts to do. Even more interesting for physical theories is the connection with Quantum Mechanics provided by Pratt in [8], recently reintroduced by Gregori in [9, 10] albeit without mentioning their inherent connection.

While the present work is of an introductory character with respect to the various branches of possible formal development, the possibility of merging the two parts of the analog/digital dichotomy becomes evident at an early stage thus leading to possible alternative realizations of universal computing machinery different form the original Turing paradigm. We also present examples where visualization reveals recursive and self-

similar structures. Similar results have recentl appeared in Kovalinka [20] but the present work attempts to shed some light on their origin that should be traced back to the primordial self-similarity of the representation of the integers as revealed through the lexicon constructs.

In the next section we introduce the necessary definitions, while in section 3, we present some interesting examples of applications in the theory of Automata. In section 4, we associate the recursive nature of the previously defined objects with a particular class of discrete dynamical systems of vast generality. In section 5, we study the connections with the theory of roots of polynomials and provide some possible alternative measures of complexity of possible use in DSP and discrete statistics problems.

2. The Hierarchy of Universal Lexicons

In this section we introduce some necessary tools and preliminary definitions that reformulate the context of symbolic series in any possible alphabet in a manner that allows simultaneous treatment of every possible case. By this we mean that any possible sequence either automatic or random for any possible length can be treated as a member of a special hierarchy of perfectly ordered objects (matrices) so that certain properties can be treated on an equal footing no matter what the origin of the particular sequence was.

The fundamental notion is that of the geometric representation of the powerset of a set of symbols forming an alphabet which coincides with the set of the integers through the polynomial representation. This representation which in the binary case is used for the construction of truth tables and more general of the so called “factorial designs”, has a natural self-similarity

inherent in the polynomial representation of the integers in any possible alphabet, albeit simpler than that of the Cantor set.

To explain the logic behind this design, assume the usual notion of a truth table in a binary alphabet. It is known that the rows of any such table are equivalent to the outputs of a binary counter or equivalently, a set of "clocks" or square oscillators that span an exponential interval [0, 2^N-1] where N is the number of oscillators. The phases of each oscillator get intermixed as they pass from the first row of all zero states to the final state of all ones state which always corresponds to the integer 2^N-1.

The same principle can be applied to any arbitrary alphabet in base *b* for all integers in an exponential interval [0, b^N-1] by taking a set of "*ramp*" instead of square oscillators thus forming a b-ary counter. We remind that every such ramp oscillator can be defined in continuous time through the family of functions

$$y(t) = \left[\frac{t}{\tau\sqrt{b}} \right] \bmod b \qquad (1)$$

In the above "[ ]" denotes the floor function. It can be verified that (1) is periodic with period $\tau\sqrt{b}$ and by taking $\tau_n = b^{n-1/2}$ and restricting the index *t* to the set of the integers we can define an "*extraction*" function in the form

$$y_n : [0, b^N - 1] \to [0, b-1] : y_n(k) = \left[\frac{k}{b^n} \right] \bmod b, \qquad (2)$$
$$0 < n < N, 0 < k < b^N - 1$$

The above simply extracts each symbol from the polynomial representation of any integer *k* in an exponential interval and assigns this symbol value to the associated matrix element (*n, k*).

Any such set of oscillators creates a table which automatically inherits the primordial self-similarity of integer representations and which comes from the mixing of the oscillator states. We

then formalize these observations by defining a hierarchy of $\mathrm{N} \times b^N$ matrices which is both recursive and recursively enumerable due to the exact, 1-1 correspondence of each and every column with the integers through the polynomial representation.

We choose to call the set of these matrices, the Universal Lexicons of order N in base b and write L_b^N. The matrices form a natural hierarchy in the sense that addition of a single oscillator adds a single row in every previous matrix with each symbol appearing exactly *b* times thus leading from the $\mathrm{N} \times b^N$ to the $(\mathrm{N}+1) \times b^{(N+1)}$ member of the hierarchy for which we can write

$$L_b^N : L_b^1 \subset L_b^2 \subset \ldots \subset L_b^n \subset \ldots \qquad (3)$$

An example of such a matrix for the 4-ary alphabet is shown in Fig. 1 and 2 standing for the $L_2^7 - lexicon$ and the $L_4^7 - lexicon$ respectively.

We now briefly mention the three fundamental decompositions of any such matrix based on three fundamental symmetries inherent in all members of the hierarchy.

> *A)* ***Complementation:*** *every Lexicon matrix is decomposable into exactly b bands for which every element of each row of the previous band is mapped to every element of the next band under the complementation operation that carries its symbol to the next mod b.*

> *B)* ***Reflection:*** *every Lexicon matrix admits decomposition into three distinct sets* $S \cup P \cup \overline{S}$ *where S is a generating set,* $\overline{S}$ *is its mirror image under reflection with respect to the first bottom or last top row and P is a set of Palindromes (self-mirroring strings).*

C) ***Cyclic Permutation:*** *every Lexicon matrix admits a decomposition into a special number #(b,n) of distinct cyclic permutations groups* $\{G_i\}$ *of which the set of generators defined as the first element of each set* $\{g_1^i\}$ *is sufficient to reproduce the whole matrix. No known analytical expression was found for #(b,n) as yet.*

We next examine some implications of the above for generic automata.

3. Theory of Automata

An important property of the ramp oscillators allows for another representation of symbolic alphabets above the binary one, to the complex roots of unity. Thus we give the following lemma

Lemma 1.1 (Complex Representation of Symbolic Alphabets): *every symbolic dynamics realized in the integer interval* $[0,\ b-1]$ *through a generic map* $\phi : N \to [0, b-1]$ *is isomorphic to a permutation of a discritized harmonic oscillator states through the correspondence of the ordered set of symbols in the interval* [0, b-1] *to the set of the complex roots of unity of order b given as* $\{\exp[2\pi i((n-1)/b)]\}_{n=0}^{b-1}$.

