Chaos and Fractals: A Computer Graphical Journey
C.A. Pickover (Editor)

CHAOS GAME VISUALIZATION OF SEQUENCES

H. JOEL JEFFREY
Computer Science Dept., Northern Illinois University, DeKalb, IL 60115

Abstract—Iterated function systems can be used to produce a representation of complex images. It was recently shown that one can produce visual representation of the structure of long (2–100 K) sequences by reversing the IFS technique, using a fixed set of affine maps and having map selection controlled by the sequence. The resulting *Chaos Game Representation* (*CGR*) is a 1-1 map between plotted points and subsequences. Nonuniformity of distribution of subsequences produces nonuniformity in the CGR. The CGR is a picture of the sequence, often with visually striking features corresponding to sets of subsequences, thus a new set of questions and areas of investigation. This paper presents the technique, several examples using DNA sequences as examples of nonrandom sequences whose structure is of great independent interest, algorithms for approximating arbitrarily closely subsequences corresponding to observable features of the CGR, and a description of a program based on these algorithms that has useful for exploring features of CGRs. Finally, several extensions to the basic CGR algorithm are proposed.

INTRODUCTION

It seems to be very difficult for human beings to develop intuition for long sequences of letters or numbers. We seem to be very good at recognizing and using patterns and structures, and comparatively poor with long sequential presentations. (A unique discussion of this fact, with some very far-reaching implications may be found in [5].) Converting a sequence, which is basically a uni-dimensional set of inputs, into a graphical form that presents the data in a way that allows visual grasp of the overall pattern, as well as detailed analysis, expands our capabilities as investigators in two ways. It makes standard investigations easier or more efficient, and allows entirely new questions and areas of investigation to develop. An excellent example of this phenomenon is the Mandelbrot set. The graphical presentation of the set is both highly motivating and provocative of many questions that quite literally could not occur without it (*e.g.*, what is the angle of any of the smaller copies of the large cardiod?).

Chaos game representation [4] is a technique for converting the one-dimensional sequence into a two-dimensional form that preserves subsequence structure while providing a provocative visual representation. Further, CGR can be used to approximate any set of subsequences in terms of other sequences with a known structure.

1. THE CHAOS GAME

The chaos game is an algorithm which allows one to produce pictures of fractal structures using paper and pencil or, obviously, a computer. In simplest form, it proceeds as follows:

1. Locate three dots on a piece of paper as the vertices of a triangle, and label them with the numerals 1 through 6 (1 and 2 for the first vertex, 3 and 4 for the second, 5 and 6 for the third).
2. Pick a point anywhere on the paper and mark it. This is the initial point.
3. Roll a 6-sided die. Since the vertices are labelled, the number that comes up on the die is a label on a vertex. Place a mark half way between the previous point and the indicated vertex. For example, if 3 is rolled, place a mark on the paper half way between the previous point and the vertex labelled "3."
4. Continue to roll the die, on each roll marking the paper at the point half way between the previous point and the indicated vertex.

This procedure is the *Chaos Game* [1]. In general, the mathematical subset of the plane defined in the limit, is known as the *attractor*. Figure 1 is a picture of this attractor after 50,000 repetitions; it is the Sierpinski Triangle.

The result of the Chaos Game on other than three points is not, as one might think, a four-, five-, etc., pointed Sierpinski-like figure. For five points, six, or seven initial points the chaos game produces a figure with visible patterns (pentagons within pentagons, a striated hexagon, or heptagons within heptagons), but for eight or more point the game yields essentially a filled-in polygon, except that the center is empty.

With four initial points, however, the result is different: It is a square uniformly and randomly filled with dots.

1.1. *Iterated function systems*

Mathematically, the chaos game is described by an *iterated function system* (IFS)[1]. An IFS is a set of pairs of linear mappings, each pair of the form $x = ax + by + e$, $y = cx + dy + f$. Each pair of maps gives the function for computing the new value of the x and y coordinates. If vertices are at (0, 0), (0, 1), and (1, 0), and 3 is rolled, vertex 2 is indicated, and the coordinates of the new point are given by $x = 0.5 \cdot (x + 0) = 0.5x$, and $y = 0.5 \cdot (y + 1.0) = 0.5y + 0.5$.

With three vertices, and one map per coordinate, we need six maps. We can write the maps in a compact

Reprinted from Comput. & Graphics Vol. 16, No. 1, pp. 25-33, 1992

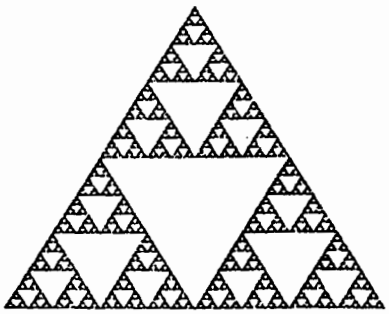

Fig. 1. The result of the chaos game on three points.

tabular form as follows. We let $w(x, y) = (ax + by + e, cx + dy + f)$, and then the map is given by the six coefficients a through f. For the Sierpinski triangle, the maps are:

w_1	0.5	0	0	0.5	0	0
w_2	0.5	0	0	0.5	0	0.5.
w_3	0.5	0	0	0.5	0.5	0.5

Since the choice of map is determined by a die (or random number generator), each map has an associated probability, all equal in the case of the unloaded die. If the probabilities are not equal, the shape of the attractor is unchanged, but the shading may be [1].

In tabular form, including the probabilities, we can use the compact notation, which is known as *the IFS code* (Table 1). Table 2 presents the IFS code for the filled-in square.

