Computing and Informatics, Vol. 28, 2009, 695–710

INTERACTIVE VISUALISATION OF OLIGOMER FREQUENCY IN DNA

Matej Makula

Faculty of Informatics and Information Technology
Slovak University of Technology
Ilkovičova 3, 812 19 Bratislava, Slovakia
e-mail: makula@fiit.stuba.sk

Ľubica Beňušková

Department of Applied Informatics
Faculty of Mathematics, Physics and Informatics, Comenius University
Mlynska dolina, 842 48 Bratislava, Slovakia

&
Department of Computer Science, University of Otago
Dunedin, New Zealand
e-mail: benus@ii.fmph.uniba.sk, lubica@cs.otago.ac.nz

Manuscript received 30 November 2007; revised 27 February 2008
Communicated by Vladimír Kvasnička

Abstract. Since 1990, bioinformaticians have been exploring applications of the Chaos Game Representation (CGR) for visualisation, statistical characterisation and comparison of DNA sequences. We focus on the development of a new computational algorithm and description of new software tool that enables CGR visualisation of frequencies of K-mers (oligomers) in a flexible way such that it is possible to visualise the whole genome or any of its parts (like genes), and parallel comparison of several sequences, all in real time. User can interactively specify the size and position of visualised region of the DNA sequence, zoom in or out, and change parameters of visualisation. The tool has been written in JAVA^{TM} language and is freely available to public.

Keywords: Chaos game representation, iterated function systems, bioinformatics, information visualisation, fractal representation

1 INTRODUCTION

Nobel Laureate in Physics Kenneth Wilson[1] is credited to be among the first to acknowledge that computing has become a third basic methodology of natural sciences side by side with the theory and experiment [8]. In addition, scientists from many fields, especially genetics, have discovered information processes in the deep structures of their fields. As a result, another Nobel Laureate David Baltimore[2] in his book *The Invisible Future* considers biology to be an information science: "Biology is today an information science. The output of the system, the mechanics of life, are encoded in a digital medium and read out by a series of reading heads. Biology is no longer solely the province of the laboratory. Contributions come from many directions" [2, p. 45].

The natural digital medium in which all information about an organism is encoded is the DNA (Deoxyribonucleic Acid). DNA is a chemical chain, present in each cell, which consists of ordered pairs of small chemical molecules (bases) which are: Adenine (A), Cytosine (C), Guanine (G), and Thymine (T), linked together by deoxyribose-phosphate nucleic acid backbone to form a double helix. A pair with T, and C pairs with G so by reading one chain of the double helix we know the primary structure of its counterpart immediately. In humans, the DNA is organized into 23 pairs of chromosomes (including the pair of so-called sex chromosomes, either XX or XY). For instance the first 70 bases of the human chromosome 4 read:
ACCCTAACCCTAACCCTAACCCTAACCCTAACCCTACCCTAACCCTAACC
CTTAACCCTTAACCCTAACC ...

The total DNA in the nucleus contained in all chromosomes is called the genome. The central dogma of the molecular biology states that DNA is transcribed into RNA, which is translated into proteins [7]. RNA (ribonucleic acid) has a similar structure as DNA, but here Thymine (T) is substituted by Uracil (U). DNA contains millions of base pairs (in humans about 3 billions), but less than 5 % or so is used for the production of proteins, and these protein-producing segments are called genes. More than 95 % of DNA is the so-called junk DNA, but some prefer the name non-coding DNA, because most of it may have another, for instance regulatory, functions by producing a variety of regulatory RNAs. Some can be an evolutionary waste, but also the source of future evolution of species.

Massive amounts of information keep accumulating into many complex chemical structure databases including genomic data, with almost 3500 microbial and eukary-

[1] For his theory of critical phenomena in connection with phase transitions in 1982.

[2] Shared with R. Dulbecco and H. M. Temin for their discoveries concerning the interaction between tumour viruses and the genetic material of the cell in the field of physiology and medicine in 1975.

otic genomes listed in the Genomes OnLine Database (GOLD)[3], either completed or being sequenced [15]. Increasing interest is now being focused on characterizing various genomes, especially for their repetitive DNA and repeated DNA motifs (i.e., biologically significant patterns), especially in the non-coding regions, important for chromatin condensation and gene regulation [4].

In this context, a flexible tool for a two-dimensional fractal visualisation of large-scale genome biodata is presented and illustrated. It is a compact and efficient visualisation tool based on Chaos Game Representation. The tool is suitable for a fast real-time comparison of huge amounts of genomic data through fractal representation.

1.1 Related Work

CGR method of visualisation of genetic data is not new. In 1990, Jeffrey introduced the principles of graphical representation of genetic structure called Chaos Game Representation (CGR) using Barnsley's iterative function systems [3, 13]. On examples of genetic sequences from invertebrates, plants, molds, phages, bacteria, and viruses, he showed that genes from each species are distinctive by the patterns they produce in CGR. To our knowledge, no one has yet attempted a systematic and exhaustive description of patterns in CGR of genomes/genes/non-coding regions, etc. of different species. The tool we introduce in this paper can be applied by biologists for such a purpose.

Fractal representation can be generalized to become applicable for sequences composed of more than 4 letters, for instance for proteins comprised of 20 amino-acids. For the 2D fractal represenatation of proteins a 20-sided regular polygon and the dividing ratio of 0.135 : 0.865 (instead of 0.5 as in the case of a square) was proposed in [9]. Further compression by means of a 12-sided regular polygon with each vertex representing a sub-sequence of amino-acids (leading to conservative substitutions) was proposed as an alternative to protein 2D visualisation in [5]. Thus it has been shown by means of the graphical representation that different functional classes of proteins follow specific distributions of different mono-, di-, tri-, or higher order peptides (short sub-sequences of aminoacids) along their primary sequences.

