Fragrep: An Efficient Search Tool for Fragmented Patterns in Genomic Sequences

Axel Mosig[1,5]*, Katrin Sameith[2], and Peter Stadler[2,3,4]

[1] *Department of Combinatorics and Geometry, CAS/MPG Partner Institute for Computational Biology, Shanghai Institutes for Biological Sciences, Chinese Academy of Sciences, Shanghai 200031, China;* [2] *Bioinformatics Group, Department of Computer Science and Interdisciplinary Center for Bioinformatics, University of Leipzig, Leipzig D-04103, Germany;* [3] *Institute for Theoretical Chemistry, University of Vienna, Vienna A-1090, Austria;* [4] *The Santa Fe Institute, Santa Fe, NM 87501, USA;* [5] *Max Planck Institute for Mathematics in the Sciences, Leipzig D-04103, Germany.*

Many classes of non-coding RNAs (ncRNAs; including Y RNAs, vault RNAs, RNase P RNAs, and MRP RNAs, as well as a novel class recently discovered in *Dictyostelium discoideum*) can be characterized by a pattern of short but well-conserved sequence elements that are separated by poorly conserved regions of sometimes highly variable lengths. Local alignment algorithms such as BLAST are therefore ill-suited for the discovery of new homologs of such ncRNAs in genomic sequences. The Fragrep tool instead implements an efficient algorithm for detecting the pattern fragments that occur in a given order. For each pattern fragment, the mismatch tolerance and bounds on the length of the intervening sequences can be specified separately. Furthermore, matches can be ranked by a statistically well-motivated scoring scheme.

Key words: Fragrep, non-coding RNA detection, fragmented pattern, *Dictyostelium discoideum*

Introduction

Methods for detecting non-coding RNAs (ncRNAs) in genomic sequence data have been a topic of intense research. While techniques for detecting protein-coding genes can rely on universal characteristics such as start and stop codons, triplet amino acid codes, or ribosome binding sites, there are no corresponding characteristics known in ncRNAs. Early approaches to ncRNA detection were designed for specific types of RNAs, in particular tRNAs (*1*). Other approaches to ncRNA searching were designed for detecting arbitrary ncRNA classes, typically based on the conserved sequences or secondary structure elements. The RNAMotif (*2*) tool allows to describe a search pattern consisting of conserved stems and helices, while the ERPIN tool (*3*) allows to annotate an alignment with secondary structure information, which is then used as a search pattern. The INFERNAL tool (*4*) also derives its query from a multiple alignment. In contrast to ERPIN, however, the alignment is translated into a stochastic context-free grammar, which is then used for the (quite time-demanding) task of scanning genomic sequences. In general, the computational complexity of searching RNAs increases with the complexity of the search pattern, limiting the use of such methods for genome-wide surveys.

In this paper, we present Fragrep, an efficient tool that is optimized for this kind of sequence-based searches. The approach implemented in the Fragrep tool is based on an elementary way of describing search patterns, allowing a highly efficient and hence genome-wide application. This approach is particularly fruitful for the classes of ncRNAs that contain stem or loop regions with well-conserved sequence patterns, such as Y RNAs and vault RNAs (*5–7*), which, however, are interrupted by non-conserved sequences of highly variable lengths. Compared to RNAMotif, we provide a statistically well-motivated ranking scheme, which relieves the user from defining an individual scoring scheme as in RNAMotif. On the other hand, Fragrep does not search for explicit secondary structure constraints.

The problem of efficiently searching a large sequence database for interrupted sequence patterns is also relevant in the context of other ncDNA motifs,

*** Corresponding author.**
E-mail: mosig@sibs.ac.cn

for example cis-regulatory modules (*8*). In this context, the approach investigated in Fragrep is complementary to the motif discovery procedures such as BioOptimizer (*9*) or Bipad (*10*): once a suitable motif has been discovered, Fragrep can be applied to scan genome databases for such patterns (or constellations of patterns). In some cases, the fragmented patterns are informative enough to be clearly distinguished between true and false positives. In many other cases, however, Fragrep can at least act as an efficient filtering technique. The Fragrep tool can be downloaded from the URL http://www.bioinf.uni-leipzig.de/Software/fragrep/.