2. NONRANDOM SEQUENCES

Quite by chance, the author and a colleague (G. M. Henry) discovered that in these cases the random number generator can make a very significant difference. As noted above, with a good random number generator, the Chaos Game on 8 points produces an almost-filled octagonal. However, when the game is played using Turbo Pascal 3.0, which has a flawed random number generator [6], elaborate patterns are visible, resembling a circle within a circle, the circles connected by 8 (or, respectively, 16) spidery lines.

Further, not all flawed random number generators produce a visible pattern from Chaos Game; using DOS Basic (Version 2.1) RND, which is a quite poor random number generator, the Chaos Game on eight points produces no visible patterns.

Table 1. IFS code for the Sierpinski Triangle.

w	a	b	c	d	e	f	p
1	0.5	0	0	0.5	0	0	0.33
2	0.5	0	0	0.5	0	0.5	0.33
3	0.5	0	0	0.5	0.5	0.5	0.33

Table 2. IFS code for the filled-in square.

w	a	b	c	d	e	f	p
1	0.5	0	0	0.5	0	0	0.25
2	0.5	0	0	0.5	0	0.5	0.25
3	0.5	0	0	0.5	0.5	0	0.25
4	0.5	0	0	0.5	0.5	0.5	0.25

3. CHAOS GAME REPRESENTATION OF DNA SEQUENCES

Intuitively, nonrandomness means that a sequence has "structure." More mathematically, it means a nonuniformity of subsequences [6]. If a sequence of numbers is used to produce an attractor for an IFS code, as described above, and that attractor has visually observable structure then we have revealed some underlying structure in the sequence of numbers, or some nonuniformity of subsequences. Thus, if the IFS for 16, 8, or 4 points is controlled by a poor (*i.e.*, not very random) random number generator, the attractor displays nonuniformity (*i.e.*, visually observable subsets or features).

Since a DNA sequence can be treated formally as a string composed from the four letters "a," "c," "g," and "t" (or "u"), it is an obvious candidate for testing the CGR to see whether in fact visually interesting features were present.

Specifically, we experimented with several DNA sequences, as follows: Instead of "rolling a 4-sided die," use the next base (a, c, g, t/u) to pick the next point. Each of the four corners of the square is labelled "a," "c," "g," or "u;" if a "c," for example, is the next base, then a point is plotted half way between the previous point and the "c" corner.

Example: The first 6 bases of the GenBank sequence HUMHBB (human beta globin region, chromosome 11) are "gaattc":

1. The first "g" is plotted half way between the center of the square and the "g" corner.
2. The next base, "a," is plotted half way between the point just plotted and the "a" corner.
3. The base "a" is plotted half way between the previous point and "a" corner.
4. Next, "t" is plotted half way between the previous point and the "t" corner, etc.

Plotting these six bases, we obtain Fig. 2.

As with the initial points of the Sierpinski triangle, little significance is visible. However, if we continue for the entire 73,357 bases of HUMHBB, we obtain Fig. 3.

The CGR of HUMHBB is a good example of the point of this visualization technique, for it illustrates a number of the characteristics of CGRs in general, and of a certain class of vertebrate DNA sequences in particular:

1. Perhaps the most obvious characteristic of this CGR is the almost empty area in the upper right quadrant (the g-quadrant). A smaller copy of this "scoop"

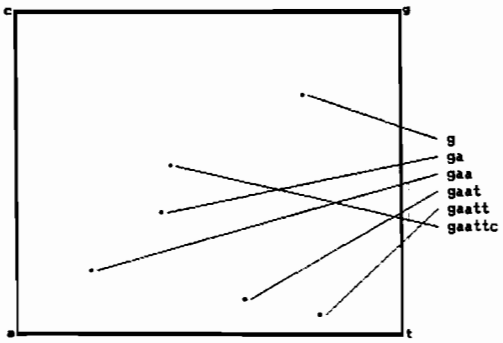

Fig. 2. CGR of the first six bases of HUMHBB.

appears in the upper left, or c-quadrant, presenting a double-scoop appearance. As discussed in the next section, each point in the CGR corresponds to exactly one subsequence (starting from the first base), up to resolution of the screen. Therefore, this graphic pattern indicates repeated patterns in the gene sequence. The same is true of any other visible pattern.

The "double-scoop" corresponds to a comparative sparseness of CG pairs in the sequence; a random sequence with all CG pairs has a CGR with a completely empty double-scoop area. (This is discussed in more detail in Section 4.)

2. Note that any base will always be plotted somewhere in the quadrant with its label since a base is always plotted half way toward its corner.
3. The CGR has self-similarity: There is a copy of the double scoop at the top of the a- and t-quadrants. Further, this continues: If we examine the picture in horizontal "strips" (in halves, quarters, etc.), we see that at the top of each quarter-strip there are four copies; at the top of each eighth-strip there are eight, and so forth.
4. A rather noticeable feature of this CGR is the set of curves on the top of the upper left quadrant of the plot, curving from the solid block up and to the right.

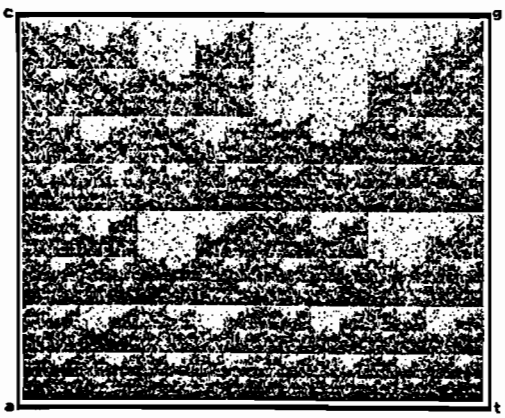

Fig. 3. CGR of human beta globin region on chromosome 11 (HUMHBB) (73,357 bases).