This is based on the simple fact that the steps of the evolution of any cyclic procedure like the ramp oscillator introduced by (1) and (2) are isomorphic to a set of equidistantly sampled points of a harmonic oscillator of amplitude 1 or, equivalently to a set of equidistantly sampled points from the unit complex circle. The above also allows making contact with the theory of polynomials with complex coefficients for any b-ary alphabet. Additionally, it gives the opportunity to transcribe certain processes in higher alphabets into sampled states of continuous oscillators. For some of the examples mentioned below this

might also mean the possibility of alternative universal analog computer architectures.

While self-similarity is evident in fig. 1 and 2, a less trivial result exists for all functions that can be defined as bit-wise or more general as symbol-wise functions of many variables. We then have the following theorem

> ***Theorem 1.*** *Let A be the set of all arithmetic functions* $f_A : N \to N$ *and let B be the set of all bounded partial symbol-wise functions*
>
> $f_B : S_1 \times S_2 \times ... \times S_M \to S_{M+1}, S_i \subset N$.
>
> *Then* $B \subset A$.

The proof is elementary. Given any function of many variables over the integers $f(x_1,...,x_M)$ we restrict the DoD of each variable in the interval [0, *b-1*]. It then suffices to use the sequence of extraction functions (2) so that

$$f \circ y_n \to f_B(y_1(k),...,y_M(k)) \cong f_B(k),\ k \in [0, b^M - 1] \tag{4}$$

This is equivalent with the concatenation operation $x_1 \| x_2 \| ... \| x_M$ where "||" stands for the concatenation operator $x \| y = x + yb^{[\log_b(x)]+1}$ and by which the set of variables $\{x_i\}_{i=1}^{M}$ is mapped to a unique integer in an exponential interval for any possible combination. This emphasizes the point that all partial multidimensional functions over the integers are essentially "one-dimensional".

There is an important application of the above with possible technological applications in the case of ordinary automata the simplest of which are Turing machines. We examine such a case taking as an example the recent finding of the simplest (2,3)-UTM without an Halting state by Wolfram and Smith [11, 12]. Its table is given in Table 1.

To proceed we first expand the states {A, B} in to {AA, BA, AB, BB} where the second symbol is an "empty" state and expand the whole table by repeating its columns to the right. We also introduce an 'empty" forth state in order to have a completely symmetric 4 x 4 matrix. We emphasize that the last row is an unreachable set of fixed points. The result is shown in Table 2.

This is now equivalent to a discrete 2-variable function over the intervals $[0, 2^2\text{-}1] \times [0, 2^2\text{-}1]$. Applying the concatenation operator it can be turned to a 1-dimensional graph in the combined $[0, 2^4\text{-}1]$ interval. This is the same as reading the table elements column-wise. Transcription is completed with the additional "Interpreter" function which assigns to the binary expansion of integer values in the above graph the special meaning of "colors" for the first two bits and "states" for the last two bits.

We next show the existence of a method by which the above can be turned to a recursive map for deducing the equations of a universal dynamical system that imitates any possible (*m, n*)-Turing Machine, where *n* are the tape symbols (n-ary system) and *m* is the number of internal control states. The first step is to replace the whole infinite tape by a single integer number which will be updated recursively starting from any finite initial condition. This is always possible due to the existence of a polynomial representation of any n-ary alphabet for any finite portion of the TM tape.

At the next step, we will need to express the head's position in a special manner. We introduce a convention by which the initial position of the TM's head is always at 0 (1^{st} symbol of the polynomial representation). The problem then is how to avoid a "negative underflow" in case the head will be finally driven at a

position below zero. This though, can be trivially incorporated into a pair of integers of which the second holds negative values but only its absolute part is taken into the actual computation. We may then introduce a vector $z_n = (x_n, y_n)^T$ where $x_n \geq 0, y_n < 0$ and write the basic evolution equations as

$$\begin{aligned} h_{n+1} &= h_n + 2m_{n+1} - 1 \\ z_{n+1} &= z_n + B(c_{n+1}, h_{n+1})\sigma_{n+1} \\ \sigma_{n+1} &= \left(\frac{sign(h_{n+1}) + 1}{2}, \frac{sign(h_{n+1}) - 1}{2} \right)^T \\ B(c_n, h_{n+1}) &= (c_{n+1} - c_n) b^{h_{n+1}} \end{aligned} \tag{5a}$$

Where h_n is the head's position at everyone time, σ_n is a kind of "spin" vector that alternates between the negative and positive representation of the tape and B_n is a symbol setting function assigning a new color to every new head's position and removing the previous one ('*write*' operation).

The '*read*' operation will now be added with the aid of the '*Bit Extraction Operator*' defined in (2) in order for (5a) to become a complete recursion. To this aim, we introduce the generic map $U_{TM} : \Sigma_n^m \to \Sigma_n^m$ with Σ_n^m being the expanded and symmetrized domain for an (*n, m*)-TM which is the result of the simple transcription of any table of a TM into an integer map described in the previous paragraphs. We can now complete the cycle represented by (5a) with the additional equations

$$\begin{aligned} m_{n+1} &= \hat{X}_h^2(w_{n+1}), \quad c_{n+1} = \hat{X}_0^n(w_{n+1}) \\ w_{n+1} &= U_{TM}(w_n), w_n = c_n \,||\, m_n \,||\, q_n \end{aligned} \tag{5b}$$

where $\hat{X}_a^b$ stands for the function (2) acting on the intermediate result w_n of a complete recursion over z_n. The separate indices *a*

and b that parametrize (2) correspond to the symbol position *a* for a b-ary alphabet respectively. The internal variable q_n stands for any control bits that are simply carried over to the next step by U_{TM} without any other participation in the actual computation.

By virtue of Th. 1 above, the set of equations (5a) and (5b) constitutes a complete recursive map which can be recast in the generic composite form

$$\begin{aligned} z_{n+1} &= z_n + F\left[h_{n+1}, U_{TM}\left(h_n, z_n\right)\right] \\ h_{n+1} &= h_n + 2\hat{X}_{h_n}^2\left(U_{TM}\left(h_n, z_n\right)\right) - 1 \end{aligned} \qquad (6)$$

The above is an algebraic form of an interpreter and from this point of view it corresponds to what is known as an "*Arithmetization*" of a TM.

