ELSEVIER

Physica A 288 (2000) 10–20

PHYSICA A

www.elsevier.com/locate/physa

Factorizable language: from dynamics to bacterial complete genomes

Bailin Hao[a,*,1], Huimin Xie[b,2], Zuguo Yu[b], Guo-yi Chen[b]

[a] *Department of Physics and Centre for Nonlinear Studies, Hong Kong Baptist University, Hong Kong*
[b] *Institute of Theoretical Physics, Academia Sinica, P.O. Box 2735, Beijing 100080, People's Republic of China*

Abstract

Symbolic sequences generated by symbolic dynamics of a dynamical system belong to a special class of language in which any admissible word is factorisable as well as prolongable. From a complete genome sequence of an organism, one may also define a factorizable language. A factorizable language enjoys the nice property that it is entirely determined by the set of minimal fobidden words or distinct excluded blocks (DEBs). We use this property to calculate the fractal dimension of patterns related to a visualisation scheme of under-represented strings in bacterial complete genomes within the limit of infinitely long strings. The same problem may be solved by using a purely combinatorial approach. The methods described in this paper may be applied to other regular fractals with self-similar and self-overlapping structure.

Keywords: Language; Symbolic dynamics; Genome

1. Introduction

We start from the following observation.

For those studying high-energy particle physics the six letters

u d c s b t

denote different types of quarks. They are associated with certain fractional charge, mass, flavour, charm, and other quantum numbers. However, many more people deal

* Correspondence address: Institute of Theoretical Physics, Academia Sinica, P.O. Box 2735, Beijing 100080, People's Republic of China.

[1] On leave from the Institute of Theoretical Physics, Academia Sinica, Beijing.

[2] On leave from the Department of Mathematics, Suzhou University, Jiangsu 215006, People's Republic of China.

0378-4371/00/$ - see front matter
PII: S0378-4371(00)00411-8

with the three letters

p n e

as names of proton, neutron, and electron with definite mass, charge, spin, magnetic momentum, etc. There is no need to know from which three quarks a proton or a neutron is made. Chemists know well the symbols

H C N O P S ···

representing different atoms. They are concerned with the atomic number, ion radius, chemical valence and affinity, of the corresponding element. Using these atomic symbols chemists write molecular formulas such as H_2O, NO, or CO_2. However, when it comes to biochemistry it is usually not necessary and convenient to write down explicitly all the 30 odd atoms forming the phosphate, sugar and base parts of a nucleotide. Suffice it to denote the nucleotides adenine, cytosine, guanine, and thymine by the four letters **a, c, g, t** and to remember that when forming a double helix of DNA a **a** is associated with a **t** by two hydrogen bonds ("weak coupling") while a **c** conjugates with a **g** with three hydrogen bonds ("strong coupling"). A long sequence of DNA made of millions of **a, c, g** and **t** may be denoted by a single symbol in a "higher"-level analysis.

What is the moral of the observation? In describing Nature, we can only concentrate on one or another level by ignoring the detailed structure and dynamics in smaller scales. This is nothing but *coarse-graining*. Coarse-graining is inevitably associated with the use of symbols and in many lucky cases these symbols make one-dimensional sequences. (By the way, "high"-dimensional strings may be treated as one-dimensional with farther than nearest-neighbor interactions if one confines to strings of finite length.) These symbolic sequences fit well into the scheme of formal languages where a wealth of knowledge has been accumulated. In fact, formal languages are not just formal. They may provide a definite framework for comparison or a computation scheme to solve practical problems.

We will touch on two.

2. Some notions from language theory

In formal language theory, one starts with an alphabet, e.g., $\Sigma=\{R,L\}$ in the symbolic dynamics of unimodal maps or $\Sigma=\{a,c,g,t\}$ for DNA sequences of an organism.[3] Let Σ^* denote the collection of all possible strings made of letters from Σ, including the empty string ε. Any subset $L\subset\Sigma^*$ is called a *language* over the alphabet Σ. This general definition of language cannot lead us very far. One must specify how the subset is formed. To this end, one may devise a generative grammar, i.e., a set of production rules to generate words in the language by applying the rules to some starting symbols.

