



FORRepeats: detects repeats on entire chromosomes and between genomes

A. Lefebvre*, T. Lecroq, H. Dauchel and J. Alexandre

ABISS - Université de Rouen, 76821 Mont Saint-Aignan Cedex, France

Received on February 1, 2002; revised on May 31, 2002; accepted on August 12, 2002

ABSTRACT

Motivation: As more and more whole genomes are available, there is a need for new methods to compare large sequences and transfer biological knowledge from annotated genomes to related new ones. BLAST is not suitable to compare multimegabase DNA sequences. MegaBLAST is designed to compare closely related large sequences. Some tools to detect repeats in large sequences have already been developed such as MUMmer or REPuter. They also have time or space restrictions. Moreover, in terms of applications, REPuter only computes repeats and MUMmer works better with related genomes.

Results: We present a heuristic method, named FORRepeats, which is based on a novel data structure called factor oracle. In the first step it detects exact repeats in large sequences. Then, in the second step, it computes approximate repeats and performs pairwise comparison. We compared its computational characteristics with BLAST and REPuter. Results demonstrate that it is fast and space economical. We show FORRepeats ability to perform intra-genomic comparison and to detect repeated DNA sequences in the complete genome of the model plant *Arabidopsis thaliana*.

Availability: see contact

Contact: arnaud.lefebvre@univ-rouen.fr

1 INTRODUCTION

Recently, the strategy of pairwise comparisons of entire chromosomes or entire genomes has proved useful in the annotation of newly sequenced genomes (Ouzounis *et al.*, 1996; Karlin *et al.*, 1998; Louis *et al.*, 2001). The interest of this inter-genomic comparative approach is to rapidly characterize similar and syntenic regions or, in contrast, to provide evidence of large-scale changes such as tandem repeats, large insertions, deletions or reversals. The availability of more, fully sequenced, genomes will increase the value of this approach, notably in the case of phylogenetically related genomes such as two closely related species or two different strains or cultivars (Wiehe *et al.*, 2000; Bennetzen, 2000; Boucher *et al.*, 2001).

Genomes are dynamic and redundant structures; genomes are regularly subject to mutations, duplications, inversions and suppressions during the life of an organism or over generations. Repetitive DNA sequences are one of the principal origins of genomic instability because recombination between similar sequences can cause chromosomal rearrangements. These repetitive sequences, from one to ten thousand nucleotides long, are present many hundreds or thousands of times in eucaryotic genomes, located at a few chromosomal sites either in tandem (minisatellites, microsatellites, rDNA repeats) or widely dispersed (mobile DNA) (Schmidt and Heslop-Harrison, 1998). Hence the intra-genomic analysis of the nature, the location and the dynamics of repeated sequences is an important theme in comparative genomics.

Finally, the search for inter- and intra-genomic similarities leads to an understanding of the global architecture of genomes. This understanding is needed for both adequate annotation and for the study of the evolution of organisms. These aspects of comparative genomics are a challenge for bioinformaticians who must develop new methods and tools to process multimegabase DNA sequences simultaneously and quickly (Miller, 2001).

Currently, to compare long sequences, two approaches can be considered (parallel approaches are not taken into account). The first one, used in BLAST (Altschul *et al.*, 1997) and MegaBLAST (Zhang *et al.*, 2000), consists in preprocessing the query sequence and then searching for similarities in a databank. The second approach, used in MUMmer (Delcher *et al.*, 1999) and REPuter (Kurtz *et al.*, 2000) consists in finding exact repeats and then extending those repeats allowing errors. In the case of BLAST, the size of the query sequence is a limiting factor (Figure 4). Since MegaBLAST computes the differences between two sequences, the underlying algorithm requires very similar genomic sequences. In the case of MUMmer or REPuter, the size of the suffix tree (McCreight, 1976) is the limiting factor: the memory space used by MUMmer is approximately 35 times the size of the query sequence; the memory space used by REPuter is approximately 12 times the size of the sequence but is optimized for a restricted

*To whom correspondence should be addressed.

four letter alphabet.

In this article, we propose a new heuristic based on a data structure called factor oracle (Allauzen *et al.*, 1999) which is space and time economical, alphabet independent, and able to find repeats within chromosomes or to compare chromosomes rapidly.

Here we describe the algorithm and the first implementation of our method named FORRepeats (Factor ORacle Repeats). Then, we give some evidence of the accuracy and space and time efficiency on long sequences of our method compared with BLAST and REPuter. Finally, we test FORRepeats ability to perform intra-genomic comparison and to detect repeated DNA sequences in the complete genome of the model plant *A. thaliana*. In this aim, we compare our results with those obtained with BLAST and MUMmer (The Arabidopsis Genome Initiative, 2000).

2 METHOD

The method we describe in this article is decomposed in two steps: the first step is the search of exact repeats in a sequence and the second step consists in an extension, allowing errors, of these exact repeats.

This general scheme is classically used in methods looking for similarities between sequences or looking for approximate repeats.