We note in passing that it is possible to turn this into a completely reversible scheme by introducing a new "memory" state that doubles the interval into [0, 2^5-1] and separating the two symmetric parts of the graph by raising the second with a threshold of the order of 2^4 similarly to a technique introduced previously by Bennett [13, 14]. Nevertheless, what is important for the present work is that if such an efficient transcription scheme exists –and this may include any possible digital circuit like an adder or even a whole CPU!- then there exists a special technique by which we can turn it into a cyclic process. We then provide the following theorem.

> ***Theorem 2.*** *Given a graph of a bounded partial function f over some subset of N there is a morphism that sends f to one of the Cyclic Groups* G_i *of* L_2^{KN} *where* $K = [\log_2(\max\{v_t, f(v_t)\})] + 1$ *and N the number of graph points.*

The proof is again an elementary result of recursive concatenation. By taking the successive concatenation of all input output pairs $\{\nu_i \mid f(\nu_i)\}_{i=0}^{N}$ we construct a second mapping of the form $\nu_i \mid f(\nu_i) \rightarrow \nu_{i+1} \mid f(\nu_{i+1})$ where indices are taken *mod N*. By repeating this process until all distinct N values have been included in each binary string, we end up with a unique mapping which has been restricted to one of the cyclic groups of a lexicon matrix of order analogous to the maximum bit of the set of all input output values by the number of distinct such values.

That most of the symbol-wise functions inherit the natural self-similarity of the representations of the integers becomes evident from the recursive structure of the Lexicon Hierarchy shown by (3). In the next section we examine an important class of such functions.

4. Iterated Sequence Systems

An elementary example of a fundamental class of self-similar functions can be given with the aid of a construct that we shall call an *Iterated Sequence System* (ISS). The simplest such function is the count of one bits in a binary Lexicon (the so called "checksum").

> ***Theorem 2.*** *Let* L_2^N *be the hierarchy of binary lexicons and let* $f_B = \sum_{i=1}^{N} \sigma_i$ *where* σ_i *are the symbol values in each row of the lexicon matrix. Then, there is an iterative procedure that reproduces the graph of* f_B *from a primordial "seed" set* $S_0 = \{0\}$ *given by the recursion* $S_{i+1} = \{S_i, (S_i + 1)\}$. *We shall call this sequence, the "Primordial Sequence".*

The proof is again elementary. For every exponential interval there is a new bit added to each row of a binary lexicon matrix. The set concatenation operation doubles the cardinality of S_i per

step. Hence, for each integer of which the equivalent polynomial representation is contained in the associated matrix row, the number of one bits increases by 1 in every new set.

The above example, although trivial, shows the way symbol-wise functions may inherit the primordial self-similarity of the set of all integer representations. A complete study of such functions is beyond the scope of the present paper and will be presented elsewhere. Nevertheless, there are a number of both elementary and fundamental results that stand as a starting point that we will summarize in the next few lemmas.

We first, generalize the ISS as follows

> ***Definition 1.*** *A generic ISS will be defined by a seed set S_0 of ordered values in some initial DoD, a set of parameters $\Pi = \{\pi_t\}_{t=1}^{K}$, an update function $g : \pi_i' = g(\pi_i)$, an arbitrary operator $\hat{O}_{\{\pi_t\}}$ of which the action is equivalent to some element-wise function $f(s_i^0; \pi_1, \pi_2, ..., \pi_K)$ and a recursion relation*
>
> $$S_i = \left\{ O_\Pi(S_{i-1}), O_\Pi{}'(S_{i-1}) \right\}, \quad O'_\Pi \to O_{g(\pi)} \tag{7}$$

The above are examples of dynamical systems with an exponentially increasing memory of previous instances of their orbit recorded in the whole sequence thus exhibiting higher order correlations. This is expressed in the following definition and the associated theorem.

> ***Definition 2.*** *A canonical ISS will be defined by the choice*
>
> $$S_i = \left\{ S_{i-1}, O_\Pi{}'(S_{i-1}) \right\}, \quad O'_\Pi \to O_{g(\pi)} \tag{8}$$

The simplest example is given by the Primordial Sequence.

Theorem 3. *For every Canonical ISS there is an Expanded Canonical Form represented by the functional operator sequence* $f^0, f^1, f^1, f^2, \ldots$ *acting on the seed set* S_0 *where* f^0 *means the identity operator, and of which the image indices follow the Primordial Sequence.*

The proof starts from the observation that every canonical ISS creates a doubling sequence of the form $\{\{\{S_0\}, S_1\}, \{S_1, S_2\}\}\ldots$ where the number of elements increases as $2^{\nu} |S_0|$ and the images of the first set are defined by the element-wise application of *f*. Consequently, the sequence of indices of the set images increases by 1 per exponential interval thus following the Primordial Sequence. Therefore, from the definition $S_i = (f)^i (S_0)$ we conclude that the sequence of image indices also follows the Primordial Sequence.

An important application of non-canonical ISS is the case of SAT (satisfiability) problems for the so called *CNF/DNF* forms of 1st Order Logic given as sequential disjunctions or conjunctions over an ordered set of logical variables. Given a list of logical variables $\{X_i\}_{i=1}^K$ any CNF/DNF form can be constructed from a fundamental set of clauses of the form $\wedge_{i=1}^K X_i$ or $\vee_{i=1}^K X_i$. Inclusion of negative literals ($\sim X_i$) will be discussed below.

We recall here that for K variables the total truth table can be constructed by an ordered set of all possible binary words which is identical with the L_2^K-lexicon. The set of all possible Boolean functions is also identical with the ordered set of all 2^K words and therefore identical with the $L_2^{2^K}$-lexicon. The particular cases of conjunctive and disjunctive operations correspond to certain rows of the later. In such a case, we may try to construct

a "*Global Clause Function"* (GCF) in the case the list of variables is complete by the recursion

$$S_{i+1} = \left\{\hat{O}_0(S_{i-1}), \hat{O}_1(S_{i-1}),\right\} \;\; \hat{O}_y(S) = \left\{s_j \wedge y\right\} \;\; or \;\; \left\{s_j \vee y\right\} \qquad (9)$$

where we take the seed set as $S_0 = \{0,1\}$.