Rigorous connection between CGR spatial scaling characteristics and the statistical characteristics of the symbolic sequences themselves was later developed by Tiňo [17]. He has generalized Jeffrey's graphic representation to accommodate (possibly infinite) sequences over an arbitrary finite number of symbol alphabets and established a direct correspondence between the statistical characterisation of symbolic sequences via Rényi entropy spectra and the multi-fractal spectra with the corresponding multi-fractal characteristics of the sequences' spatial representations.

Besides visualisation, CGRs were studied with respect to statistical investigation of genomes and comparison purposes. For instance, in [1], distance between end

[3] `www.genomesonline.org`

positions in the 2D image has been used to measure similarity between the corresponding sequences. In addition, the distribution of positions in the CGR plane was shown to be a generalization of Markov chain probability tables that accommodate non-integer orders. Frequencies of short K-mers of nucleotide bases, called oligomers ("oligo" means few), were used for fast alignment-free comparisons of genomes [11]. However, CGR has been shown to be utilized also for making alignment-based comparisons by identifying all local alignments between two long DNA sequences using the sequence information contained in CGR points [14].

In spite CGR being so popular for fast and efficient visualisation of genomes, only one freely available working tool is available (by the date of writing this paper). It is the SeeDNA program that runs on Red Hat Linux with GTK+ support [16]. It displays 2D or 1D histograms of the K-string distribution of a given sequence and its randomized counterpart. It is also capable of showing the difference of K-string distributions between two sequences.

Our program introduced in this paper is also a tool for visualization of frequencies of K-mers (oligomers) in large genomes by CGR (described in the next Section 2). CGR transforms DNA sequence to the colored image. Every pixel corresponds to a short sequence of K symbols. Color of pixel reflects K-mer frequency in the analyzed sequence. This gives us a histogram that allows seeing sparse and dense occurrences of K-mers quickly. The tool that we have developed and implemented in Java can be run on any platform, e.g. Windows, which for many users may be more suitable than Linux. It uses an original efficient algorithm for implementation of CGR (Section 3) and has several attractive features (like zooming, on-line sliding along the sequence, colour code change, etc.), which we describe in detail in Section 4 of this paper.

2 FRACTAL VISUALISATION OF DNA

Mathematically, the CGR is described by an Iterated Function System (IFS). In general, IFS represents a system that transforms a sequence of symbols into a unique set of points in the 2-dimensional space. In particular, IFS transforms DNA sequences into the 2-dimensional fractal. An important characteristic of the representation space is that there are so-called attractor points in the space, e.g. in the corners of the unit square, representing subsequences AAAA..., CCCC..., and so on. Equation (1) shows the four IFS transformations in the rectangular coordinate space to be aplied to successive bases of the codon DNA:

$$\begin{aligned} \omega_T(x,y) &= (0.5x + 0.5, 0.5y) \\ \omega_A(x,y) &= (0.5x, 0.5y + 0.5) \\ \omega_G(x,y) &= (0.5x, 0.5y) \\ \omega_C(x,y) &= (0.5x + 0.5, 0.5y + 0.5) \end{aligned} \tag{1}$$

where x and y are coordinates in a rectangular unit square. Iteration can start in arbitrary point and a corresponding transformation is applied for each successive

base in the DNA sequence to the previous point x, y. A limit set of points emerging from an infinite application of the IFS is called an IFS attractor. Figure 1 illustrates the IFS principle. It can be easily shown that each transformation contracts coordinates to quarter of the original unit square. This results in an important characteristic, i.e. the relationship between the actual position in the unit space and suffix of sequence processed by the IFS (Figure 1). Thus, in Figure 1a, 'C' is plotted in the C-quadrant. Then 'A' is plotted in the upper right of the A-quadrant, or what might be called the 'CA' sub-quadrant (Figure 1 b)). Thus, 'A' produces a copy of the C-quadrant that is one-half the size (side length) of the C-quadrant, or one-fourth of the size of the entire picture. The next letter 'G' then produces a one-half size copy of the 'CA' sub-quadrant, the 'CAG' sub-sub-quadrant, within the 'AG' sub-quadrant (Figure 1 c)).

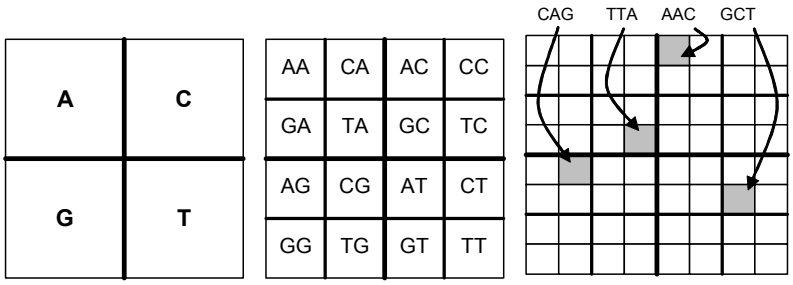


Fig. 1. Correspondence between the final position in the IFS space and the suffix of the processed DNA sequence

Further, due to the 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 sub-quadrant, the sequences have the same last two bases; if they are in the same sub-sub-quadrant they have the same last three bases, etc. For example, in Figure 2 random and biological sequences are visualised. In the genome visualisation sparse and dense areas correspond to rare and frequent subsequences, respectively.