2.1 Computing exact repeats with a factor oracle

The method described in (Lefebvre and Lecroq, 2000), based on a factor oracle (Allauzen *et al.*, 1999) gives, in linear time and space, the length and an occurrence of a repeated suffix of each prefix of a word p (see Figure 1). In this section, after a few basic definitions, we briefly present this method.

Definitions Let $p = p[1..m]$ be a word of length $|p| = m$ over an alphabet Σ . Let ε be the empty word ($|\varepsilon| = 0$). A word $w \in \Sigma^*$ is a *factor* of p if and only if p can be written $p = uwv$ with $u, v \in \Sigma^*$. A word $u \in \Sigma^*$ (resp. $v \in \Sigma^*$) is a *prefix* (resp. *suffix*) of p if and only if p can be written $p = uv$ with $u, v \in \Sigma^*$. An *occurrence* of a factor w of p is denoted by the position $i \in [1..m]$ of its ending letter. A *repeated factor* of a word p is a factor of p which has at least two distinct occurrences in p .

The factor oracle The factor oracle of a word p of length m , denoted by $Oracle(p)$, is a deterministic finite automaton (Q, q_0, F, δ) where $Q = \{0, 1, \dots, m\}$ is the set of states, $q_0 = 0$ is the starting state, $F = Q$ is the set of terminal states and δ is the transition function. The factor oracle of a word p of length m has the following properties: it has exactly $m + 1$ states; it has within m and $2m - 1$ transitions; it recognizes at least all the factors of p and slightly more words. The exact characterization of the

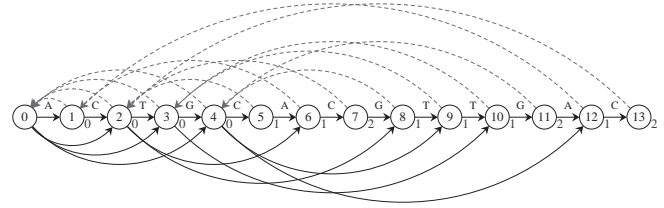


Fig. 1. Factor oracle of the word $p = ACTGCACGTTGAC$. The black plain arrows represent the transitions (labels of external transitions are omitted because every transition leading to a state i is already labeled with the letter $p[i]$). The red dashed arrows represent the suffix links. The values written near the states are the lengths of the repeated suffixes. All the factors of p are recognized from state 0. The word $C TT$ is recognized (with the path $(0, 2, 3, 10)$) though it is not a factor of p . This factor oracle can be represented by p itself, and the list of couples $(0, 2)$, $(0, 3)$, $(0, 4)$, $(2, 6)$, $(2, 8)$, $(3, 10)$, $(4, 9)$ and $(4, 12)$.

set of words recognized by the factor oracle is still under study.

There is a bijection between the states of the oracle and the $m + 1$ prefixes of p (including the empty one). Each transition leading to state i is labeled by $p[i]$. We distinguish two kinds of transitions: transitions from state i to state $i + 1$ are called *internal transitions* and transitions from state i to state j such that $j - i > 1$ are called *external transitions*, with $0 \leq i < j \leq m$. There are exactly m internal transitions. Thus, to store the oracle, one needs to store only the word p and at most $m - 1$ external transitions without their label. All the other informations can be deduced from the word p . This representation is economical: the memory space used is approximately equal to 10.5 times the size of the sequence. This structure is linear in space, and its construction is linear in time (Allauzen *et al.*, 1999). Figure 1 shows the oracle of the word $ACTGCACGTTGAC$.

Computing the lengths of repeated factors The factor oracle of a word p is used to compute the length of long repeated suffixes for each prefix of p . Let us introduce some definitions.

DEFINITION 1. We denote by $LRS(i)$ the longest repeated suffix of $p[1..i]$: $LRS(i) = \max\{v \mid v \text{ is a suffix of } p[1..i] \text{ and } v \text{ is a factor of } p[1..i-1]\}$.

DEFINITION 2. Allauzen et al. (1999) $S[i]$, the suffix link of a state i of $Oracle(p)$, is equal to the state in which the longest repeated suffix of $p[1..i]$ is recognized: $S[i] = \delta(0, LRS(p[1..i]))$.

The state $S[i]$ is equal to an occurrence of a repeated suffix of $p[1..i]$.

Now we define the array lrs where $lrs[i]$, $0 \leq i \leq m$, is a good approximation of $|LRS(i)|$ (see section 3). The reader can refer to (Lefebvre and Lecroq, 2000) for the details of the computation of the lrs values. For each state i of $Oracle(p[1..|p|])$, $lrs[i]$ is equal to the length of a repeated suffix of $p[1..i]$ such that one of its occurrences is equal to $S[i]$. For example, in Figure 1, an exact repeat of length 2 is found in positions 4 and 11 since $S[11] = 4$ and $lrs[11] = 2$. This exact repeat is ‘TG’.

We are able to locate efficiently those repeats, given by tables lrs and S , in large sequences. It should be noticed that the values computed with this method can never be greater than the lengths of the longest repeats. This means that the method can provide partial repeats but never provides false positives.