The significance of the above is evident as every new pair of values for the final truth table follows a doubling sequence with either 0 or 1 used for previous copies in the same manner a lexicon matrix is formed. Such a function will be fairly simple in case no variable is missing from the list resulting either in a $b_1 = [0,1,1...,1]$ or a $b_2 = [0,0,...,0,1]$ bit vector. Inclusion of negative literals can be done with the aid of additional variables by taking their cross-section at the end. Observe that these two obey a particular symmetry law expressed as $b_2 = \hat{\mu}(\sim b_1) = \sim(\hat{\mu}(b_1))$ where $\hat{\mu}$ is the reflection operator that reverses the bit order as mentioned in the classification of section 2.

Assume now that instead of the complete list of variables we have partial clauses with consecutive variables. In the simplest case we could have $X_i \circ X_{i+1}, \; \circ \to \{\wedge, \vee\}$. Then, expansion of the 0 and 1 blocks gives a similar expansion of the first 0 block in the output vector which is always a word of the form $(0)^{\mu}(1)^{\nu}$ with $\mu = 2^{i-1}, \nu = 2^{K-i+1}$. In general any such partial clause will be of the same form *iff* the variables are consecutive.

The only other case that remains is that of a random selection of variables from the complete list forming a partial clause. In such a case though, the only additional effect is the appearance of additional 0 blocks. It is trivial to see that this is in fact a de-synchronization effect due to the isomorphy between the rows of any lexicon matrix with a sampling of cyclic oscillators as indicated in section 1 and further by Lemma 1.1. Hence, it is

always possible to construct a *"Global Clause Function"* in the form of a 2^K x 2^K matrix which simply takes into account all possible combinations of partial clauses.

An example with 5 variables is shown in Fig. 4 for the "OR" and the "AND" case respectively where the recursive structure and the complementary nature of the two graphs is evident. These observations may lead to an effective algorithm for all SAT problems as well as an analog realization based on oscillator states that will be reported elsewhere.

In the next section we concentrate in alternative representations of the rows of binary lexicons in terms of their associated polynomials and their possible relations with previously proposed complexity measures.

5. Polynomial representations and sequence complexity

There are at least two previous cases in the literature [15, 16] where the roots of polynomial representations of the integers has been studied. The first case has been studied by Odlyzko et al [15] for all polynomials of the form

$$\sum_{\iota=1}^{\nu} \sigma_{\iota} z^{\nu}, \quad \sigma_{i} \in [0,1] \qquad (10)$$

and it is shown in fig. 3. A second case of closely associated "Rotated" Binary Patterns which give rise in the so called Littlewood polynomials [16] shown in fig. 4 can be defined as

$$\sum_{\iota=1}^{\nu} \sigma_{i}' z^{\nu}, \quad \sigma_{i}' = 1 - 2\sigma_{i} \in [-1,1] \qquad (11)$$

A third representation is introduced here for the first time.

Representation Lemma 2.1: *given an arbitrary finite binary sequence* $\{s \mid s_i \in [0,1], i = 1...N\}$ *and an encoding scheme*

$s \to s' : s'_i = 2s_i - 1 : s'_i \in [-1,1]$ *we define the Cluster Representation C(s) of the same sequence*

$$C(s) = \{C_j \in N\}_{j=1}^{k} : C_j = \sum_{i:s_i = s_{i+1}} s_i \qquad (12)$$

where k is the number of subsequent clusters of same symbols which shall be called the "Cluster Dimension". The cluster representation contains the same information as the original sequence in the sense that it is absolutely invertible and the original can always be reconstructed by the contracted form of C(s).

Based on the above, we also provide a possible complexity index for finite sequences.

Definition*: given a cluster representation of an arbitrary finite binary sequence we define the Cluster Index as*

$$I_c = \frac{1}{D_c} \sum_{j=1}^{D_c} |C_j| \log_2 |C_j|, \quad \sum_{j=1}^{D_c} |C_j| = N \qquad (13)$$

Dc being the "Cluster Dimension" and C_j the population of each cluster.

Next, we construct a second type of index through a second representation, namely the associated polynomial root representation.

Representation Lemma 2.2: *given an arbitrary finite binary sequence* $\{s \mid s_i \in [0,1], i = 1...N\}$ *and its associated Cluster Representation of the same sequence C(s) we define the Polynomial Root Representation* ρ(*s*) through *the set of all complex roots of the associated polynomial*

$$\rho(s) = \left\{\rho_j\right\}_{j=1}^{Dc-1} : p_s(z) = \sum_{j=1}^{D_c} C_j z^j$$
$$= C_{D_c-1} \prod_{j=1}^{D_c-1} (z - \rho_j)^m, z \in C \qquad (14)$$

The root representation contains the same information as the original sequence in the sense that it is absolutely invertible and the original can always be reconstructed from the contracted form of ρ(s).

The set of roots of (14) for L_2^{12} is shown in Fig. 5.

With the above, we may attempt a second alternative definition of a complexity index.

***Definition**: given a polynomial root representation ρ(s) of an arbitrary finite binary sequence s we define the generic Root Index as*

$$I_\rho^\lambda = \sum_{j=1}^{D_c-1} \rho_j^\lambda, \quad \lambda \in R \qquad (15)$$

This is the only case where a non-trivial structure is revealed for the sum of roots index. For l=1 the result is shown in Fig. 6 and it is always real as demanded by the Fundamental Theorem of Algebra.

We may compare the measure provided by (5) with the Block-complexity defined in Allouche [17] which is based on constant length "sliding" over an arbitrarily long sequence. A basic result is that for ultimately periodic sequences, the number of different blocks of length *k* scales as O(*k*).

Karamanos in [19] and [20] also argued about the possibility of using the so called "lumping" technique where only successive blocks of constant length over a sequence are used. In this case,

he shows that the same number becomes constant for sequences that are ultimately periodic.