[3] All genome sequences mentioned in this paper are fetched by anonymous FTP from GenBank maintained by the National Center for Biotechnology Information at `http://ncbi.nlm.nih.gov`

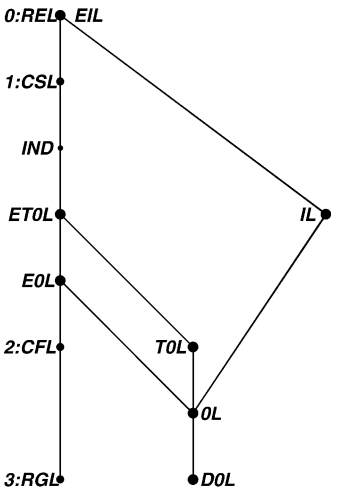

Fig. 1. Relationship of various classes of languages.

Two generative schemes are well-developed: the sequential production of Chomsky and the parallel production of Lindenmayer. We show their relationship in Fig. 1.

The set $L' = \Sigma^* - L$ defines the complementary language. A language L is a factorizable language if any substring of a word $x \in L$ also belongs to L. A factorizable language has a minimal set of forbidden words or *Distinctive Excluded Blocks* [1] (DEBs) L'' such that if $x \in L''$, then any proper substring of x belongs to L. A factorizable language is completely determined by its set of DEBs:

$$L = \Sigma^* - \Sigma^* L'' \Sigma^* .$$

A prominent example of factorizable language is given by the admissible symbolic sequences in the symbolic dynamics of a dynamical system, see, e.g., Refs. [2,3]. Another class of factorizable languages may be obtained from a complete genome as follows. Let G be a complete genome of an organism; it may consist of one or more linear or circular sequences. All possible substrings of G, including the empty string ε and G itself, obviously form a subset of Σ^* and, thus, define a language which is factorizable by construction.

Any language $L \subset \Sigma^*$ introduces an equivalence relation R_L in Σ^* with respect to L. For any pair $x, y \in \Sigma^*$ $xR_L y$ iff for each $z \in \Sigma^*$ either both $xz, yz \in L$ or both $xz, yz \notin L$. The number of equivalence classes in Σ^* with respect to L defines the *index* of R_L, denoted by index (R_L).

An important theorem (Myhill–Nerode) says that L is a regular language iff index (R_L) is finite and L being regular implies that the minimal deterministic automaton corresponding to L, $\min \mathrm{DFA}(L)$, is unique up to an isomorphism, i.e., to renaming of the states. Moreover, the number of states in the $\min \mathrm{DFA}(L)$ equals to index (R_L).

Let L be a factorizable language and L'' be its set of all DEBs. Define a set

$$V = \{v \mid v \text{ is a proper prefix of some } y \in L''\} .$$

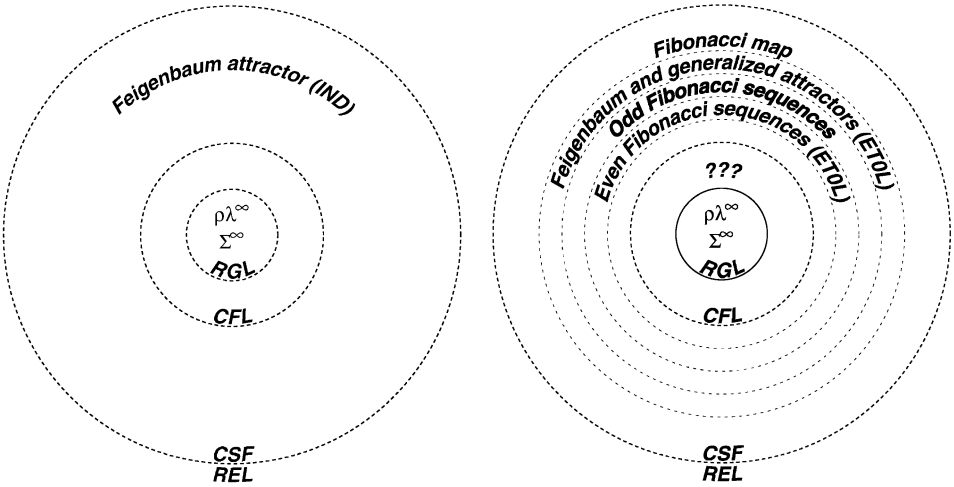

Fig. 2. Grammatical complexity of symbolic sequences in unimodal maps: left: what was known in 1991; right: what is known now.