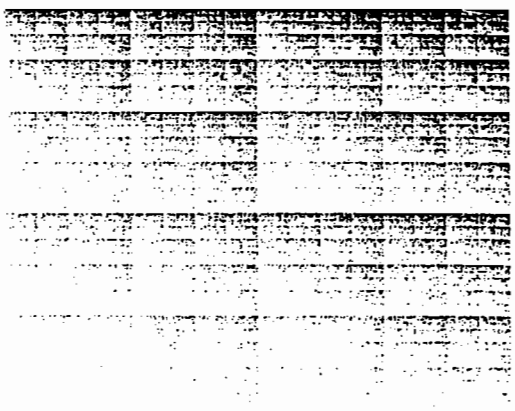

Fig. 4. The shaded filled-in square (see Table 3).

5. The self-similarity and division into quadrants, subquadrants, etc., is a consequence of the non-uniform probabilities of subsequences in general, rather than a particular property of DNA sequences, as may be seen by examining Fig. 4, the attractor for the IFS in Table 3 [1].

The usefulness of the CGR for investigating DNA sequence structure is currently under investigation.

4. PROPERTIES OF THE CGR OF A SEQUENCE

In this section we discuss the relation between the CGR and the sequence itself. We shall assume that the sequence is a string over an alphabet of four letters $\{a, b, c, d\}$, which label a square beginning at the lower left corner in clockwise order.

THEOREM 1. *There is a one-to-one map between the sequence and the interior of the square, in which the k-th point plotted on the CGR of a sequence corresponds to the first k-long initial subsequence of the sequence, and no other subsequence (up to the resolution of the screen). Thus, there is a one-to-one correspondence between the subsequences (anchored at the start) of a sequence and points of the CGR.*

Proof. (By induction on k.) Assign coordinates of $(0,)$, $(0, 1)$, $(1, 1)$, and $(1, 0)$ to the a-, b-, c-, and d-vertices of the square, respectively. For $k = 1$, the only point present is located at $(\frac{1}{4}, \frac{1}{4})$, $(\frac{3}{4}, \frac{1}{4})$, $(\frac{3}{4}, \frac{3}{4})$, or $(\frac{3}{4}, \frac{1}{4})$. Conversely, if a point is at $(\frac{1}{4}, \frac{1}{4})$, it can only have come from the letter "a" (and similarly for "b," "c," or "d").

Now suppose we have a point at location (x, y) and assume (without loss of generality) that next letter is

Table 3. IFS code for the shaded fill-in square.

w	a	b	c	d	e	f	p
1	0.5	0	0	0.5	0	0	0.1
2	0.5	0	0	0.5	0	0.5	0.2
3	0.5	0	0	0.5	0.5	0	0.3
4	0.5	0	0	0.5	0.5	0.5	0.4

a. Then the next point is at $(x/2, y/2)$. Conversely, if a point corresponding to (x, y) is present in the CGR, there is another point (u, v) such that $(x, y) = w_i(u, v)$, $i = 1, 2, 3$, or 4. By the induction hypothesis (u, v) corresponds to the initial k-long sequence s. Therefore, (x, y) corresponds to the $(k + 1)$-long sequence sa, sb, sc, or sd, (depending on whether $i = 1$, 2, 3, or 4, and we have the result.

The definition of the chaos game is that a point is plotted half-way between the previous point and the vertex with its label; mathematically, the new x-coordinate is $\frac{1}{2} \cdot x_{\text{old}}$ (for letters "a" and "b") or $\frac{1}{2} \cdot (x_{\text{old}} + 1)$ (for "c" and "d"). Therefore, any sequence is always plotted somewhere in the quadrant of square with that label: for any string s, sa is plotted somewhere in the a-quadrant, sb in the b-quadrant, etc. Conversely, any two points in the same quadrant must have the same last letter. Further, the notion of quadrant is recursive; each quadrant can be divided into quadrants, etc. Thus, in Fig. 2, "g" is plotted in the g-quadrant, and then "a" is plotted in the upper right subquadrant of the g-quadrant (the one with its upper right vertex at the center of the entire figure), or what might be called the "ga" subquadrant. Any point in the square would be mapped to this subquadrant. Thus, "a" produces a copy of the g-quadrant that is one-half the size (side length) of the g-quadrant, or one-fourth of the size of the entire picture. The next "a" then produces a one-half size copy of the "ga" subquadrant, the "gaa" sub-subquadrant, and then "t" produces a "gaat" sub-sub-subquadrant.

Further, due again to fact that a base is plotted in its quadrant, the converse holds as well: If two points are within the same quadrant, they correspond to sequences with the same last base; if they are in the same subquadrant, the sequences have the same last two bases; if they are in the same sub-subquadrant they have the same last three bases, etc.

Thus, we have the following:

THEOREM 2. *In a CGR whose side is of length* 1, *two sequences with suffix of length k are contained within the square with side of length* 2^{-k}. *Further, the center of the square is given by the following recursive definition:*

1. *The center of the suffix of* 0 *length is* $(\frac{1}{2}, \frac{1}{2})$
2. *If the center of the square containing sequences with suffix w is at* (x, y), *then*
   - *(a) the center of the square containing sequences with suffix wa is* $(x/2, y/2)$;
   - *(b) the center of the square containing sequences with suffix wb is* $(x/2, (y + 1)/2)$;
   - *(c) the center of the square containing sequences with suffix wc is at* $((x + 1)/2, (y + 1)/2)$;
   - *(d) the center of the square containing sequences with suffix wd is at* $((x + 1)/2, y/2)$.