From what has been said in section 2, it is evident that all such sequences are members of a subset of the binary lexicon hierarchy. If a sequence becomes ultimately periodic with a period τ above some critical length N_0 then there will be a subsequence of lexicon matrices $\left\{L_2^{N_0+\nu\tau}\right\}$ such that each cluster of τ symbols will be either on the left half or the right half of each lexicon matrix as the two parts are always complementary with respect to the exchange operation (0 → 1, 1 → 0). Thus the cluster dimension of ultimately periodic sequences will be a symmetric function on any exponential interval. Moreover, as the number of clusters increases linearly with the iteration index v, so must do the Cluster Dimension. Therefore we conclude the following

Theorem 2*: For any ultimately periodic binary sequence the associated Cluster Index scales as* $C_0 + \nu\kappa,\ \kappa = \tau \log_2 \tau$. *Accordingly, the Cluster Dimension scales as O*(v).

Alternatively, one can check the set of all cluster indices in an increasing sequence of exponential intervals for their associated lexicon matrices. An example is given in Fig. 7 for the L_2^{10} case.

As it is immediately evident, such a function is again self-similar, thus its graph will be most probably associated with an appropriate ISS. If this is the case, then the graph of this and similar functions may be algebraically known beforehand which leads to the important conjecture that for every sequence of which the length is bounded from above we may know its degree of pseudo-randomness based on a hierarchy of properties of increasing complexity. That is to say that a possible complete classification of all finite sequences of arbitrary length might be

possible due to the existence of self-similar index functions like (15).

6. Conclusions

The work presented here contains collective evidence from many areas of modern mathematics that appear to contain an algebraic structure and a natural self-similarity inherited from the primordial self-similarity of the natural numbers.

Moreover, the targeting was towards a more fundamental foundation of the above conclusion in the lines of the old Kronecker saying according to which "God made the *integers*, all else is the work of man". Thus, it represents a rather "Neo-Pythagorian" approach to the whole subject of discrete mathematics.

Nevertheless, from a practical viewpoint, it appears that the toolbox of the Universal Lexicons constructs can serve a number of practical purposes quite well, due to its ability to incorporate the power of induction through visualization. Hopefully, some of the applications discussed above will be further explored in future work especially with respect to the possibility of new, alternative computational architectures.

References

[1] D. Bailey, "New Math Formulas Discovered With Supercomputers", NAS News 2 (24), 1997.

[2] D. Bailey, J. M. Borwein, "Experimental Mathematics and Computational Statistics", Wires Comp. Stat. 1, 12 – 24 (2009).

[3] J. M. Borwein, D. Bailey, "*Mathematics by Experiment: Plausible Reasoning in the 21st Century",* A.K. Peters, Nattick, MA (2004).

[4] A. Church, "A Formulation of the Simple Theory of Types", *Journal of Symbolic Logic*, Volume 5 (1940).

[5] J. Copeland ed. "*The Essential Turing: Seminal Writings in Computing, Logic, Philosophy, Artificial Intelligence, and Artificial Life plus The Secrets of Enigma",* Clarendon Press (Oxford University Press), Oxford UK (2004).

[6] Barr, M. (1996). "The Chu construction". *Theory and Applications of Categories* **2** (2): 17–35.

[7] Pratt, V. R.. "The Stone gamut: A coordinatization of mathematics". *Proc. 10th Annual IEEE Symp. on Logic in Computer Science, Montreal, June 1995*. pp. 444–454.

[8] Pratt, V. R., "*Chus Spaces: Automata with Quantum Aspects*", Workshop on Physics and Computation, 1994. PhysComp '94, Proceedings, p. 186 – 195.

[9] A. Gregori, "Relativity as classical limit in a combinatorial scenario", arxiv.org/abs/0911.0518

[10] A. Gregori "Beyond Quantum Mechanics and General Relativity", arxiv.org/abs/1002.4491

[11] S. Wolfram, "A New Kind of Science" (2002), p 709.

[12] http://en.wikipedia.org/wiki/Wolfram's_2-state_3-symbol_Turing_machine

[13] C. H. Bennett, "Logical reversibility of computation," IBM Journal of Research and Development, vol. 17, no. 6, pp. 525-532, (1973).

[14] C. H. Bennett, "The Thermodynamics of Computation -- A Review," International Journal of Theoretical Physics, vol. 21, no. 12, pp. 905-940, (1982).

[15] A. M. Odlyzko and B. Poonen, *L'Enseign. Math.*, 39 (1993), 317-348.

[16] P. Borwein (2002) "*Computational Excursions in Analysis and Number Theory*." CMS Books in Mathematics. Springer-Verlag. pp. 2–5,121–132.

[17] J. P. Allouche, "Algebraic and Analytic Randomness", Lect. Notes in Phys. (2000), Vol. 550, 345

[18] K. Karamanos, Chaos, Solitons & Fractals, 10 (7) 1135 – 1150 (1999).

[19] K. Karamanos, J. Phys. A.: Math. Gen. 34, 9231 – 9241 (2001).

[20] J. Konvalina, "Combinatorial Fractal Geometry with a Biological Application," *Fractals* , 14 (2006) 133-142

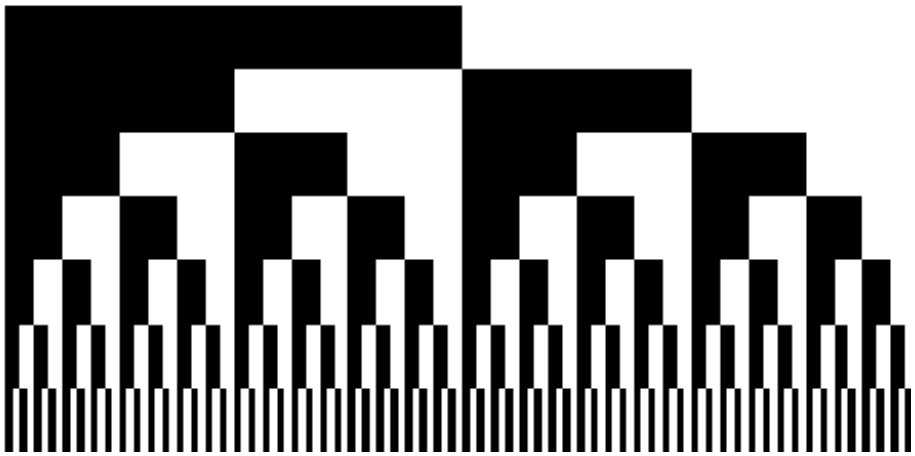

Fig. 1 (black → 0, white → 1)