The question of when two points close in the CGR represent similar sequences is a bit complicated. In general, two close points may correspond to different sequences. However, this situation can only occur if the two points, although close, are in different quadrants of the picture. Since a base is always plotted in its quadrant, any sequence will always be plotted somewhere in the quadrant of its last base, and conversely any two points in the same quadrant must have the same last base. Further, the notion of quadrant is recursive; each quadrant can be divided into sub-quadrants, etc. Thus, from the biological point of view, similar oligomers, i.e. small fraction of DNA sequence of fixed size K, are situated spatially close to each other in the visualised quadrant as a result of suffix similarity.

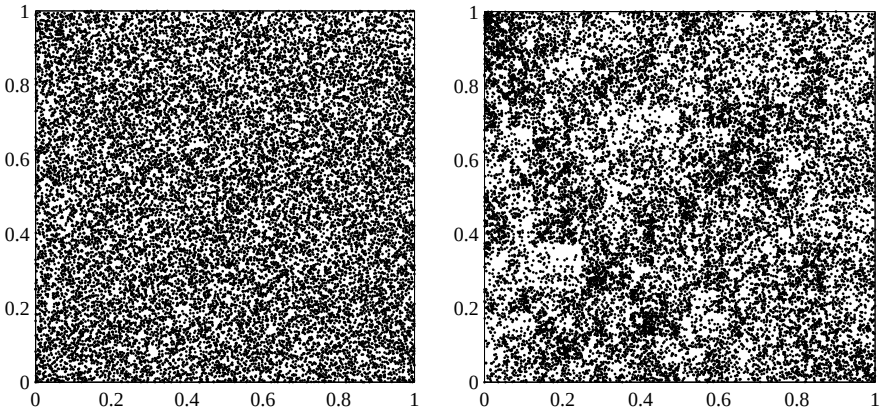
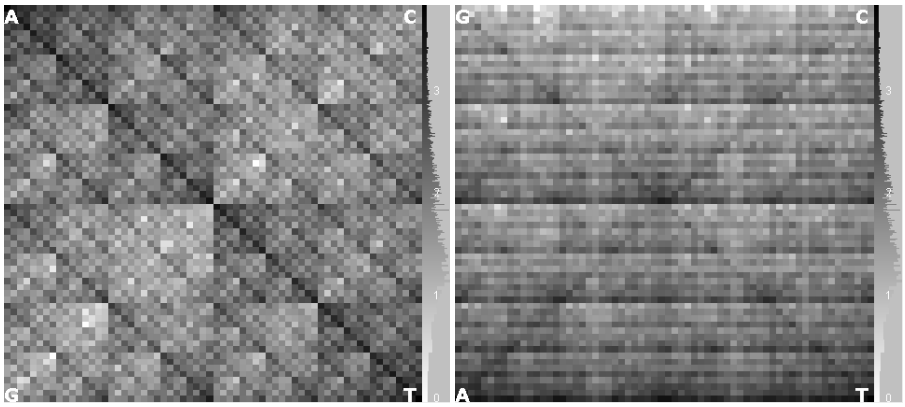


Fig. 2. Chaos game visualisation using Equation (1); a) the first 20 000 letters of random four-symbol sequence; b) the first 20 000 bases of E.coli (the most common intestine bacterium) strain K12 genome. Dense and sparse areas in the right visualisation indicate a non-uniform oligomer distribution in E. coli genome (the total count of bases is 4.639 Mbp (megabasepairs))

Limited precision while rendering image on a computer inevitably causes loss of information. Especially regularities in the form of self-similar subsequences in genomic data result in a lot of overlapping points in visualisation. Practically, only the last 8–10 letters can be identified due to the finite resolution of computer displays. In addition, since in CGR the number of plotted points always equals to the number of bases, very long sequences can potentially fill out most of the plane and no fine details would be resolvable. Straightforward solution to this problem is to divide visualisation space into smaller quadrants, count occurrences of points in each quadrant and visualise their density either in greyscale or colour, as it was suggested in [11, 16].

Probability or frequencies of occurrence of oligomers in arbitrary DNA sequence are not equal and they also reflect specific biological characteristics. When the division of CGR space is based on the layout from Figure 1, i.e. the space is divided into $2^K \times 2^K$ regions, frequencies of all oligomers of length K in analysed sequence will be displayed in the resulting plot, which also forms a fractal (see Figure 3). Another important part of the CGR visualisation, which was not mentioned earlier, is the layout of image related to the location of the so-called attractor points. These points represent subsequences of identical bases, e.g. AAAA..., CCCC..., etc. and are located in the corners of visualisation space. Positions of attractors can be easily rearranged by changing bases assignments in transformations (Equation (1)). Different assignments can significantly change the resulting plot.

In Figure 3 two different layouts for visualisation of Aster Yellows Witches' Broom phytoplasma genome were chosen. Phytoplasmas are wall-less prokaryotic



a) b)

Fig. 3. The images represent oligomer frequency fractals created by the CGR visualisation of Aster Yellows Witches' Broom phytoplasma genome for different corner base attractors, from left to right: a) A, C, G, T, b) G, C, A, T. Black colour is for more frequent oligomers, and white colour stands for less frequent oligomers ("oligo" means few, in this case $K = 6$).

microbes and obligate parasites of plants. They belong to Mollicutes, known to have AT-rich genomes. As expected the abundance of A-rich and T-rich sequences at opposite corners and diagonal is immediately evident in the left image (Figure 3 a)). On the contrary similar feature is represented as horizontal dark patterns between A and T attractor points in the right image (Figure 3 b)). Unfortunately there is no general rule which CGR layout is the best, and it has to be chosen according to the analysed sequence.