2.2 Computing approximate repeats

To find approximate repeats, we extend each exact repeat in a same manner as the hit extension step of BLAST, to the left and to the right allowing errors: each exact repeat, found with the same data structure as in the previous method and longer than a value given by the user, is extended to the left and to the right. This extension phase is realized until the similarity percentage between the two extended factors drops below a value fixed by the user.

Different ways to compute the similarity can be used such as Hamming (without gaps) or Levenshtein (with gaps) distances. In the case of Hamming distance and using reasonable parameters (percentage > 85%, minimal length equal to 15 bp), running times are never more than doubled compared to the exact case.

2.3 Implementation of the oracle

This method has been implemented in the C++ language, using Hamming distance for the hit extension step. Concerning the memory space, the factor oracle has to store the sequence (one byte per character), the external transitions (four bytes per character and four bytes per transition) and the suffix links (four bytes each). In practice this leads to an average memory space of 10.5 bytes per character. The output of FORRepeats consists in a list of triplets $(pos_1, pos_2, length)$ where pos_1 and pos_2 are the ending positions of the repeated factor, and $length$ is its length.

Examples of running times and spaces are given in Section 4. All the results presented in this paper have been obtained on a computer running Linux, with 1Go RAM and a 500 MHz processor.

2.4 Display of the repeats computed by FORRepeats

FORRepeats computes repeats irrespective of their lengths and kinds (tandem, dispersed...).

A visualization interface enables us to display repeats

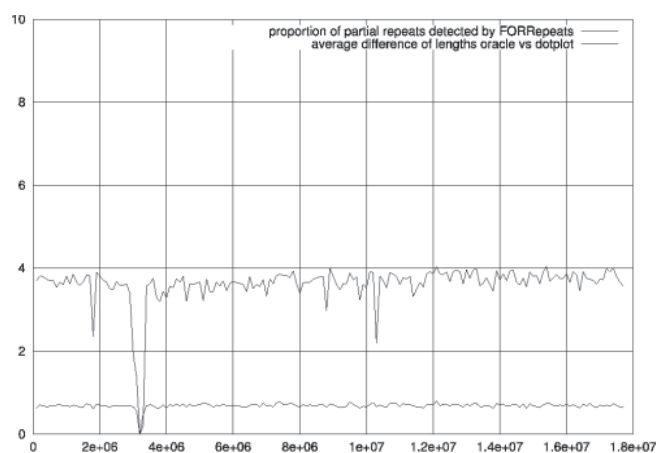


Fig. 2. Comparison of the lengths of the exact repeats found with FORRepeats and the dotplot method, irrespective of their lengths, performed on the chromosome IV of *A. thaliana*. Each point of the curves corresponds to a window of size 100 000. The Y-axis represents two parameters: the proportion of positions (normalized to ten) where the two computed lengths are different (upper curve) and the average difference between these lengths (lower curve).

according to their lengths, positions and distances between occurrences.

A web interface, currently under development, enables us to access the repeated segments by clicking on the corresponding links (see Figures 5, 6 and 7).

3 ACCURACY OF THE METHOD

Since this method is a heuristic, we performed some experiments in order to evaluate the number of partial detected repeats. In this aim, we estimate the number of states i such that $lrs[i]$, the length of the detected repeat, is different from $|LRS(i)|$, the length of the longest repeat. The comparison has been made with a dotplot on different prokaryotic and eukaryotic genomes.

For all the results concerning *A. thaliana* presented in this article, we used the genome sequence assembly version v040701 available at the MIPS ftp site <ftp://ftpmips.gsf.de/cress>.

Figures 2 and 3 show experimental results on the chromosome IV of the model plant *A. thaliana*. As far as the dotplot method is slow, we performed this comparison on a sliding window. In order to check if the window size does not introduce a bias on the accuracy, different window sizes have been tested (25 000, 50 000, 100 000 and 200 000). Whatever the tested window sizes, the accuracy is equivalent, thus, only results concerning windows of size 100 000 are presented.

Figure 2 deals with all the repeats irrespective of their lengths. It reveals a proportion of positions i where

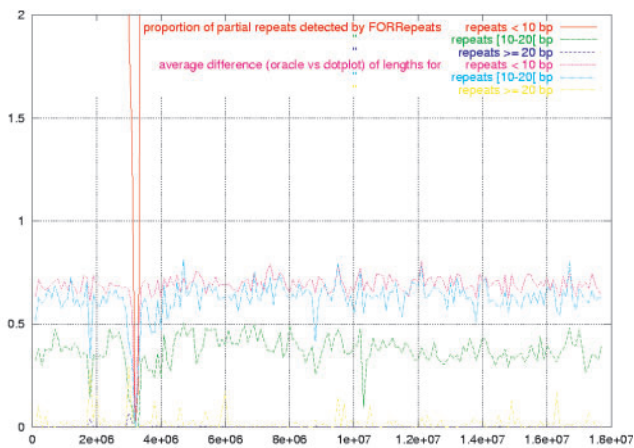


Fig. 3. Comparison of the lengths of the exact repeats found with FORRepeats and the dotplot method, respective of their lengths, performed on the chromosome IV of *A. thaliana*. Each point of the curves corresponds to a window of size 100 000. The Y-axis represents two parameters for three classes of lengths: the proportion of positions (normalized to ten) where the two computed lengths are different and the average difference between these lengths.