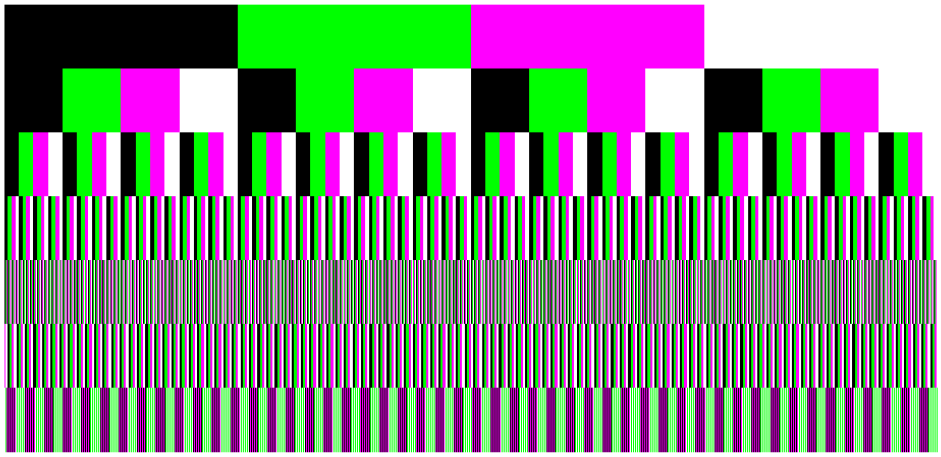

Fig. 2 (black → 0, green → 1, pink → 2, white → 3)

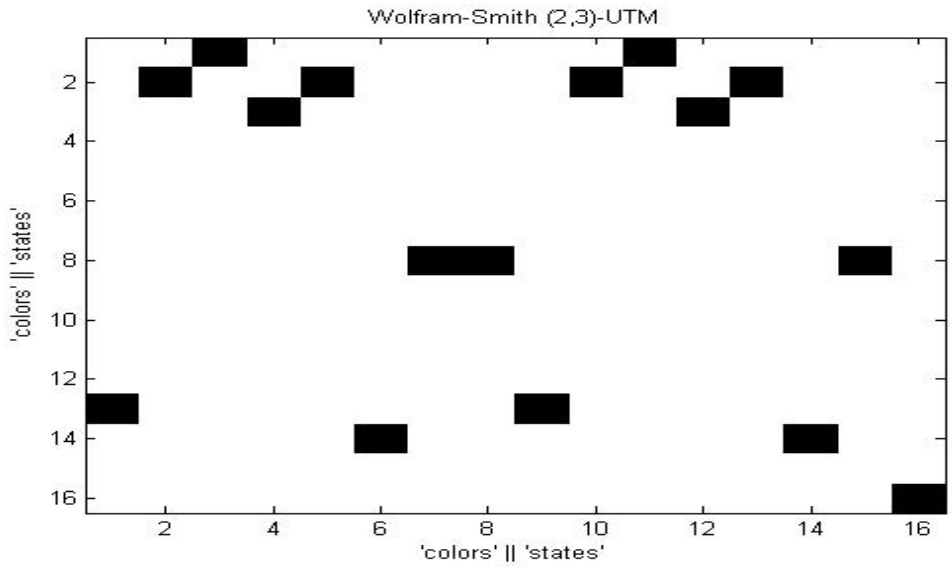

Fig. 3

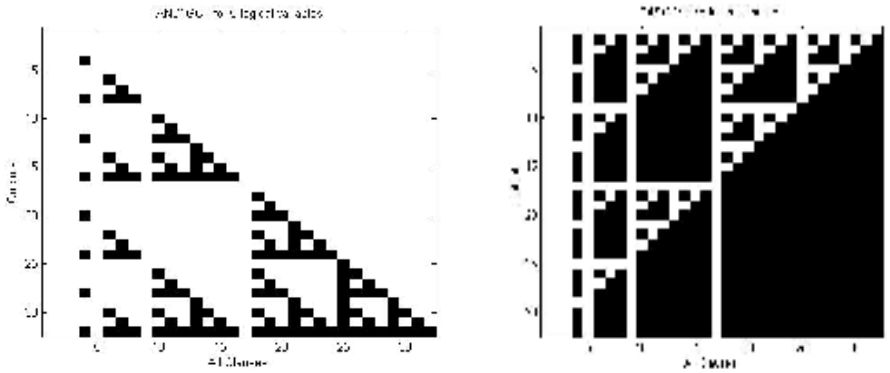

Fig. 4 (black → 0, white → 1)