3 NEW ALGORITHM FOR CGR IMPLEMENTATION

The main challenge for interactive visualisation was to optimize process of CGR images creation in a way that would be feasible to display and explore visualisation in real time. This section describes the process and optimizations used by our tool to transform sequences into the desired plot. Consider input sequence of length N:

$$\mathbf{s} = s_1, s_2, s_3, \ldots s_{N-2}, s_{N-1}, s_N \tag{2}$$

where s_i represents basis at position i in a genome. To produce visualisation for oligomers of size K a window has to be moved through the input sequence and occurrences of all observed oligomers have to be collected into an array of size 4^K. Counter values are transformed into colours and displayed in image in the layout described earlier.

The most time consuming part of algorithm is optimized by using logical operations such as AND, OR and SHIFT. These are more efficient than arithmetical

operations in Equation (1). The basic idea is to transform input sequence $\mathbf{s}$ into two binary sequences of length N, i.e. $\mathbf{x} = x_1, x_2 \dots x_N$ and $\mathbf{y} = y_1, y_2 \dots y_N$, where $x_i, y_i \in \{0, 1\}$. Values of x_i and y_i are determined as follows:

$$s_i = \begin{cases} A \rightarrow x_i = 0, y_i = 1 \\ C \rightarrow x_i = 1, y_i = 1 \\ G \rightarrow x_i = 0, y_i = 0 \\ T \rightarrow x_i = 1, y_i = 0 \end{cases} \tag{3}$$

Binary sequences $\mathbf{x}$ and $\mathbf{y}$ are used to retrieve counter index while processing input sequence. When examining oligomer in position i, index can be created by aligning following bits from arrays $\mathbf{x}$ and $\mathbf{y}$, that is:

$$index = y_i, y_{i-1} \dots y_{i-K}, x_i, x_{i-1} \dots x_{i-K}, \tag{4}$$

where y_i represents the most significant bit and x_{i-K} the least significant bit of $index$. Then corresponding counter is updated, and window is moved to the next input symbol. When all occurrences are counted, each value is transformed into colour and drawn in the final image of size $2^K \times 2^K$ pixels. Precise location corresponding to a specific counter can be easily extracted from the counter index by splitting it into two K-bit binary integers x and y representing horizontal and vertical positions, respectively:

$$index = y_K \dots y_2 y_1 x_K \dots x_2 x_1, \tag{5}$$

where x_K and y_K are the most significant bits and x_1 and y_1 are the least significant bits. Zooming in the final image can be implemented easily. To display smaller region from Figure 1, e.g. region 'CAG', only occurrences of oligomers that are preceded by the sequence 'CAG' have to be counted. This can be implemented directly by creating a zooming mask corresponding to the prefix sequence and testing bits at positions $i+1 \dots i+M$ in sequences $\mathbf{x}$ and $\mathbf{y}$ before incrementing the counter.

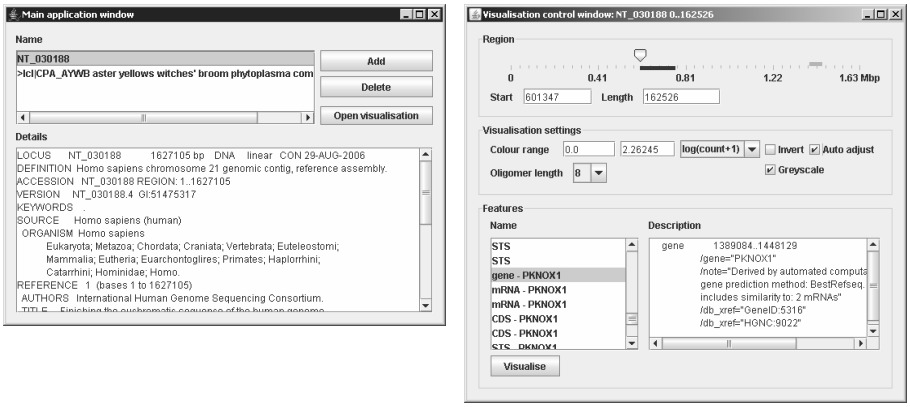
Similar idea of calculation was already proposed in [11]. In their case the whole sequence was transformed into a single binary sequence and bits corresponding to coordinates x and y in $index$ were mixed. In our approach, by splitting the input sequence into two separate sequences, only two shift registers are needed to retrieve index value. Thus, one step of our algorithm includes: updating values of registers with SHIFT operations, and composing $\mathbf{x}$ and $\mathbf{y}$ bits to desired form using bitwise OR operation. Then corresponding counter value is updated and next input symbol can be processed.

The most significant advantage of our approach, i.e. visualisation in real time, results from layout of counters in memory, which corresponds to layout of pixels in an image buffer (i.e. successive rows of pixels are concatenated together). This is due to the fact that least significant bits of $index$ encode horizontal position (column) and most significant bits encode vertical position (row). Identical layout of counters in memory and pixels in CGR visualisation allows us to simply transform 24-bit counter value to RGB colour value (8 bits per colour) and use the same part of

memory as the image buffer for drawing procedure. Therefore complicated pixel by pixel transformation of counters to final image according to CGR addressing scheme can be eliminated by our method of *index* calculation.