$lrs[i] < |LRS(i)|$ of about 40%. This means that, in 60% of the cases, FORRepeats detects the longest repeats. The average difference between lrs and dotplot values is less than 1bp.

We estimated the contribution of different size repeats (shorter than 10 bp, between 10 and 20 bp, and longer than 20 bp) in the proportion of the partially detected repeats. Figure 3 reveals that almost all the positions where a difference is observed concern short repeats (shorter than 10 bp). In the case of the longest repeats, between 10 and 20 bp, and longer than 20 bp, the proportion drops to 6% and 1% respectively and the average difference becomes negligible. This experiment show that the longest the repeats the better the accuracy of FORRepeats is.

4 TIME AND SPACE EFFICIENCY

In this section we compare FORRepeats, in terms of space and time efficiency, to two well-known methods. First, we compare FORRepeats to BLAST which is the most widely used heuristic in bioinformatics. As we have no assumption on the similarity of the sequences to be studied, we used BLASTn version 2.1.3 (Altschul *et al.*, 1997) rather than MegaBLAST (Zhang *et al.*, 2000). Second, as far as our method is based on the search of exact repeats in a first step, we compare FORRepeats to REPuter (Kurtz *et al.*, 2000), a method based on the search of exact repeats using a suffix tree (McCreight, 1976).

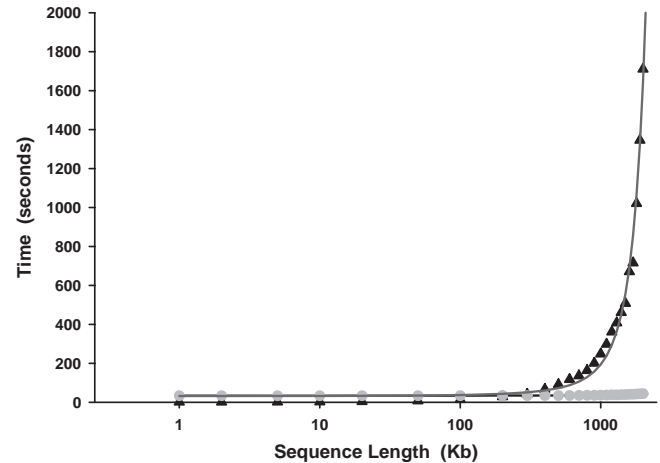


Fig. 4. Processing time versus sequence length for BLAST (black triangles) and the factor oracle (red circles): curves are fitted with an exponential regression for BLAST, and with a linear regression for FORRepeats. The database used only contains chromosome IV of *A. thaliana*. Query sequences are longer and longer prefixes of chromosome IV of *A. thaliana*.

4.1 FORRepeats versus BLAST

We ran BLASTn and FORRepeats (approximate repeats version) on longer and longer prefixes (1 kb to 2000 kb, from 5' end) of chromosome IV of *A. thaliana* against the entire chromosome. Concerning BLASTn, we used the default parameters whilst FORRepeats parameters were 15 for the size of the exact repeats to be extended and 80% of identity.

Processing times are shown in Figure 4: curves are fitted with an exponential regression for BLAST, and with a linear regression for FORRepeats.

According to these regressions the processing time on chromosome I of *A. thaliana*, which is the longer one, with BLAST would be 8.24×10^{29} seconds which is approximately 3×10^{20} centuries, while it takes two minutes to process it with FORRepeats.

Experiments show that for sequences longer than 200 kb, FORRepeats becomes a more suitable method than BLAST, in terms of speed efficiency. The fact that the length of the query sequence is a limiting factor of BLAST explains the usual strategy of splitting long query sequences into short fragments The Arabidopsis Genome Initiative (2000); Mayer *et al.* (1999).

4.2 FORRepeats vs REPuter

We performed tests on few chromosomes of increasing sizes from different linear and circular genomes. In Table 1, given times correspond to the construction times of the two data structures. These results show that FORRepeats is twice as fast as REPuter on all these