(In the subsection below, "Arbitrary Subsequences," we shall see that the converse holds as well: Each subquadrant of size $\frac{1}{2}^k$ with center is given by the above definition corresponds exactly to those sequences that terminate in the k-long sequence given in the definition.)

The resolution of most monitors or printers is such that the points for sequences with identical suffixes of length over 10 (maximum) are superimposed. However, any portion of the CGR may be magnified to any degree, revealing finer structure of the CGR and therefore of the sequence. Thus, there is no limit on the length of subsequences that may be displayed.

5. SIERPINSKI CARPETS

5.1. *AC- and BD-deficient sequences*

Figure 5 shows the CGR of a 50,000-letter sequence with all "ac" subsequences removed. It is similar to a Sierpinski carpet [3]. We shall use this figure as examples and motivation for the discussion (in the following section) of finding the sets of subsequences corresusing visual features of a CGR.

The large empty square, of side length $\frac{1}{4}$, with its lower left corner at the center of the CGR, is the area in which all sequences terminating in "ac" would be plotted. (It is empty because, of course, there are no such subsequences in this sequence.) The square of side length $\frac{1}{8}$, with its lower left corner at the upper right corner of the largest empty square, is represents all "acc-"terminated sequences. Similarly, the square of side $\frac{1}{16}$, next on the diagonal toward the c-corner, represents all sequences "accc"-terminated subsequences, and so on. Figure 6 shows this set of subquadrants.

It is very instructive to examine another set of subquadrants, the set in the upper right subquadrant of the lower right quadrant (or, the c-subquadrant of the d-quadrant) (see Fig. 7). In this subquadrant we see a $\frac{1}{2}$-size copy of the "ac-," "acc-," "accc-," etc. subquadrants marked in Fig. 6. The largest of this set corresponds to all subsequences terminating in "acd;" the next to all terminating in "accd," the next to "acccd," and so on. A similar set, each a $\frac{1}{2}$-size copy of the set in the c-quadrant, is found in the c-subquadrant of the b-quadrant and the c-quadrant of the a-quadrant.

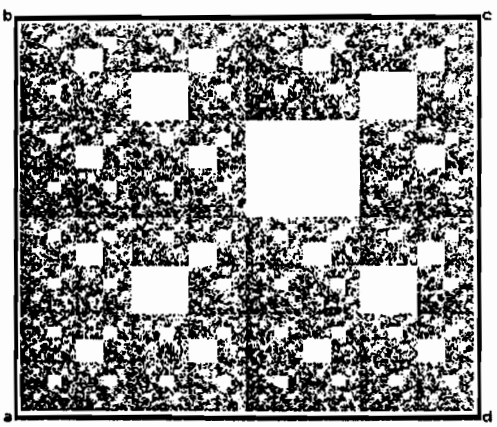

Fig. 5. Random sequence without "ac" subsequences.

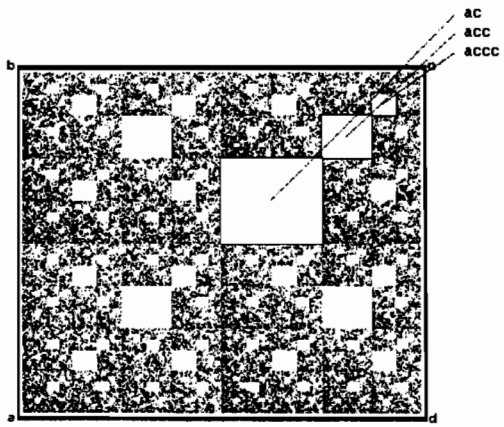

Fig. 6. Empty squares and their corresponding subsequences.

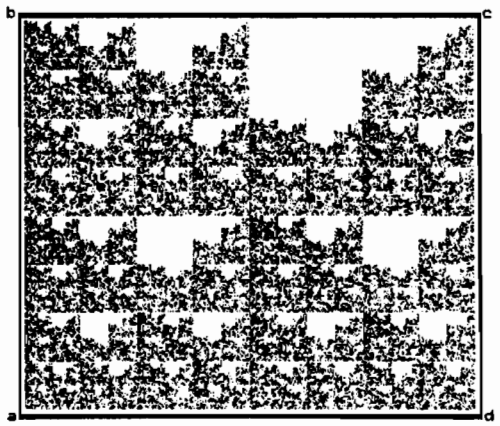

Fig. 8. Random sequence without "bc" subsequences.

In general, close examination of Fig. 5 reveals that in every subquadrant, of side length $\frac{1}{2^k}$, $k = 1, 2, \ldots$, there is a complete copy of *the entire CGR*.

The CGR of a random sequence with no "bd" subsequences shows the same form, but the sequences of empty area, in decreasing size order, go up and to the left.

5.2. *BC- and AD-deficient sequences*

The CGR of a sequence with no "bc" or "ad" subsequences (*i.e.*, letters from the same side of the square) has a similar set of empty areas as discussed in the above subsection. However, when the letters of the missing subsequence share a side of the square, the boundaries of the missing areas are colinear (as opposed to above case, when the boundaries share only the single corner point). As a result, although the same size areas are missing, the boundaries of the areas are fractal, and the attractor appears quite different from Fig. 5. Figure 8 shows the CGR for a 50,000 letter sequence with no "bc" subsequences. Figure 9 is the same CGR, but with areas corresponding to various missing subsequences noted.