4 TOOL DESCRIPTION AND VISUALISATION PARAMETERS

The algorithm described in the previous section was used in our interactive visualisation tool. We concentrated on the main feature, i.e. visualisation of partial regions of the whole DNA sequence in real time. It was written in JAVA^{TM} language and is freely available at `http://www.fiit.stuba.sk/~makula/ifs/`. The application works with genomic data in commonly used FASTA and GENEBANK formats. Sometimes the GENEBANK format contains descriptive information about sequence such as the sequence description, source, authors, references and specification of other features included in the genome. These are extracted by program and used later in the visualisation process. In the *Main application window* all opened sequences with their details are shown (Figure 4 a)). Afterwards several visualisation windows can be opened and handled simultaneously. All available parameters of visualisation are shown in the *Visualisation control window* (Figure 4 b)), which is used to interactively adjust visualisation.

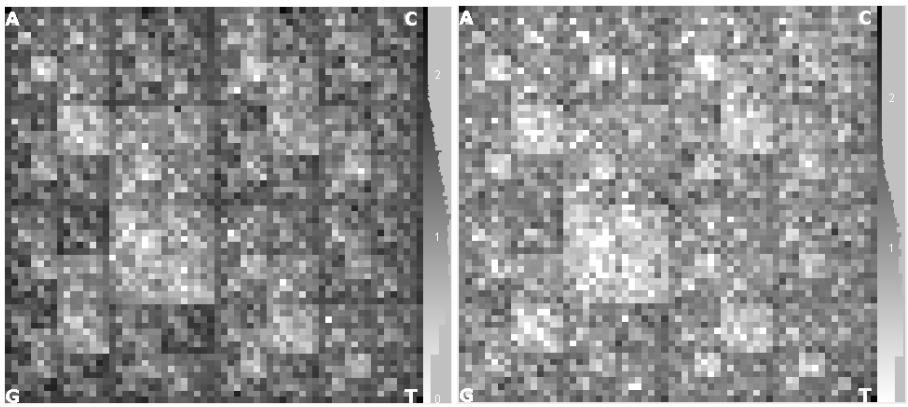


a) b)

Fig. 4. a) *Main application window* is used to open several sequences of FASTA or GENEBANK format. Description of the selected (highlighted) sequence (i.e. NT_030188) is shown in the *Details* text area. b) *Visualisation control window* controls all available parameters of visualisation, i.e. starting point, length of sequence, K, frequency range, colour and greyscale options and their inverse.

The main parameter of visualisation, i.e. region in the DNA sequence employed in the visualisation is controlled by the slider on the top of the Visualisation control window (Figure 4 b)). User can specify region directly by entering start position and length, or using slider control to move region swiftly through the sequence.

The image of oligomer frequencies changes continuously as the slider moves along the length of analysed sequence. In all cases, the vertical greyscale (or heatmap) histogram on the right-hand side is used to display the histogram of levels of grey (colours) and their corresponding $\log(count + 1)$ used in the image. Actual position and length of region is displayed in slider control as a red bar with corresponding size. As we have mentioned earlier, the GENEBANK format might include specification of genomic features such as genes, mRNA, etc. These are extracted from input file and listed in the *Features* part of *Visualisation control window* (Figure 4 b)). After a user selects feature its description is displayed in the right part of the control window. The *Visualise* button can be used to visualise selected feature (i.e. gene, RNA sequence, etc.) immediately. This allows us to easily analyse important parts of the sequence simply by a single click (Figure 5). In addition, each visualisation is always opened in a separate window, the title of which displays the sequence name and visualisation range. Thus, we can see immediately that these two genes are different and not the copies of the same gene (it is not uncommon in genetics that there are several copies of a gene in one genome). However, there are many similarities between these two fractals so it would be interesting to learn more about proteins these genes code for and about their evolutionary origin because they might be related. Thus, this comparison can spark new questions which otherwise would not be asked.

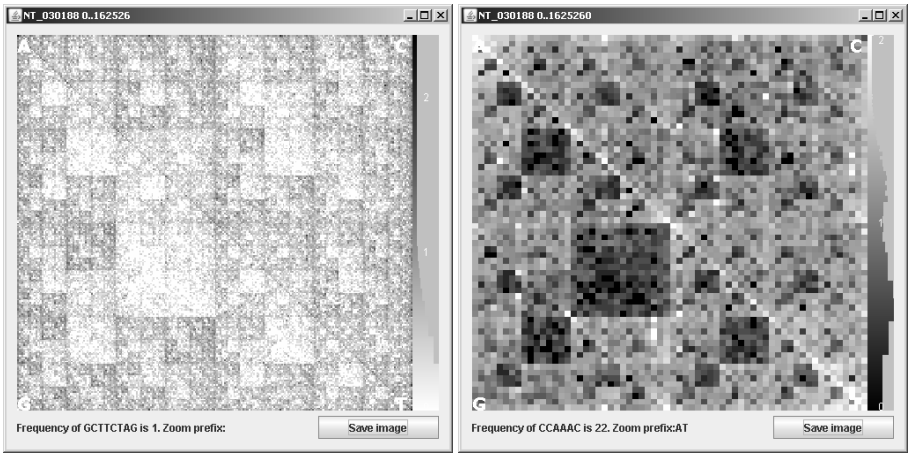


a) b)

Fig. 5. $K = 6$ oligomer frequency for a) gene denoted as ABCG1 and b) gene denoted as WDR4, both from human chromosome 21