Table 1. Comparison of the factor oracle and REPuter running times on nine chromosomes (taken from GenBank except *A. thaliana* taken from the MIPS ftp site). Running times (**time**) are given in seconds, **bpc** indicates the number of bytes per sequence character used by the factor oracle

| Genomes | size | Factor oracle bpc | time | REPuter |
|---------------------------|------|----------------------|------|---------|
| <i>S. cerevisiae</i> III | 0.31 | 10.70 | 0.37 | 0.71 |
| <i>S. cerevisiae</i> VIII | 1.1 | 10.6 | 0.7 | 1.37 |
| <i>S. cerevisiae</i> IV | 1.5 | 10.52 | 2.2 | 4.16 |
| <i>N. meningitidis</i> | 2.1 | 10.16 | 2.95 | 6.0 |
| <i>B. subtilis</i> | 4.2 | 10.34 | 6.53 | 12.3 |
| <i>E. coli</i> | 4.6 | 10.3 | 7.18 | 13.6 |
| <i>A. thaliana</i> IV | 17.5 | 10.16 | 29.0 | 55 |
| <i>A. thaliana</i> II | 19.6 | 10.16 | 32.7 | 63 |
| <i>H. sapiens</i> XXI | 33.8 | 10.0 | 55.6 | 109 |

examples, keeping in mind that REPuter is an exhaustive method while FORRepeats is a heuristic (see section 3). Due to an optimization on an alphabet restricted to the four letters A, T, C and G, the memory space used by REPuter is equal to 12.5 times the size of the sequence; without any optimization, the space occupancy of FORRepeats is approximately equal to 10.5 times the size of the sequence. Moreover, FORRepeats is alphabet independent and can thus deal with the degenerated nucleotide codes and also with amino acids.

5 SOME APPLICATIONS

In order to test FORRepeats ability to perform intra-genomic comparisons and to detect DNA repeated sequences, we use the complete genome of the model plant *A. thaliana*. It comprises 125 megabases segmented into five chromosomes and low repetitive DNA content ($\approx 25\%$). A first complete analysis of *A. thaliana* genome, describing structural and functional annotation of predicted genes, mobile elements and other repeats distributions, was realized by the Arabidopsis Genome Initiative (The Arabidopsis Genome Initiative, 2000). A complete view of chromosomal organization, including distribution of tandemly repeated genes, transposable elements, and centromeric, telomeric repeats was assessed by using the program BLAST, in combination with the program MUMmer in the case of duplication evidences.

This previous work, permitted us to have a reference to test FORRepeats on this genome. Our aim, here, is not to re-analyze exhaustively the genome organization of *A. thaliana*, but to estimate the efficiency of FORRepeats method to permit a first filtration of multimegabase sequences and access rapidly the identification and distribution of different types of repetitive DNA in a genome. In this section, we focus on the examples of the

distribution of repeated sequences in chromosome IV, the comparison of chromosomes II and IV, and the search of duplicated segments.

5.1 Intra-chromosomal comparison

We use FORRepeats to access distribution of repetitive sequences in the chromosome IV of *A. thaliana*, comprising a total sequence of around 18 Mb. On Figure 5, each line represents a repeat, its ends give the positions of this repeat. The top graph (Mayer *et al.*, 1999) is obtained after approximately 10,000 runs of BLASTn and shows approximate repeats (length > 150 bp and 85% of similarity). The middle and lower ones are obtained with the factor oracle. The middle one represent exact repeats. All the exact repeats are computed in 30 seconds, only repeats of length greater than 75 bp are displayed. The bottom one represents approximate repeats. All the exact repeats are computed, then only those of length greater than 15 bp are extended. Only repeats of length greater than 150 bp and 85% of similarity are displayed. Those parameters have been chosen in order to meet BLAST parameters. The extension phase increases the overall execution time from 30 seconds to 1 minute. As expected, density and lengths of repeats are increased in the case of approximate repeats.

Most genomes, have a high content of repetitive DNA, both tandem and dispersed. Tandemly repeated satellite DNA sequences are generally found at the centromeres and telomeres. As we can see Figure 5, the two methods lead to graphs with similar shapes, giving evidence of these two types of repetitions. This result is consistent with the experiments presented in the Section 3, since repeats of length 15 are extended and since this kind of repeat is detected by FORRepeats with 6% of differences compared to a dotplot. Chromosome IV is acrocentric, with the shorter arm exhibiting much less repetitive sequences than the longer arm on the two graphs. On the shorter arm of the chromosome, the two graphs show the canonical telomeric repeats, which comprise the nucleolar organizer (NOR 4), a cluster of tandemly repeated blocks of uninterrupted ribosomal DNA genes (18S, 5.8S and 25S subunits) (Copenhaver and Pikaard, 1996; Mayer *et al.*, 1999). At the centromere, the factor oracle enables to find local repeats that are typical of the pericentromeric heterochromatin (Round *et al.*, 1997). For instance we can see those located around the knob which is a satellite region highly repeated in tandem of 180 bp (also named pAL1). This region has been highlighted using *in situ* hybridization (Martinez *et al.*, 1986; Heslop-Harrison *et al.*, 1999) on all five chromosome pairs of *A. thaliana* and constitutes between 2 and 5% of the genome. Around both sides of the centromere, very dense dispersed repeats, are observed on the three graphs. They can correspond to retro-elements of Athila type (Pelissier