Algorithm

Suppose that the ncRNA of interest contains k conserved sequence fragments, denoted by $C_1, \ldots, C_k$, which occur in a given order in a set of known examples. In practice, the fragment C_i is obtained as the consensus sequence of conserved blocks in a multiple alignment. Scanning a genome T for these blocks, we expect to find a non-conserved sequence segment X_i between any two fragments C_i and C_{i+1}. Fragrep solves the problem of determining whether there are sequences $X_1, \ldots, X_{k-1}$ so that the string $C_1X_1C_2X_2 \ldots X_{k-1}C_k$ is contained as a substring in T. Additionally, Fragrep can take into account two further aspects:

Gap length bounds: For each X_i, the user can specify the upper and lower bounds of the length, denoted by u_i and ℓ_i, respectively; only the matches satisfying $\ell_i \leq |X_i| \leq u_i$ will be taken into account by Fragrep.

Mismatches: The fragment C_i does not need to match the corresponding sequence part of T exactly; the user can specify the number of mismatches (m_i). Denoting C_i' as the modified fragment of C_i by at most m_i arbitrary mismatches, Fragrep will report the occurrences of $C_1'X_1C_2'X_2 \ldots X_{k-1}C_k'$ as well.

We also refer to the string $C_1X_1C_2X_2 \ldots X_{k-1}C_k$ satisfying all these constraints as a matching subsequence of T. Similar features are incorporated in other tools such as RNAbob (ftp://ftp.genetics.wustl.edu/pub/eddy/software/rnabob-2.1.tar.Z), which is based on a nondeterministic finite state machine with node rewriting rules instead of the dynamic programming approach used by Fragrep.

The algorithm underlying Fragrep essentially works in two steps:

1. For each $i \in [1:k]$, compute a list of all occurrences of C_i in T.

2. Apply a dynamic programming algorithm to the lists computed in the first step in order to find all matching subsequences in T.

Performing step 1 is straight-forward. As a result, we obtain an ordered list of indices $Y_i = \langle y_{i,1}, y_{i,2}, \ldots, y_{i,L_i} \rangle$, with $y_{i,j}$ denoting the position of the j^{th} occurrence of C_i in T, and L_i denoting the number of occurrences of C_i in T. Using $Y_1, \ldots, Y_k$, we now set up a graph $G = (V, E)$ with the vertex $V = \{(i,j) \mid i \in [1:k],\ j \in [1:L_i]\}$ and an edge between (i,j) and $(i+1, j')$, whenever the corresponding occurrences of C_i and C_{i+1} satisfy the upper and lower bounds for the gap in between. Obviously, any path of length $k-1$ in G corresponds to a valid occurrence of $C_1, \ldots, C_k$ in T. For each (i,j), $i \in [1:k]$ and $j \in [1:L_i]$, we now compute

$$M_{i,j} := \begin{cases} 1 & \text{if } T \text{ contains the model} \\ & \langle C_1, \ell_1, u_1, \ldots, C_i, \ell_i, u_i \rangle \\ & \text{with } C_i \text{ occurring at position } y_{i,j} \\ 0 & \text{otherwise.} \end{cases}$$

Apparently, we have a valid occurrence of $C_1, \ldots, C_k$ in T whenever we have $M_{k,j} = 1$ for some $k \in [1:L_k]$. Furthermore, we have $M_{1,j} = 1$ for all $j \in [1:L_1]$ (since every occurrence of C_1 is a valid occurrence up to the first fragment). Now, using the graph G, we can compute all $M_{i,j}$ for $i > 1$ as

$$M_{i,j} = \begin{cases} 0 & \text{if } M_{i-1,j'} = 0 \\ & \text{for all } \big((i-1,j'),(i,j)\big) \in E \\ 1 & \text{otherwise.} \end{cases}$$

Starting with $i = 1$, the $M_{i,j}$ values can be computed using dynamic programming in a straightforward way. Now, each non-zero entry in the k^{th} row of M indicates at least one valid match. Defining $L := \max_i L_i$, altogether $k \cdot L$ matrix entries are computed, so that the overall time complexity for computing M is $O(kL)$; enumerating all μ matching subsequences of T can be done in $O(kL + \mu)$ time. The match number μ is proportional to the tolerance allowed by the gap length bounds, that is, $u_i - \ell_i$. While in principle μ is bounded by $O(L^2)$ (since each occurrence of C_1 might yield one matching subsequence for each occurrence of C_k), one is naturally interested in queries that produce few significant rather than an abundance of insignificant matches. Hence, for all practical purposes, $O(kL)$ should be seen as the dominating term in the running time.