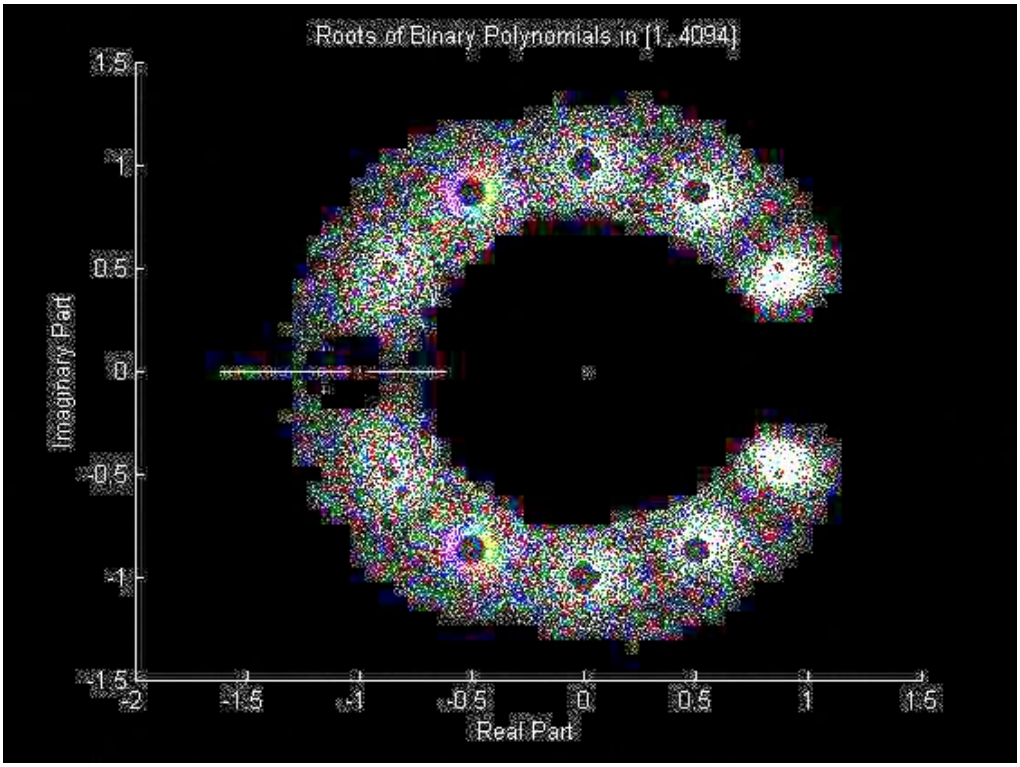
Roots of Binary Polynomials in [1, 4094]
Imaginary Part
Real Part
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-1.5
-1
-0.5
0
0.5
1
1.5

Fig. 5

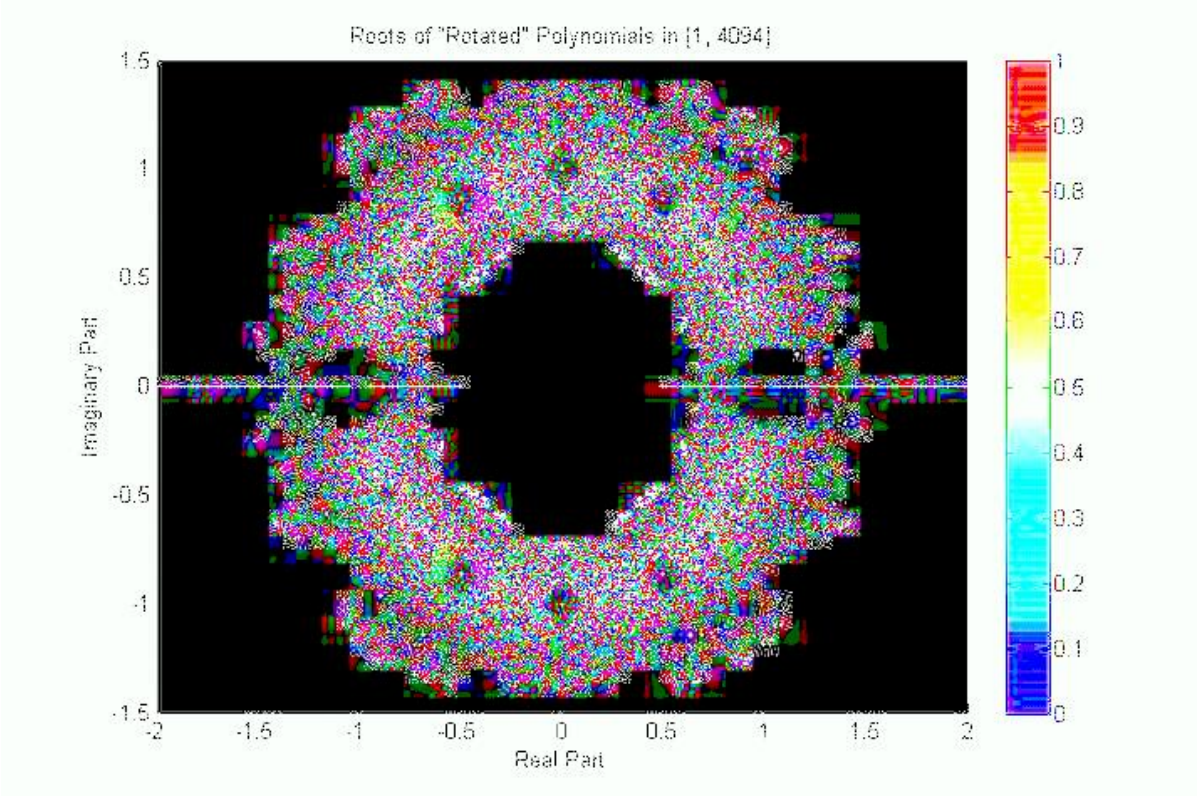
Roots of "Rotated" Polynomials in [1, 4094]
Imaginary Part
Real Part
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-1.5
-1
-0.5
0
0.5
1
1.5
2
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1