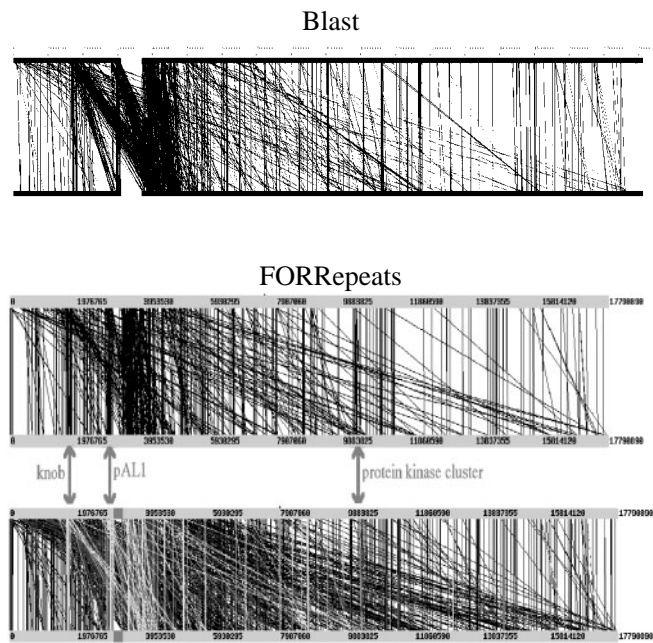


Fig. 5. The three graphs show repeats in chromosome IV of *A. thaliana*. Horizontal bars represent this chromosome. Each vertical or oblique line represents a repeat, its ends give the positions of the repeat. The top graph (Mayer *et al.*, 1999), obtained after approximately 10 000 runs of BLASTn, shows approximate repeats while the middle and bottom ones, obtained with the factor oracle, show exact and approximate repeats respectively. Only exact repeats of length greater than 75 bp and approximate repeats of length greater than 150 bp (85% of similarity) are displayed.

et al., 1995) which are the most abundant dispersed repetitive elements characterized in *A. thaliana*, long of 10 500 bp, representing at least 0.3% of the genome.

Finally, the factor oracle detects, like BLAST, intra-chromosomal local repeats on the same DNA strand corresponding to multigenic families. For example, a fragment of chromosome IV, localized around 10Mb from the telomere presents a repeat of length 3259 corresponding to a gene cluster of protein kinases multigenic family. The region is identically and consecutively repeated nine times. The comparison of this region with the coding or predicted coding region on the chromosome IV cartography shows a good correlation with the result obtained with the factor oracle.

5.2 Inter-chromosomal comparison

In order to compute inter-chromosomal comparisons with a factor oracle, we concatenate the two sequences to be compared, separated by a 'joker' (symbol absent from both sequences). The two sequences are then considered by FORRepeats as a single one. Only repeats which have occurrences on both sides of the 'joker' are considered.

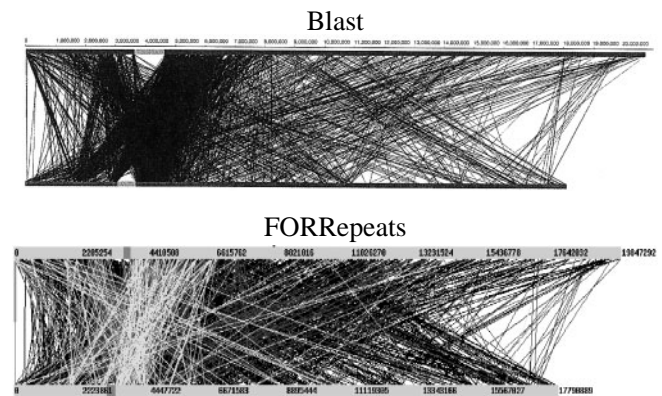


Fig. 6. The two graphs show approximate repeats found in the comparison of chromosome II (upper bar) with chromosome IV (lower bar) of *A. thaliana*. Computation and visualization parameters are identical to the ones of Figure 5.

Figure 6 shows the results of the comparison of chromosomes II and IV computed by the factor oracle. Computation and visualization parameters are identical to the ones of Figure 5. As in the intra-chromosomal case, we can observe a good correlation with the results obtained with a series of BLASTn (Mayer *et al.*, 1999).

5.3 Duplicated segments detection

Figure 7 presents an illustration of FORRepeats ability to detect duplicated segments. This figure shows two large duplicated segments present in chromosomes III and IV of *A. thaliana*. Results obtained with FORRepeats are compared to those obtained with MUMmer and BLAST (from the MIPS web site <http://mips.gsf.de>). Concerning FORRepeats, two filtration approaches are proposed: the first one consists in the choice of the rate of similarity (80% for graphs (a) and (b), 50% for (c) and (d)); the second choice is to display only repeats longer than a fixed threshold (50 bp for (a), 100 bp for (c), and, 200 bp for (b) and (d)).

In the four cases, the duplicated segments, highlighted by the MIPS, are detected by FORRepeats. The different filter parameters allow to isolate and study different kinds of repeats contained in these regions: for instance, parameters used to obtain graph (b) allow to isolate good candidates for clusters of coding regions.