For each word $x \in L$ there exists a string $v \in V$ such that xR_Lv. In other words, all equivalence classes of L are represented in the set V. In order to find all equivalence classes of Σ^* with respect to L it is enough to start from L''. In addition, L' is an equivalence class of Σ^*. For two given strings $u, v \in V$, uR_Lv iff for each $z \in \Sigma^*$ uz contains a DEB as its suffix $\Leftrightarrow vz \in L'$ and vice versa. This statement sets the computation rule to identify all equivalence classes. Each equivalence class may be named after a member $x_i \in L$ and be denoted as $[x_i]$. The transfer function between states of $\min \mathrm{DFA}(L)$ is defined as $\delta([x_i], s) = [x_i s]$ for $x_i \in L$ and $s \in \Sigma$.

3. Complexity of symbolic sequences in unimodal maps

Grammatical complexity of symbolic sequences in unimodal maps is shown in Fig. 2.

4. Visualization of under-represented strings

We start by considering on how to visualize the avoided and under-represented strings in a bacterial genome whose length is usually the order of a few million letters.

There are 4^K different strings of length K made of four letters. In order to check whether all these strings appear in a genome, we use 4^K counters to be visualized as a $2^K \times 2^K$ square array on a computer screen. These can be realized as a direct product of K identical 2×2 matrices:

$$M^{(K)} = M \otimes M \otimes \cdots \otimes M \,,$$

where

$$M = \begin{pmatrix} g & c \\ a & t \end{pmatrix}.$$

We call this $2^K \times 2^K$ square a K-frame. In practice, it is convenient to use binary subscripts for this 2×2 matrix and it is easy to develop an algorithm that depends only on the total length of the genome but not on the string length K. Put in a frame of fixed K and described by a color code biased towards small counts, each bacterial genome shows a distinctive pattern which indicates on absent or under-represented strings of certain types [4]. For example, many bacteria avoid strings containing the string *ctag*. Any string that contains *ctag* as a substring will be called a *ctag*-tagged string. If we mark all *ctag*-tagged strings in frames of different K, we get pictures as shown in Fig. 3. The large-scale structure of these pictures persists but more details appear with growing K. Excluding the area occupied by these tagged strings, one gets a fractal in the $K \to \infty$ limit. It is natural to ask as what is the dimension of this fractal for a given tag or for a given set of tags.

In fact, this is the fractal dimension of the complementary set of the tagged strings. The simplest case is that of *g*-tagged strings. As the pattern has an apparently self-similar structure the fractal dimension is easily calculated to be

$$D = \frac{\log 3}{\log 2}.$$

Moreover, the fractal dimension of all other cases must lie in between $\log 3/\log 2$ and 2. However, the calculation of these dimensions is somewhat tricky as one must take into account the overlap of patterns precisely. For example the $K = \infty$ frame with all *cg*-tagged strings marked is shown in Fig. 4. There is one big shaded square representing all possible strings with the two leading letters being *cg*. There are four middle-size squares coming from strings starting from *g*, *c*, *a*, or *t* with the next two letters being *gc*. There are 16 small squares, 64 tiny squares, etc., whose meaning may be read off in a similar manner. However, one of the 16 small squares is contained in the big square; eight of the tiny squares are contained in the larger ones. At the next level (not shown), 47 of 256 even tinier squares are located in larger ones. Without taking into account these overlaps precisely, the fractal dimension of the limiting complementary set cannot be calculated.

Now, let a_K be the number of all strings of length K that do not contain the given tag. As the linear size δ_K in the K-frame is $1/2^K$, the fractal dimension may be calculated as

$$D = \lim_{K\to\infty} \frac{\log a_K}{-\log \delta_K} = \lim_{K\to\infty} \frac{\log a_K^{1/K}}{\log 2}.$$