5.3. *Arbitrary subsequences*

Suppose we have a subsequence s of interest. Let us call this the *primary subsequence.* (We might, for instance, be interested in whether s is more or less frequent in the sequence than average.) Define the "primary image" (denoted by the symbol □) to be the square of size $\frac{1}{2^{|s|}}$ (where $|s|$ denotes the length of s) given by the recursive definition of Theorem 2. If there are no subsequences s in the sequence, the primary image is empty; if there are additional instances of s, the primary image will be denser than average.

If the center of the primary image is (x, y), and it has side length $\frac{1}{2^k}$, we can define the a-image of the primary image to be the subquadrant whose center is $(x/2, y/2)$, and whose side length is $\frac{1}{2^{k+1}}$. The b-, c-, and d-images have the same side length and are centered at $(x/2, (y + 1)/2)$, $((x + 1)/2, (y + 1)/2)$, and $((x + 1)/2, y/2)$, respectively.

Thus, there is a point in the a-image of the primary square if and only if there is a point in the primary image that is mapped into the a-image by the a-map.

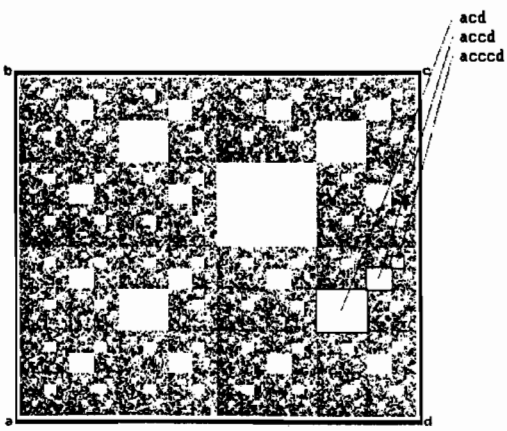

Fig. 7. Half-size copy of the squares marked in Fig. 6.

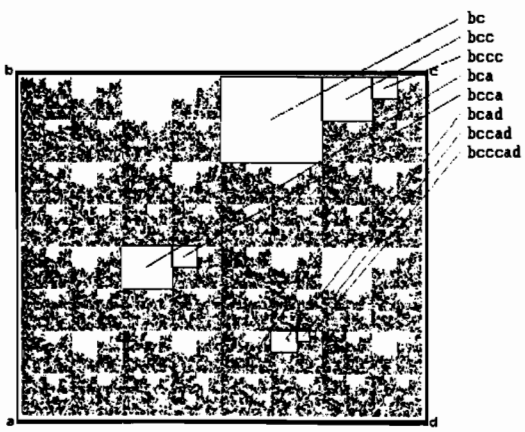

Fig. 9. Subsequences corresponding to empty areas of Fig. 8.

This in turn can only occur if the subsequence *sa* occurs in the sequence. Thus, the points in the a-image of the primary image are the a-maps of all points in the primary image, and similarly for the b-, c-, and d-images of the primary square.

Now let us consider a 2-letter extension of s, say *sab*. The subsequences of this form in the sequence produce a set of points in the b-map of the a-map of the primary image, or $\bar{b}(\bar{a}(\square))$. Thus, *ab* produces an image of the primary image of size $\frac{1}{4}$ of the primary image size, whose center is defined by Theorem 2.

In general, every sequence t defines an image of the primary image, of size $\frac{1}{2^{|t|}}$. If there are no instances of the primary sequence present in the sequence, then the primary image and all t-images of the primary image will be empty, for all sequences t.

For any sequence s of length greater than 1 we can therefore define a CGR in Cantor set-like fashion as follows:

1. Remove the s-image of the unit square. This is the primary image.
2. For each $k > 0$,
   For each string t of length k,
   Remove the t-image of the primary image.

It should be noted that if t is of the form xa, then the t-image of the primary image is the a-image of the x-image of the primary image, and therefore the above algorithm may be formulated recursively.

6. SUBSEQUENCES CORRESPONDING VISUAL FEATURES

It is common, when using the CGR to investigate a set of sequences (*e.g.*, a set of DNA sequences for the same protein in several species) to find a feature of the CGR and want to identify the sets of subsequences that correspond to the feature. Figure 10, for example, shows the CGR of 15 concatenated coding sequences in the human globin region on chromosome 11. (Not all the DNA sequence actually codes for proteins.) An easily identifiable feature is the roughly oval area marked on the figure in which the density of the points is several times that of the CGR as a whole. This indicates a set of subsequences several times more frequent than average; we would like to be able to easily identify the subsequences corresponding to this feature.

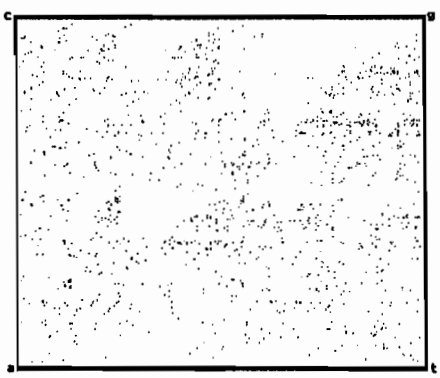

Fig. 10. Fifteen concatenated coding sequences of HUMHBB.