6 DISCUSSION

FORRepeats is a new and efficient tool for the detection of repeats and for the comparison of large genomic sequences. It can detect exact repeats and enables the detection of approximate ones within a single sequence as well as between two sequences. It is based on a factor oracle, which is a fast and space economical data structure.

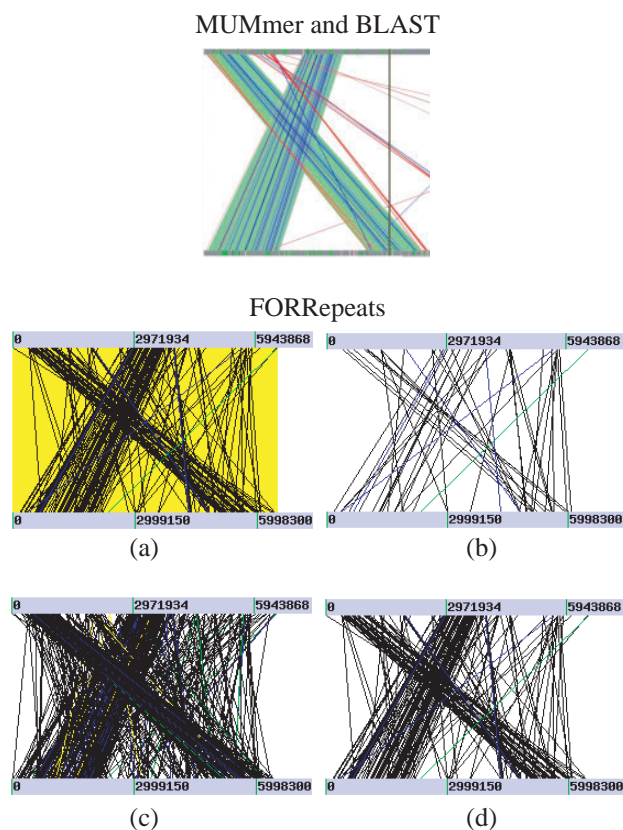


Fig. 7. Two large duplicated segments present in chromosomes III (upper bar) and V (lower bar) of *A. thaliana*. The top graph (from the MIPS web site <http://mips.gsf.de>) was obtained using MUMmer and then tBLASTx: the repeats shown (thin lines) are in coding regions, longer than 1000 bp with at least 50% of similarity. In the top figure, large green bands highlight the duplicated segments. The other four graphs were obtained with FORRepeats and represent different analyses of the same region as analysed in the top graph. Computing parameters used to obtain graph (a) and (b) are 80% similarity. Only repeats longer than 50 bp are displayed in graph (a) while graph (b) displays repeats longer than 200 bp. Yellow lines in (a) represent repeats shorter than 100 bp whilst black lines represent repeats longer than 100 bp but shorter than 500 bp. Computing parameters used to obtain graphs (c) and (d) are 50% similarity. Only repeats longer than 100 bp are represented in graph (c) whilst graph (d) displays repeats longer than 200 bp.

The construction of this data structure is simpler than the construction of a suffix tree. FORRepeats is a heuristic, it can detect partial repeats but in any case would not generate false positives.

It should be noted that the factor oracle is constructed such that analysis of a sequence in the 5′–3′ direction would give repeats that would be slightly different from those given by analysis in the 3′–5′ direction (not shown).

In terms of applications, FORRepeats constitutes an

original and efficient tool for detecting interesting regions in genomic sequences. It offers a convenient method for the study of genome repetitions such as local or distant repeats. Thus it can be used for intra- and inter-species comparative genomics. The intra-genome approach is particularly appropriate for the study of the organization of an individual genome, whilst the inter-species approach is useful for the annotation of newly sequenced genomes or for the study of phylogenetic relationships: for instance between *A. thaliana* and *Oriza sativa*, between human and mouse, or between related strains of bacteria or viruses.

Two adjacent repeats can have three orientations: tandem (same strand), converging (complementary strand, same direction) or diverging (complementary strand, opposite direction). FORRepeats can detect all of the three orientations providing the sequence is concatenated with its complement.

FORRepeats can also detect helical regions in the secondary structure of RNA providing sequences are concatenated with their complement.

Experiments to determine the accuracy of FORRepeats show that the present version is better detecting repeats that are longer than 20 bp. This means that it is well-suited for the detection of minisatellites or duplicated genes and segments rather than microsatellites or short motifs. We mentioned earlier that FORRepeats can deal with amino acids but for working with protein sequences, usually short, BLAST is more appropriate than FORRepeats.

A web interface is currently under development. It will enable the user to display repeats according to parameters such as similarity percentage or length. The user will be able to obtain the actual nucleotides in the repeated segments and their lengths and coordinates, by clicking on the lines joining the repeats. Different colors will be used for different lengths of repeats.

In this paper we give some evidence of the accuracy and the time and space efficiency of FORRepeats. A systematic statistical characterization of the repeats found by FORRepeats is under study. Subsequent clustering and determination of statistical significance of the repeats detected by FORRepeats will help in the extraction of the biological significance and hence in the interpretation of the structure and the evolution of genomes.