Suppose the generating function of a_K is known:

$$f(s) = \sum_{K=0}^{\infty} a_K s^K,$$

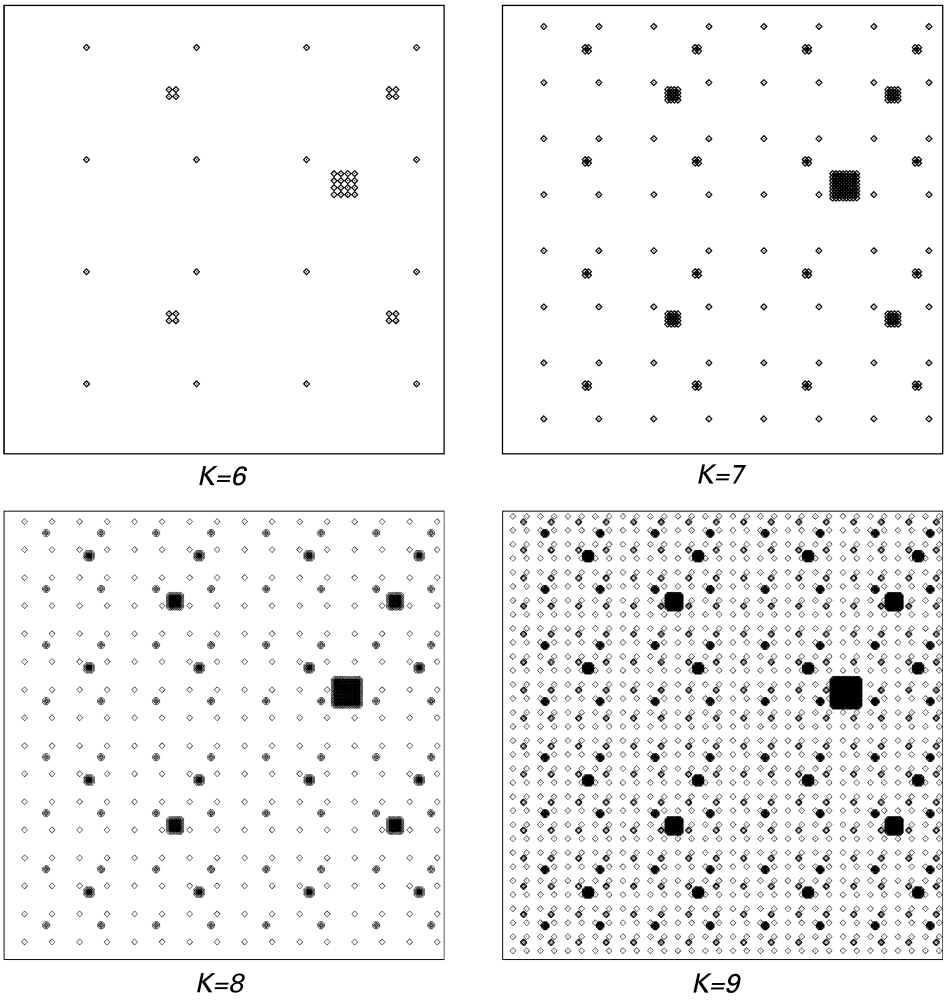

Fig. 3. *ctag*-tagged strings in $K = 6$–9 frames.

where s is a complex auxiliary variable, then according to the Cauchy criterion of convergence we have

$$\lim_{K\to\infty} a_K^{1/K} = \frac{1}{|s_0|} ,$$

where s_0 is the minimal positive zero of $f^{-1}(s)$. This finally determines the fractal dimension

$$D = -\frac{\log|s_0|}{\log 2} .$$

Before undertaking to calculate the generating function $f(s)$ for various tags we indicate another related problem, namely, the problem of *redundant* and *true* avoided strings.

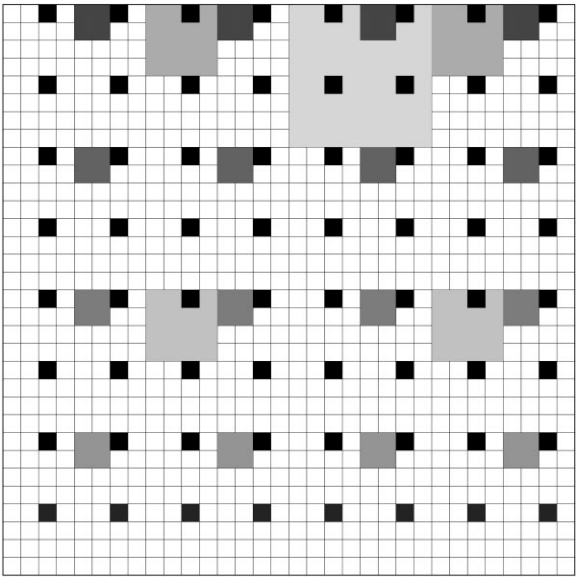

Fig. 4. The pattern of *cg*-tagged strings showing the overlaps.