Remaining parameters of visualisation are listed in the section *Visualisation settings* and illustrated in the greyscale images of oligomer frequencies of the whole sequence of human chromosome number 21 in Figure 6. Colour range, i.e. transformation of the oligomer frequency to colour heatmap or greyscale image can be adjusted. When automatic (default) adjustment of greyscale range is chosen, maxi-

mum value is always recalculated to display the most frequent oligomer in the black colour, and the least frequent ones in white. For this publication we use greyscale images, with the attractors order in corners from left to right: A, C, G, T, if not stated otherwise. As we can see in Figure 6a, there are very few frequent oligomers (black), but many which occur once or twice (white). An example is GCTTCTAG which occurs only once. Greyscale image (or coloured heatmap) can be inverted, which is useful when looking for infrequent oligomers (Figure 6 b)). Here, infrequent oligomers are black and frequent ones are white.



a) b)

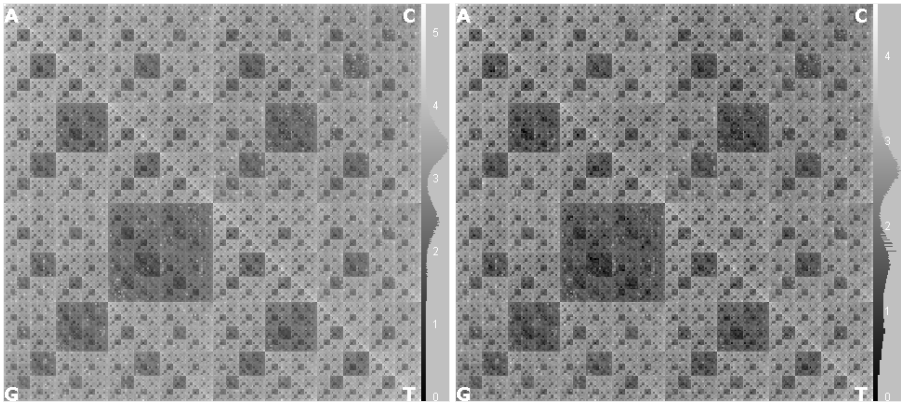
Fig. 6. Visualisation windows for different oligomer sizes: a) $K = 8$, b) $K = 6$. Greyscale in the left image a) is inverted to highlight infrequent oligomers in black colour in b).

Visualisation window can also be used to identify regions of sparse and dense occurrences of oligomers. As the mouse pointer moves over image area, oligomer corresponding to pointer position and its frequency is shown at the bottom part of window. The visualisation can be zoomed by mouse click which results in enlarging corresponding image quarter. For example visualisation only for oligomer frequencies occurring after nucleotide bases AT (Figure 6 b)) can be done by mouse clicks in A and T quarters of image in the Figure 6a, respectively. Thus, depending on the scale of interest number of cells in CGR visualisation can vary from 4×4 to $1\,024 \times 1\,024$.

5 MORE EXAMPLES

In this section we would like to provide more examples of the use of our genomic visualisation tool to show its usefulness. First let us visually compare human chromosomes X (≈150 Mbp) with the human chromosome Y (≈ 55 Mbp) (Figure 7). We use greyscale images, with the attractor order in corners from left to right: A,

C, G, T, oligomer length 6, black codes for less frequent oligomers and white for more frequent.



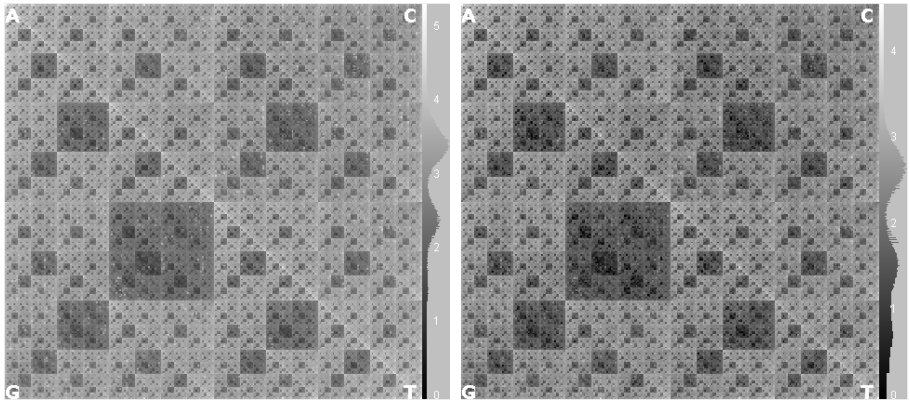
a) b)

Fig. 7. Visualisation of the $K = 8$ oligomers for a) human chromosome X, and b) human chromosome Y

We can see that fractals of these two human chromosomes are very similar. However, the histograms on the right hand side reveal differences in the oligomer distribution towards the shift to less frequent oligomers in the chromosome Y. This reflects the fact there are less bases in the Y chromosome than in the X chromosome, and also there is a slightly different distribution of oligomer frequencies visible in the histogram. The colour heatmaps would contain more information for the human eye, and the differences would be more visible. In the next Figure 8 we show the same chromosomes, but this time for the species *Pan troglodytes*, which is a common chimpanzee.