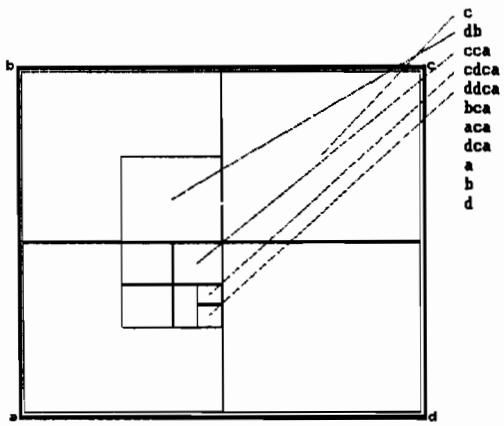

Fig. 11. Subquadrant–subsequence correspondence.

6.1. *Approximation of features*

Recall from the above section that the a-image of the primary square is the $\{pvbarp = w_a(q)$, for some q in the CGR, that the a-image is contained in the a-quadrant, and similarly for the b-, c-, and d-images. The points in the a-quadrant correspond to any subsequence ending in "a," and conversely any subsequence ending in "a" produces a point in the a-quadrant. The a-quadrant is thus the a-image of the entire CGR, and similarly for the b-, c-, and d-quadrants. Figure 11 illustrates the fundamental correspondence of string to subquadrant.

Let us denote by $\bar{a}$ the map corresponding to a, and similarly for $\bar{b}$, $\bar{c}$, and $\bar{d}$. Then any sequence $s = s_1 s_2 \cdots s_n$ defines a map $\bar{s}_n \bar{s}_{n-1} \cdots \bar{s}_1$, such that if the sequence t corresponds to point p of the CGR, the sequence ts corresponds to $\bar{s}(p) = \bar{s}_n \bar{s}_{n-1} \cdots \bar{s}_1(p)$.

From the definition of the IFS maps, if the CGR is based on the unit square, the square of side length 1, with lower left corner = (0, 0) (as in Theorem 2), then the center is at $(\frac{1}{2}, \frac{1}{2})$, and centers of the a-, b-, c-, and d-quadrants are at $(\frac{1}{4}, \frac{1}{4})$, $(\frac{1}{4}, \frac{3}{4})$, $(\frac{3}{4}, \frac{3}{4})$, and $(\frac{3}{4}, \frac{1}{4})$, respectively. The center of the aa-subquadrant is then at $(\frac{1}{8}, \frac{1}{8})$, the ca-subquadrant (the upper-right, or c-, subquadrant of the a-quadrant) is at $(\frac{3}{8}, \frac{3}{8})$, etc. Using binary representation, we have Table 4.

The center of each subquadrant is represented by two bit strings, the x- and y-coordinates in binary with the binary point dropped.

Since the corners of the CGR are at (0, 0), (1, 0), (1, 1), *and* (1, 0), only two operations are involved

Table 4. Centers of quadrants, binary representation.

Quadrant	Center
a	(.01, .01)
b	(.01, .11)
c	(.11, .11)
d	(.11, .01)
aa	(.001, .001)
ca	(.011, .011)

in finding a new point in the CGR, whether the center or corner of a subquadrant or any other point: multiplying by $\frac{1}{2}$ and multiplying by $\frac{1}{2}$ and then adding $\frac{1}{2}$. The $\bar{b}$ map of (x, y) is $(x/2, y/2 + \frac{1}{2})$; $\bar{c}$ is $(x/2 + \frac{1}{2}, y/2 + \frac{1}{2})$; $\bar{d}$ is $(x/2 + \frac{1}{2}, y/2)$. Multiplying by $\frac{1}{2}$ is represented in binary by prefixing a 0, and if the left-most bit of a number is 0, then adding $\frac{1}{2}$ is represented by simply changing that 0 to 1. Therefore, the combination of multiplying by and adding $\frac{1}{2}$ is represented by prefixing a 1. Therefore, if a location is given by bit strings x and y, we can describe the $\bar{a}$, $\bar{b}$, $\bar{c}$, and $\bar{d}$ maps as shown in Table 4.

Now let us consider the converse: Suppose we have a point p whose coordinates are given by bit-strings x and y (*e.g.*, 101, 011). The left-most bits of x and y are 1 and 0, respectively. Examining Table 5, we see that the only map that can produce this pair of leftmost bits is $\bar{d}$, and thus $p = \bar{d}(01, 11)$. Continuing, if q is the preimage of p, then the leftmost bits of q are 0 and 1, so $q = \bar{b}(1, 1)$. Similarly, $(1, 1) = \bar{c}(0, 0)$. Thus, $(101, 011) = \bar{d}(\bar{b}(\bar{c}((0, 0))))$, which means that (101, 011) is the point corresponding to subsequence "cbd." (Note the reversal as we go to the string from the composition of maps.) The correspondence of bit-pairs to maps is summarized in Table 6.

6.2. *Algorithms*

Based on the above analysis, we can define algorithms for locating, resizing, and moving a box corresponding to any subquadrant of a CGR, as follows:

1. Define the initial subsequence to be "a."
2. To draw a box corresponding to a subsequence, find the corners of the box by finding the image of each of the corners of the CGR under the map corresponding to the sequence: If $s = s_1 s_2 \cdots s_n$, then $\bar{s}((0, 0)) = \bar{s}_n \bar{s}_{n-1} \cdots \bar{s}_1((0, 0))$, and similarly for $(0, 1)$, $(1, 1)$, and $(1, 0)$.
3. To move a box,
   (a) Convert the subsequence corresponding to the current box to the corresponding x and y bit-strings.
   (b) Add/subtract 1 to/from the x and/or y bit-strings:
   Up: Add 1 to the y bit-string
   Down: Subtract 1 from the y bit-string
   Left: Subtract 1 from the x bit-string
   Right: Add to the x bit-string.
   (c) Convert the new x and y bit strings to a sequence of letters.
   (d) Redraw the box.
4. To enlarge a box, delete the rightmost letter of the subsequence, and draw the box corresponding to the sequence.