Once we know that there are avoided strings in the complete genomes from the visualization scheme, one can perform a direct search for these strings. The direct search has the merit of not being significantly limited by the string length K. Take, for example, the complete genome of *E. coli*. At $K=7$ the first avoided string *gcctagg* is discovered. At the next $K=8$ level a total of 173 avoided strings are identified. However, these 173 strings are not all true avoided strings as some must be the consequence of the absence of the $K=7$ string *gcctagg*. A naive estimate of the redundant avoided strings without taking into account any possible overlap of substrings would lead to $4^i(i+1)$: if there is only one avoided string at the Kth level, it would take away 8, 48, 256, 1280, 6144, 28 672, ..., strings at the next $K+i$ levels. This estimate works well for *E. coli* until $K=13$ when the overlap of the first and the last letter g in the true avoided string *gcctagg* would show off. Applying the Goulden–Jackson cluster method to the case of only one "bad word" *gcctagg* leads to the following generating function:

$$f(s)=\frac{1+s^6}{1-4s+s^6-3s^7}\,.$$

The number of redundant avoided strings are given by

$$\frac{1}{1-4s}-f(s)=s^7+8s^8+48s^9+256s^{10}+1280s^{11}$$
$$+6144s^{12}+28671s^{13}+131063s^{14}+\cdots\,.$$

The deviation from the naive estimate appears from the term s^{13}.

For a non-trivial example, we consider the newly published complete genome of the hyperthermophilic bacterium *Aquifex aeolicus* [5]. For this 155 1335-letter sequence

four avoided strings are identified at $K = 7$. They form the set B of "bad words":

$$B = \{gcgcgcg, gcgcgca, cgcgcgc, tgcgcgc\} .$$

As there are significant overlaps among these strings, the naive estimate of redundant avoided words can hardly work. The application of the Goulden–Jackson cluster method requires the solution of a system of four linear equations and leads to the following generating function:

$$f(s) = \frac{1 + s^2 + s^4 + s^6 + s^8 + s^{10} + s^{12}}{1 - 4s + s^2 - 4s^3 + s^4 - 4s^5 + s^6 - 4s^8 - 4s^{10} - 4s^{12}} .$$

The numbers of redundant avoided strings are given by

$$\frac{1}{1-4s} - f(s) = 4s^7 + 27s^8 + 152s^9 + 784s^{10} + 3840s^{11} + 18176s^{12} + 83968s^{13} + \cdots .$$

In what follows, we show that these results may be obtained by an entirely different method, namely, by making use of formal language theory. For convenience of presentation, we first collect a few notions from language theory without proofs. The details may be found, e.g., in Ref. [2] and references therein.

5. Language theory solution

Now we apply what has just been said to the complete genome of *Aquifex aeolicus* with its set B of four avoided strings at length $K=7$. Although there are longer avoided strings we take B to be its L'' for the time being. From the proper suffixes of these strings we get the set

$$V = \{g, gc, gcg, gcgc, gcgcg, gcgcgc, c, cg, cgc, cgcg, cgcgc, cgcgcg, t, tg, tgc, tgcg, tgcgc, tgcgcg\} .$$

By checking the equivalence class of these strings only 13 out of these 18 strings are kept as representatives of each class. Adding the class $[L'] \subset \Sigma^*$ we get the following 14 equivalence classes of Σ^*:

$$[\varepsilon]\ [g]\ \ [gc]\ \ [gcg]\ \ [gcgc]\ \ [gcgcg]\ \ [gcgcgc]$$
$$[c]\ [cg]\ [cgc]\ [cgcg]\ [cgcgc]\ [cgcgcg]\ [L'] .$$

The transfer function $\delta([x_i], s) = [x_i s]$, $x_i \in V$ and $s \in \Sigma$, is determined by attributing $[x_i s]$ to the existing equivalence classes. They are listed in Table 1. The particular transfer function $\delta([x_i], s) = [L']$ leads to a "dead end".