ACKNOWLEDGEMENTS

This work was partially supported by a NATO grant PST.CLG.977017.

REFERENCES

- Allauzen, C., Crochemore, M. and Raffinot, M. (1999) Factor oracle: a new structure for pattern matching. In Pavelka, J., Tel, G. and Bartosek, M. (eds), *SOFSEM '99, Theory and Practice of Informatics*, Lecture Notes in Computer Science, 1725, Springer, Milovy, Czech Republic, Berlin, pp. 291–306.

- Altschul,S., Madden,T., Schäffer,A., Zhang,J., Zhang,Z., Miller,W. and Lipman,D. (1997) Gapped blast and psi-blast: a new generation of protein database search programs. *Nucleic Acids Res.*, **25**, 3389–3402.
- Bennetzen,J.L. (2000) Comparative sequence analysis of plant nuclear genomes: microcolinearity and its many exceptions. *Plant Cell*, **12**, 1021–1029.
- Boucher,Y., Nesbo,C.L. and Doolittle,W.F. (2001) Microbial genomes: dealing with diversity. *Curr. Opin. Microbiol.*, **4**, 285–289.
- Copenhaver,G.P. and Pikaard,C.S. (1996) Two dimensional RFLP analyses reveal megabase-sized clusters of rRNA gene variants in *Arabidopsis thaliana*, suggesting local spreading of variants as the mode for gene homogenization during concerted evolution. *Plant J.*, **9**, 273–282.
- Delcher,A.L., Kasif,S., Fleischmann,R.D., Peterson,J., White,O. and Salzberg,S.L. (1999) Alignment of whole genomes. *Nucleic Acids Res.*, **27**, 2369–2376.
- Heslop-Harrison,J.S., Murata,M., Ogura,Y., Schwarzacher,T. and Motoyoshi,F. (1999) Polymorphism and genomic organization of repetitive DNA from centromeric regions of *Arabidopsis thaliana*. *Plant Cell*, **11**, 31–42.
- Karlin,S., Campbell,A.M. and Mrazek,J. (1998) Comparative DNA analysis across diverse genomes. *Annu. Rev. Genet.*, **32**, 185–225.
- Kurtz,S., Ohlebusch,E., Schleiermacher,C., Stoye,J. and Giegerich,R. (2000) Computation and visualization of degenerate repeats in complete genomes. In *Proceedings of the 8th International Conference on Intelligent Systems for Molecular Biology*. La Jolla, CA, The AAAI Press, pp. 228–238.
- Lefebvre,A. and Lecroq,T. (2000) Computing repeated factors with a factor oracle. In Brankovic,L. and Ryan,J. (eds), *Proceedings of the 11th Australasian Workshop on Combinatorial Algorithms*. Hunter Valley, Australia, pp. 145–158.
- Louis,A., Ollivier,E., Aude,J.-C. and Risler,J.-L. (2001) Massive sequence comparisons as a help in annotating genomic sequences. *Genome Res.*, **11**, 1296–1303.
- Martinez-Zapater,J., Estella,M.A. and Somerville,C.R. (1986) A highly repeated DNA sequence in *Arabidopsis thaliana*. *Mol. Gen. Genet.*, **204**, 417–423.
- Mayer,K. et al. (1999) Sequence and analysis of chromosome 4 of the plant *Arabidopsis thaliana*. *Nature*, **402**, 769–777.
- McCreight,E.M. (1976) A space-economical suffix tree construction algorithm. *J. Algo.*, **23**, 262–272.
- Miller,W. (2001) Comparison of genomic DNA sequences: solved and unsolved problems. *Bioinformatics*, **17**, 391–397.
- Ouzounis,C., Casari,G., Sander,C., Tamames,J. and Valencia,A. (1996) Computational comparisons of model genomes. *Trends Biotechnol.*, **14**, 280–205.
- Pelissier,T., Tutois,S., Deragon,J.M., Tourmente,S., Genestier,S. and Picard,G. (1995) Athila a new retroelement from *Arabidopsis thaliana*. *Plant Mol. Biol.*, **29**, 441–452.
- Round,E.K., Flowers,S.K. and Richards,E.J. (1997) *Arabidopsis thaliana* centromere regions: genetic map positions and repetitive DNA structure. *Genome Res.*, **7**, 1045–1053.
- Schmidt,T. and Heslop-Harrison,J.S. (1998) Genomes genes and junk: the large-scale organization of plant chromosomes. *Trends Plant Sci.*, **3**, 195–199.
- The Arabidopsis Genome Initiative (2000) Analysis of the genome sequence of the flowering plant *Arabidopsis thaliana*. *Nature*, **408**, 796–815.
- Wiehe,T., Guigo,R. and Miller,W. (2000) Genome sequence comparisons: hurdles in the fast lane to functional genomics. *Brief Bioinform.*, **1**, 381–388.
- Zhang,Z., Schwartz,S., Wagner,L. and Miller,W. (2000) A greedy algorithm for aligning DNA sequences. *J. Comput. Biol.*, **7**, 203–214.