Table 5. CGR maps in binary.

$\bar{a}(x, y) = (0x, 0y)$
$\bar{b}(x, y) = (01, 1y)$
$\bar{c}(x, y) = (1x, 1y)$
$\bar{d}(x, y) = (1x, 0y)$

Table 6. Map to bit-pair correspondence.

Map	Bit pair
$\bar{c}$	0, 0
$\bar{b}$	0, 1
$\bar{c}$	1, 1
$\bar{d}$	1, 0

5. To shrink a box, append "a" to the subsequence and redraw the box corresponding to the sequence. (Any letter could have been used; "a" is an arbitrary choice.)

These algorithms only allow boxes to be defined and moved in units of quadrants, subquadrants, sub-subquadrants, etc. In other words, one cannot use them to define a box of side length $\frac{1}{2}$ centered at the point $(\frac{1}{2}, \frac{1}{2})$. Such a box would not correspond to a single subsequence; the sequences that correspond to this box are the four subquadrants corresponding to subsequences "ag," "ct," "ga," and "tc" (Fig. 12).

The above algorithms allow the definition of a set of subquadrants of any size, each corresponding to a set of subsequences with a fixed suffix, that approximate any area (or combination of areas) of a CGR as closely as desired. In essence, the feature of interest is covered with a set of squares of varying sizes, approximating the feature as desired.

6.3. *Description of program*

The above algorithms have been implemented in a working program for analyzing CGRs. The program is approximately 1,600 lines of Turbo Pascal 5.0, running on an IBM AT or PS/2 with EGA or VGA monitor. The program produces a CGR from a user-specified sequence file and then allows the user to define and move a box on the CGR. The eight outer keys of the keypad permit movement of the box in any direction (including diagonals). A user may at any point elect to retain a box; it is redrawn in another color and the subsequence defining the box is added to an on-

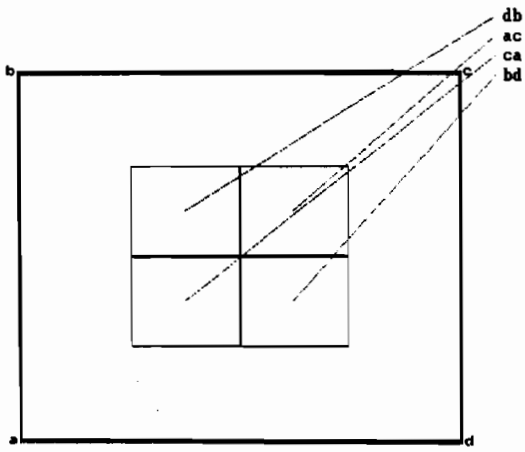

Fig. 12. Square not corresponding to a single subsequence.

screen list. This on-screen list may be edited, and deleted sequences (and their corresponding boxes) are removed.

Figure 13 demonstrates the use of the CGR analysis program to find the subsequences that define features of a CGR. In lower-left quadrant of the CGR of the human alpha-1 antitrypsin gene (HUMA1ATP) is an irregularly shaped figure that is visibly sparser. The figure shows that the area is approximated by the subquadrants for six different sequences, from four to six bases in length. (Most of the figures in this paper were produced with this program.) It would be very difficult to recognize such a set of sequences via ordinary statistical analysis methods, for the area does not correspond to a single comparatively rare subsequence (such as a single tetranucleotide, for example). Rather, it corresponds to a set of comparatively rare sequences, of different lengths. Finding such sets might be possible in principle, by examining or otherwise processing sets of frequency counts, but some of the sets are large enough to effectively preclude effective human pattern recognition. (The whole point of the CGR, and other visualization techniques, is to avoid long sequences of data.) For example, there are $4^5 = 1{,}024$ sequences of length 5, and $4^6 = 4{,}096$ of length 6. Further, the "tgatca"-square is part of the sparse area. It is difficult to see how such a set of sequence characteristics might be discovered by examining these large sets of frequency counts other than via CGR analysis.

Figure 14 is a good example of using the CGR, together the CGR approximation algorithm, to recognize and analyze features of a sequence that in all likelihood would not occur without it. Figure 14 is the CGR of the amino acids of the bacteriophage PT7. Note the somewhat star-shaped region approximately one quarter of a side length from the right edge. This region results from the absence of a somewhat complex set of subsequences of amino acids missing from the DNA sequence. Traditional statistical analysis would be unlikely to reveal such a feature, as it produces statistics on the relative frequency of subsequences (usually of a single length), and the star shape results from the absence of several subsequences of several lengths.

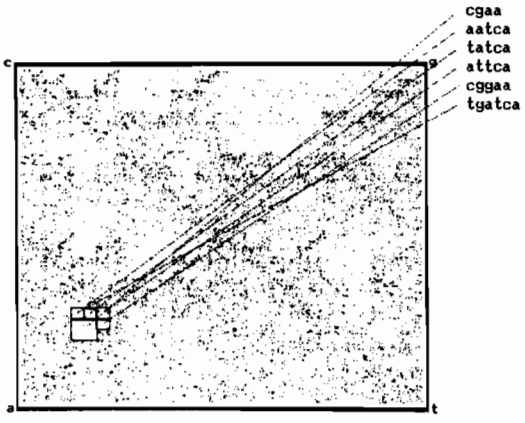

Fig. 13. Approximating an irregular feature of HUMA1ATP.

Fig. 14. CGR of bacteriophage PT7 (using amino acids).