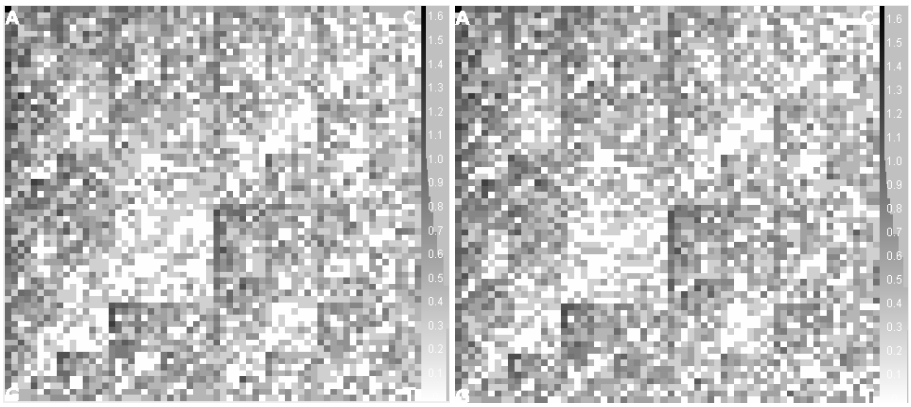
Based on Figures 7 and 8 we can see a striking similarity between the two sex chromosomes of our closest relative and us. However, there would be quite big differences in intensity (or oligomer counts) revealed in the colour images between these two species. Thus, fractal images obtained by our method show both the similarity and differences between genetic sequences. However, of course this comparison is only qualitative and not quantitative, although the program can be modified to generate such alignment-free statistics. This image comparison supports comprehensive comparison of the genomes of humans and chimpanzees, which showed that the DNA sequences of the two species are 96 % identical [18]. More subtle differences were revealed by detailed comparisons in the same study. Scientists have generated a largely complete catalogue of the genetic differences between humans and chimps constituting approximately thirty-five million single-nucleotide changes, five million insertion/deletion events, and various chromosomal rearrangements.

Another example is the comparison of the genome of the Influenza A viruses. A scientist has a genome from the virus found in chicken in Fujian in 2005 and



a) b)

Fig. 8. Visualisation of the $K = 8$ oligomers for a) the chimp chromosome X, and b) chimp chromosome Y



a) b)

Fig. 9. Visualisation of Influenza A (H9N2) virus from a) Shantou partridge in 2002, and b) Fujian chicken in 2005. $K = 6$.

wants to see immediately, whether it is the same virus as the one found in partridge in Shantou in 2002. Figure 9 shows graphical comparisons for these two viruses of the same length (10 kbp). Greyscale images cannot reveal subtle differences but the colour ones can. Then in colour images by pointing the cursor over pixels with different colour we can reveal such a small differences in occurrence of oligomers as 1. For instance the oligomer TAAAAG has frequency 12 in Fujian virus and 13 in the Shantou virus, etc.

To conclude, our tool is useful for fast and interactive graphical comparisons of genomes. This graphical visualisation can reveal both similarity and differences

between DNA sequences, the latter especially by means of colour coding of oligomer frequencies.

6 CONCLUSIONS

Growing amount of genomic and proteomic data is stretching bioinformaticians to develop efficient large-scale methods for pattern identification and knowledge discovery. Multiple alignments of many genomes are already used for interspecies comparisons [6, 10], but more compact data summarization methods are needed. Analyzing whole genomes to quickly reveal their salient features and to extract new knowledge is an essential goal for biological and computing sciences along the view of Peter Denning (the past President of ACM): "The old definition of computer science – the study of phenomena surrounding computers – is now obsolete. Computing is the study of natural and artificial information processes" [8, p.15].

We advocate the solution of compressing information about oligomer frequencies in long sequences into small, coloured fractal representations in 2D or 3D space. This can achieve compression of genome data by a million times. We offer a new tool written in Java that can be used in Windows by biologists and bioinformaticians all over the world since we have made it freely available.

By using our tool for a detailed analysis of any genome, one obtains fractal histograms with different oligomer lengths to identify specific repeated motifs in the genome. The oligomer lengths could be variable, depending on the scale of interest, up to oligomer size 10, which would map all unique single-copy sequences in a separate grid cell. The fractal spaces of the different region length can be viewed successively as a moving colour video track for quick visualisation of the relevant features, with several genomes shown side by side in synchrony. Successive sections of the genome can be analyzed separately, so that one can find out repeat-rich regions, coding and non-coding regions and so on in the genome. Keeping track of the oligomer coordinates as well would enable one to map specific oligomer groups to specific locations in the genome as well. Such a tool is thus a very versatile method of visual exploration and comparison of genomes. For fractal coding of genome oligomer distribution, an example of phytoplasma genome showed that specific types of repeats can be visualized effectively.

Of course, there is a room for improvement [12]. Overlaying/substracting from a plot of similar length random sequence with same ratios of A/T/C/G could show statistically significant differences according to a specific cutoff. Similarly, comparing two or more genomes by overlaying could be easily accomplished, to pinpoint the relevant changes in abundant or under-represented oligomers in the genome. This would be effective for immediate and informative genome scale visual comparisons. Finally, automation of the method could be accomplished by image processing of the overlayed/subtracted images to highlight/extract the oligomer clusters of interest in the fractal space, down to the specific most common oligomers differing in frequency between the genomes. Various extensions of the fractal method seem

worth pursuing for novel types of DNA sequence pattern clustering and classification approaches. Finally, moving the bioinformatics domain symbolic data into bitmap representation domain makes it possible to use the wide variety of bitmap image analysis methods developed in other fields outside biology. This interdisciplinary approach should be both interesting and fruitful for informative visualization, data mining and knowledge discovery in bioinformatics and chemoinformatics datasets.

Acknowledgments

This work was supported by the APVV-20-030204 and VG-1/0848/08 grants.

REFERENCES

[1] Almeida, J. S.—Carrico, J. A.—Maretzek, A.—Noble, P. A.—Fletcher, M.: Analysis of Genomic Sequences by Chaos Game Representation. Bioinformatics, Vol. 17, 2001, No. 5, pp. 429–437.