Fig. 6

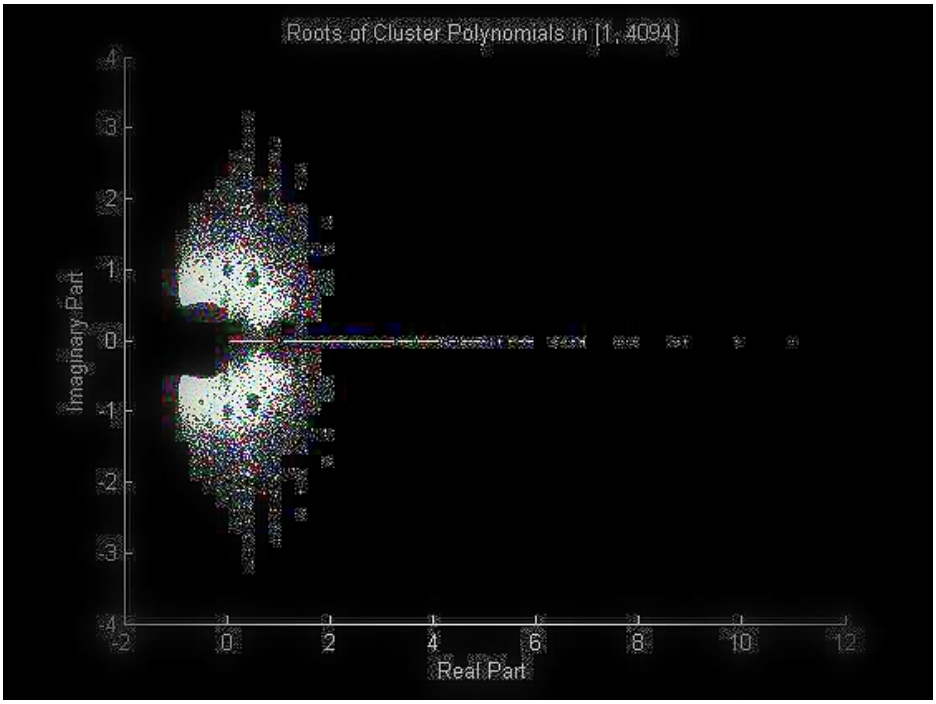
Roots of Cluster Polynomials in [1, 4094]
Imaginary Part
Real Part
4
3
2
1
0
-1
-2
-3
-4
-2
0
2
4
6
8
10
12

Fig. 7

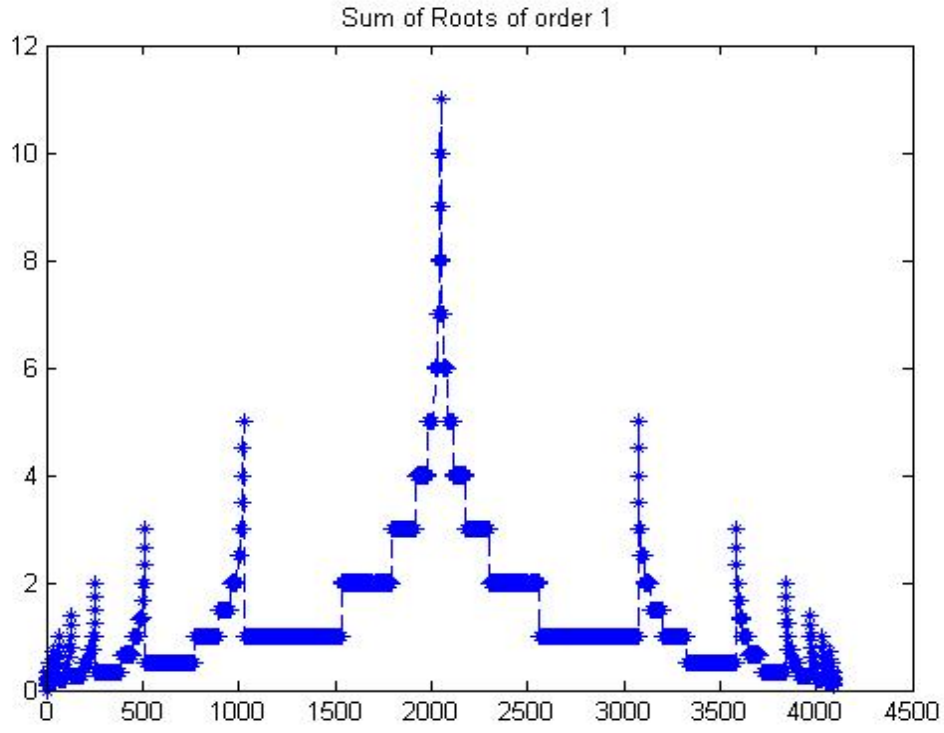
Sum of Roots of order 1
12
10
8
6
4
2
0
0
500
1000
1500
2000
2500
3000
3500
4000
4500

Fig. 8

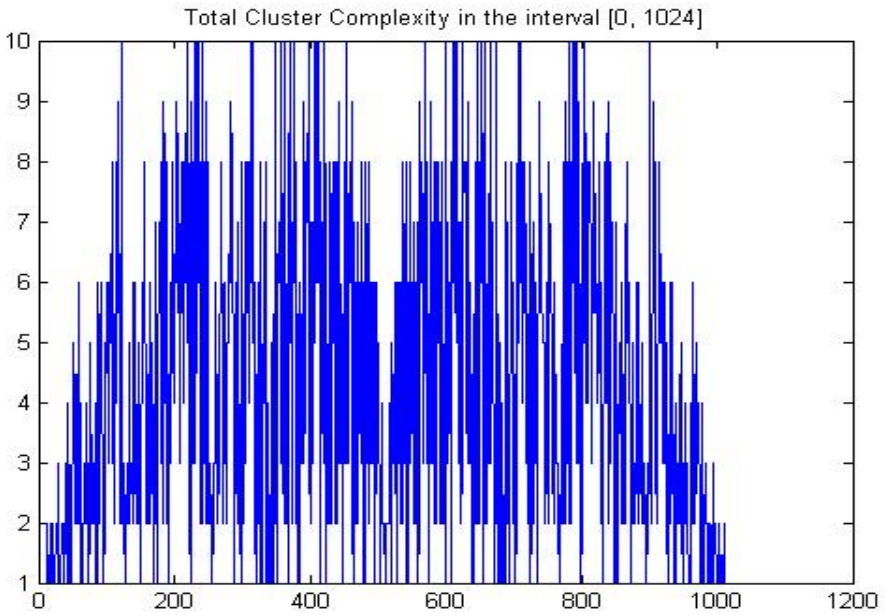

Fig. 9

	A	B
0	P1,R,**B**	P2,L,**A**
1	P2,L,**A**	P2,R,**B**
2	P1,L,**A**	P0,R,**A**

Table 1

	AA(0)	**BA(1)**	**AB(2)**	**BB(3)**
0	P1,R,**B**	P2,L,**A**	P1,R,**B**	P2,L,**A**
1	P2,L,**A**	P2,R,**B**	P2,L,**A**	P2,R,**B**
2	P1,L,**A**	P0,R,**A**	P1,L,**A**	P0,R,**A**
3	P3,L,A	P3,L,B	P3,R,A	P3,R,B

Table 2