7. RELATED WORK

Symbolic dynamics[2], a topic in dynamical systems theory, associates strings of symbols with orbits of a dynamical system. It is a powerful tool for analyzing the orbits. It would be interesting to find out whether this approach can be reversed, finding dynamical systems whose behavior is represented by a given sequence.

To date we have relied solely on visual characterization of the patterns found in CGRs, both as to recognizing features and judging similarity of features. An objective, mathematical measure is needed. The pattern recognition literature reveals little in the way of a formal definition or characterization of "pattern" that is directly applicable. However, digital image enhancement techniques are clearly applicable. The Hausdorff distance[1] has been used to formalize similarity of patterns, and may prove useful in defining similarity of CGRs.

To our knowledge CGR is the first attempt to use particular sequences of numbers (or symbols) to control the chaos game, and thereby analyze/characterize the sequences themselves. Since the approach is new, so far as we know, we conclude with some open questions and extensions of the technique, most of which have been barely, if at all, explored. (We invite the reader to take this list as provocative, in the spirit of initiating a brain-storming session, rather than in any way complete or exhaustive.)

8. EXTENSIONS AND OPEN QUESTIONS

8.1. *Pattern recognition and similarity*

To date, all work with CGRs has relied on using them as an aid to human intuition, presenting a great deal of information visually. We do not view this as a fault, as visualization of scientific information is, in our view, a valuable and growing field. However, there is no reason to limit CGRs to this use. Specifically, the great body of image enhancement and pattern recognition work can be applied to CGRs. These approaches may well improve the usefulness of the CGR as a visualization method, and provide the obvious possibilities of standard feature extraction.

The above-mentioned Hausdorff distance as a measure of similarity of CGRs is a formalization related to these developments that seems quite useful. For ex-

ample, it may be that one can find CGRs that are representative of groups of DNA sequences, and the Hausdorff distance from an unknown DNA sequence to the representative CGRs provides information as to what the unknown sequence codes for.

In some applications this may be a necessary step in applying CGRs to a large body of sequences. For example, biologists typically believe this step is necessary. We feel a cautionary note is in order, however: human visual processing is far superior to any currently known technology, so there well be human-recognizable characteristics of CGRs that are resistant to current technology.

8.2. *Sequences over larger alphabets*

The CGR technique presented in this paper works only for sequences over an alphabet of four letters. For larger alphabets of size $n > 4$, one might think of defining the CGR on a regular n-gon. Although this can be done, it does not provide a good visualization of the sequence, because a random sequence on an n-gon does not produce a filled-in polygon.

The following approach does work: Divide the unit square (as with the standard CGR) into n equal nonoverlapping portions covering the square (*i.e.*, a tiling of the square). This tiling defines n maps of an IFS code in which map k defines the mapping of the unit square onto tile k. A random input sequence then produces a filled-in unit square, as with the standard CGR, and so any nonuniformity in the input sequence produces a nonuniformity in the density of areas of the CGR, and vice versa. We therefore have the same conditions that permit the standard CGR to represent a sequence on four letters.

We have experimented with this approach in DNA sequences. In a DNA sequence three bases (letters) define one amino acid; such ordered triples are known as codons. Each codon defines one amino acid (the building blocks of proteins), although there are only 20 amino acids; several codons code for the same amino acid. In addition, there are three "stop codons" and a particular "start" codon which control cellular machinery that produces the protein based on the DNA sequence, so a total of 24 equivalence classes of codons are present. We have produced a CGR in which the unit square is divided into 24 (4 by 6) squares, and the CGR is produced by 24 mappings onto these squares. If two codons code for the same protein, the same map is used. The result is a CGR with identifiable areas of greater or lesser density. We have also produced a CGR using only the codons for the 20 amino acids, with a 4×5 tiling of the square. The usefulness of this approach to biological investigations remains unknown.

8.3. *Natural language representation*

The input sequence for a CGR can be any sequence. The CGR analysis program discussed above ignores any input letters not in the four-element alphabet. English text files can, and in fact have been used produce CGRs with visual features. (As illustrated in [4], such CGRs are unlike those of DNA sequences.) An application in which this might prove useful is in establishing authorship of texts. One part of such work depends on syntactic and stylistic characteristics of texts, rather than semantics or historical references. It is conceivable the CGRs of works of various authors, or bodies of texts, may show distinctive characteristics.

The standard CGR is in certain respects poorly suited to this application, as it is based on only four letters, but the extension to larger alphabets discussed above addresses this difficulty.

A somewhat different extension would be to very large sets of maps, perhaps 100 (10×10 tiling) or 10,000 (100×100 tiling). This would allow plotting of points corresponding to words, and the result would be a visualization of the sequences of words used. Since in some sense a grammar of defines the possible legitimate sequences of words of the language, this graphical representation of text may prove useful in natural language processing in general, as well as authorship analysis.

REFERENCES

1. M. F. Barnsley, *Fractals Everywhere,* Springer-Verlag, New York (1988).
2. R. L. Devaney, *An Introduction to Chaotic Dynamical Systems,* Addison Wesley Publishing, Redwood City, CA (1989).
3. J. Feder, *Fractals,* Plenum Press, New York (1988).
4. H. J. Jeffrey, Chaos game representation of genetic sequences. *Nucleic Acids Research* **18**(8), 2163–2170 (1990).
5. P. G. Ossorio, *"What Actually Happens"—The Representation of Real World Phenomena,* University of South Carolina Press, Columbia, SC (1978).
6. S. K. Park and K. W. Miller, Random number generators: Good ones are hard to find. *Com. of the ACM,* **31**(10), 1192–1201 (1988).