[2] Baltimore, D.: The Invisible Future. Wiley, New York, 2001.

[3] Barnsley, M. F.: Fractals Everywhere. Academic, New York, 1988.

[4] Barrett, T.—Xie, T.—Piao, Y.—Dillon-Carter, O.—Kargul, G. J.—Lim, M. K.—Chrest, F. J.—Wersto, R.—Rowley, D. L.—Juhaszova, M.—Zhou, L.—Vawter, M. P.—Becker, K. G.—Cheadle, C.—Wood, H. W.—McCann, U. D.—Freed, W. J.—Ko, M. S.—Ricaurte, G. A.—Donovan, D. M.: A Murine Dopamine Neuron-Specific cDNA Library and Microarray: Increased COX1 Expression During Methamphetamine Neurotoxicity. Neurobiology of disease, Vol. 8, 2001, No. 5, pp. 822–833.

[5] Basu, S.—Pan, A.—Dutta, C.—Das, J.: Chaos Game Representation of Proteins. Journal of Molecular Graphics and Modelling, Vol. 15, 1997, No. 5, pp. 279–289.

[6] Brudno, M.—Poliakov, A.—Salamov, A.—Cooper, G. M.—Sidow, A.—Rubin, E. M.—Solovyev, V.—Batzoglou, S.—Dubchak, I.: Automated Whole-Genome Multiple Alignment of Rat, Mouse, and Human. Genome Research, Vol. 14, 2004, No. 4, pp. 685–692.

[7] Crick, F.: Central Dogma of Molecular Biology. Nature, Vol. 227, 1970, pp. 561–563.

[8] Denning, P. J.: Computing As Natural Science. Communications of the ACM, Vol. 50, 2007, No. 7, pp. 13–18.

[9] Fiser, A.—Tusnady, G. E.—Simon, I.: Chaos Game Representation of Protein Structures. Journal of Molecular Graphics, Vol. 12, 1994, No. 4, pp. 302–304.

[10] Frazer, K. A.—Pachter, L.—Poliakov, A.—Rubin, E. M.—Dubchak, I.: VISTA: Computational Tools for Comparative Genomics. Nucleic Acids Resources, Vol. 32 (Web Server issue), 2004, pp. W273–W279.

[11] Hao, B.—Lee, H. C.—Zhang, S.: Fractals Related to Long DNA Sequences and Complete Genomes. Chaos, Solitons and Fractals, Vol. 11, 2000, No. 6, pp. 825–836.

[12] Havukkala, I.—Beňušková, Ľ.—Pang, S.—Jain, V.—Kroon, R.—Kasabov, N.: Image and Fractal Information Processing for Large-Scale Chemoin-

formatics, Genomics Analyses and Pattern Discovery. In: J. C. Rajapakse, L. Wong and R. Acharya (Eds.): Pattern Recognition in Bioinformatics, PRIB 2006. Lecture Notes in Bioinformatics, Vol. 4146, 2006, pp. 163–173.

[13] Jeffrey, H. J.: Chaos Game Representation of Gene Structure. Nucleic Acids Resources, Vol. 18, 1990, No. 8, pp. 2163–2170.

[14] Joseph, J.—Sasikumar, R.: Chaos Game Representation for Comparison of Whole Genomes. BMC Bioinformatics (Availaible on: http://www.biomedcentral.com/1471-2105/7/243), Vol. 7, 2006, No. 1, pp. 243.

[15] Liolios, K.—Tavernarakis, N.—Hugenholtz, P.—Kyrpides, N. C.: The Genomes On Line Database (GOLD) v.2: A Monitor of Genome Projects Worldwide. Nucleic Acid Resources, Vol. 34, 2006, pp. D332–D334.

[16] Shen, J.—Zhang, S.—Lee, H. C.—Hao, B.: SeeDNA: Visualisation of k-String Content of Long DNA Sequences and Their Randomized Counterparts. Genomics, Proteomics and Bioinformatics, Vol. 2, 2004, No. 3, pp. 192–196.

[17] Tiňo, P.: Spatial Representation of Symbolic Sequences Through Iterative Function Systems. IEEE Transactions on Systems, Man, and Cybernetics Part A: Systems and Humans, Vol. 29, 1999, No. 4, pp. 386–392.

[18] The Chimpanzee Sequencing and Analysis Consortium: Initial Sequence of the Chimpanzee Genome and Comparison With the Human Genome. Nature 437, 2005, pp. 69–87.

Matej Makula received the B. Sc. degree in 1999 and the M. Sc. degree in 2001 in computer science from the Slovak University of Technology, Bratislava, Slovakia. In 2009, he successfully defended his Ph. D. thesis in applied informatics at the Faculty of Informatics and Information Technologies at the Slovak University of Technology. His research interests include neural networks and symbolic time-series processing.

Ľubica Beňušková received the RNDr. degree in physical electronics in 1982 and the Ph. D. degree in biophysics in 1994, from Comenius University, Bratislava, Slovakia. She also received the M. A. degree in psychology from Vanderbilt University, Nashville, TN, in 1993. Currently, she is a Senior Lecturer at the Department of Computer Science, University of Otago, Dunedin, New Zealand. She is also an Associate Professor at the Department of Applied Informatics at the Faculty of Mathematics, Physics and Informatics, Comenius University. Her research interests include computational modelling of synaptic plasticity, computational neurogenetics, and processing time series in neural networks.