One draws the minimal deterministic automaton according to the above transfer function. As it is no longer a planar graph we do not show it here. By counting the

Table 1
The transfer function for the minimal deterministic automaton for *Aquifex aeolicus*

$[x_i] \setminus s$	a	c	g	t
$[\varepsilon]$	$[\varepsilon]$	$[c]$	$[g]$	$[c]$
$[g]$	$[\varepsilon]$	$[gc]$	$[g]$	$[c]$
$[gc]$	$[\varepsilon]$	$[c]$	$[gcg]$	$[c]$
$[gcg]$	$[\varepsilon]$	$[gcgc]$	$[g]$	$[c]$
$[gcgc]$	$[\varepsilon]$	$[c]$	$[gcgcg]$	$[c]$
$[gcgcg]$	$[\varepsilon]$	$[gcgcgc]$	$[g]$	$[c]$
$[gcgcgc]$	$[L']$	$[c]$	$[L']$	$[c]$
$[c]$	$[\varepsilon]$	$[c]$	$[cg]$	$[c]$
$[cg]$	$[\varepsilon]$	$[cgc]$	$[g]$	$[c]$
$[cgc]$	$[\varepsilon]$	$[c]$	$[cgcg]$	$[c]$
$[cgcg]$	$[\varepsilon]$	$[cgcgc]$	$[g]$	$[c]$
$[cgcgc]$	$[\varepsilon]$	$[c]$	$[cgcgcg]$	$[c]$
$[cgcgcg]$	$[\varepsilon]$	$[L']$	$[g]$	$[c]$

number of lines leading from one state to another, we write down an *incidence matrix*:

$$M = \begin{bmatrix}
1 & 1 & & & & & & 2 & & & & & \\
1 & 1 & 1 & & & & & 1 & & & & & \\
1 & & & 1 & & & & 2 & & & & & \\
1 & 1 & & & 1 & & & 1 & & & & & \\
1 & & & & & 1 & & 2 & & & & & \\
1 & 1 & & & & & 1 & 1 & & & & & \\
 & & & & & & & 2 & & & & & \\
1 & & & & & & & 2 & 1 & & & & \\
1 & 1 & & & & & & 1 & & 1 & & & \\
1 & & & & & & & 2 & & & 1 & & \\
1 & 1 & & & & & & 1 & & & & 1 & \\
1 & & & & & & & 2 & & & & & 1 \\
1 & 1 & & & & & & 1 & & & & &
\end{bmatrix}.$$

The columns and rows of the matrix M are ordered as elements in the first column in Table 1 of the transfer function.

To make connection with the generating function

$$f(s) = \sum_{0}^{\infty} a_K s^K ,$$

obtained by using the Goulden–Jackson cluster method, we note that the characteristic polynomial of M is related to $f(1/\lambda)$:

$$\det(\lambda I - M) = \lambda^{13} f\left(\frac{1}{\lambda}\right) .$$

Table 2
Elements of the first rows of M_K and their sum

$K=$	1	2	3	4	5	6	7	8	9	10	11
	1	4	16	64	256	1024	4095	16 378	65 501	26 19 60	1 047 664
	1	2	8	32	128	512	2048	8190	32 756	131 002	523 920
	0	1	2	8	32	128	512	2048	8190	32 756	131 002
	0	0	1	2	8	32	128	512	2048	8190	32 756
	0	0	0	1	2	8	32	128	512	2048	8190
	0	0	0	0	1	2	8	32	128	512	2048
	0	0	0	0	0	1	2	8	32	128	512
	2	7	28	112	448	1792	7168	28 665	114 640	458 483	1 833 624
	0	2	7	28	112	448	1792	7168	28 665	114 640	458 483
	0	0	2	7	28	112	448	1792	7168	28 665	114 640
	0	0	0	2	7	28	112	448	1792	7168	28 665
	0	0	0	0	2	7	28	112	448	1792	7168
	0	0	0	0	0	2	7	28	112	448	1792
Sum:	4	16	64	256	1024	4096	16 380	65 509	261 992	1 047 792	4 190 464
							−4	−27	−152	−784	−3840

Moreover, the sum of elements in the first row of the Kth power of M is nothing but a_K [1]:

$$a_K = \sum_{j=1}^{13} (M^K)_{1j} .$$

The summation runs over all equivalence classes except for L'. We list the elements of the first row of M^K in columns of Table 2.

The negative numbers in the last row of Table 2 show the difference of a_K and 4^K. They are precisely the coefficients in the expansion of $1/(1-4s) - f(s)$, shown at the end of Section 6. We see that the transfer function and the incidence matrix contain more detailed information on the combinatorial problem than the generating function alone. The consequence of this approach has to be further elucidated in the future.

6. Combinatorial solution

The generating function for the numbers of strings of various length made of the four letters that do not contain certain designated strings ("bad words" as called in Ref. [6]) may be calculated by using the Goulden–Jackson cluster method [7], well-described by Noonan and Zeilberger [6]. In particular, the case of a single tag – one "bad word" only – is easily treated and some of the results are shown in Table 3.

A related question is the number $G(n)$ of different types of generating functions for a given tag length n. These numbers turn out to be independent of the size of the alphabet Σ as long as there are more than two letters in Σ [8]:

n	1	2	3	4	5	6	7	8	9	10	11	12	13	14
$G(n)$	1	2	3	4	6	8	10	13	17	21	27	30	37	47 .

Table 3
Generating function and dimension for some single tags

Tag	$f(s)$	D	Tag	$f(s)$	D
g	$\frac{1}{1-3s}$	$\frac{\log 3}{\log 2}$	ggg	$\frac{1+s+s^2}{1-3s-3s^2-3s^3}$	1.98235
gc	$\frac{1}{1-4s+s^2}$	1.89997	$ctag$	$\frac{1}{1-4s+s^4}$	1.99429
gg	$\frac{1+s}{1-3s-3s^2}$	1.92269	$ggcg$	$\frac{1+s^3}{1-4s+s^3-3s^4}$	1.99438
gct	$\frac{1}{1-4s+s^3}$	1.97652	$gcgc$	$\frac{1+s^2}{1-4s+s^2-4s^3+s^4}$	1.99463
gcg	$\frac{1+s^2}{1-4s+s^2-3s^3}$	1.978	$gggg$	$\frac{1+s+s^2+s^3}{1-3s-3s^2-3s^3-3s^4}$	1.99572

In fact, these $G(n)$ are the so-called correlations of n as given by the integer sequence $M0555$ in Ref. [9], see also Ref. [8].

Acknowledgements

BLH thanks Prof. Zeilberger for calling his attention to the excellent presentation of the Goulden–Jackson cluster method in Ref. [6]. This work was partially supported by Chinese Natural Science Foundation and the Project on Nonlinear Science.

References

[1] S. Wolfram, Computation theory of cellular automata, Commun. Math. Phys. 96 (1984) 15–57.
[2] Huimin Xie, Grammatical Complexity and One-Dimensional Dynamical Systems, World Scientific, Singapore, 1996.
[3] Bai-lin Hao, Wei-mou Zheng, Applied Symbolic Dynamics and Chaos, World Scientific, Singapore, 1998.
[4] Bai-lin Hao, Hoong-Chien Lee, Shu-yu Zhang, Fractals related to long DNA sequences and complete genomes, Chaos, Solitons and Fractals 11 (2000) 825–836.
[5] G. Deckert et al., The complete genome of the hyperthermophilic bacterium Aquifex aeolicus, Nature 392 (1998) 353–358.
[6] J. Noonan, D. Zeilberger, The Goulden–Jackson cluster method: extensions, applications and implementations, downloadable from `http://www.math.temple.edu/~zeilberg`
[7] I. Goulden, D.M. Jackson, An inversion theorem for cluster decomposition of sequences with distinguished subsequences, J. London Math. Soc. 20 (1979) 567–576.
[8] L.J. Guibas, A.M. Odlyzko, Periods in strings, J. Combin. Theory A 30 (1981) 19–42.
[9] N.J.A. Sloane, and S. Plouffe, The Encyclopedia of Integer Sequences, Academic Press, New York, 1995; and `http://akpublic.research.att.com/~njas/sequences`