Note that the above procedure can be easily adapted to start dynamic programming with the most informative sequence C_a rather than C_1 by starting in the a^{th} row of M. This increases the search efficiency and in practice leads to a significant speedup, in particular when short or ambiguous fragments are part of the pattern. The C++ implementation of Fragrep has been optimized in this and several other algorithmic details to improve the runtime. The current implementation only searches for gap-free C_i patterns. This limitation could be relaxed by employing a different pattern matching algorithm that allows gaps at a prescribed gap function. The use of the exact Smith-Waterman local sequence alignment algorithm for this purpose is possible but computationally quite expensive. In our applications, gaps were restricted to specific positions. In this case, it is more efficient to break the search pattern into smaller units linked by short intervals of variable lengths. Since the identification of these breaks is self-evident in the applications under our consideration, one obtains significantly more specific query patterns than for gapped alignment matches.

A major issue in evaluating the quality of the matches produced by the above procedure is to assess how surprising a given match is, that is, how likely it is to be observed in a random sequence. To this end, Fragrep provides p- and E-value-like ranking schemes that are computed from a dinucleotide-based Markov model.

In order to adapt the Markov models to the occurrences of a fragmented rather than a contiguous sequence pattern, we let E_j denote the event of $C_1, \ldots, C_j$ being observed at least once in a sequence of length N in the given order and satisfying the distance constraints given by the respective upper and lower bounds u_i and ℓ_i for each C_i (formally dealing with a probability space over all sequences of length N). Furthermore, let $q(j, L)$ denote the event of observing C_j at least once in a sequence of length L. Our main interest obviously is to determine the probability $p(E_k)$.

We start with computing the probability $p(E_1)$. Denote $\mathbf{M}_T$ as the first order Markov model resulting from the dinucleotide frequency distribution in T, we may compute $P_T(C_1) := P(C_1|\mathbf{M}_T)$ as the probability of the fragment C_1 being produced by $\mathbf{M}_T$. In order to obtain $p(E_1)$, we assume that the probability of C_1 being produced at a position x in a sequence of length N is independent of the probability of C_1 being produced at any other position y in T—note that this assumption holds for the fragments that do not contain any substring of length 2 more than twice, and, for all practical purposes, is a sufficiently good approximation for the fragments that are short and contain only few repetitive substrings. Now, under this assumption of independence, we obtain

$$p(E_1) = q(1, N) = P_T(C_1) \sum_{0 \le \nu < N - |C_1|} \left(1 - P_T(C_1)\right)^{\nu} \tag{1}$$

The probabilities $q(j, L)$ can be computed analogously for arbitrary j and L. It is now easy to see that for $j \in \{2, \ldots, k\}$, we have

$$p(E_j) = p(E_{j-1}) q(j, u_j - \ell_j) \tag{2}$$

(since, in a sense, C_j needs to be generated by $\mathbf{M}_T$ in a sequence of length $u_j - \ell_j + |C_j|$) so that we finally obtain

$$p(E_k) = p(E_1) \prod_{2 \le j \le k} q(j, u_j - \ell_j) \tag{3}$$

As described above, $p(E_k)$ is the probability of observing at least one exact match of the given fragmented pattern in a sequence of length N. This value can be easily adapted to the scenario involving occurrences with a certain number of mismatches by modifying the probabilities $P_T(C_j)$ accordingly. Since different matches obtained by Fragrep generally have different mismatches in different positions, we can also compute the analogous probabilities for the individual matches detected by Fragrep. Finally, $w(E_k) := -\log\left(p(E_k)\right)$ provides a convenient and statistically well-motivated ranking scheme for the matches.

Results and Discussion

We used Fragrep to studying the evolution of a class of ncRNAs in the slime mold *Dictyostelium discoideum* that was discovered in an experimental survey by Aspegren *et al* (*11*). We searched the genomic sequence (*12*) for the type-I ncRNAs using the following simple pattern:

```
0   0  GTTGRCCTTACAGCAA   2
0 120  GTCAACTG           2
```

The first two columns contain the minimal and maximal distance between the pattern fragment (always 0 for the first fragment, of course), the last column is

the maximal number of mismatches that is tolerated in each fragment. The gap length in the sequences derived from the study by Aspegren *et al* (*11*) ranges between 58 and 88 nucleotides, so that 120 is a reasonable choice for the gap length's upper bound. We recovered 45 candidates, of which 34 were sufficiently similar to the experimentally determined sequences to be alignable. The other 11 very divergent sequences were not included in the further analysis. A neighborjoining tree summarizing both known sequences and the novel candidates detected by Fragrep is displayed in Figure 1. We find that the class-I ncRNAs are located in small clusters in all six chromosomes. Interestingly, there are two subclasses, denoted by A and B, that alternate in the larger clusters, even though their directions on the chromosomes do not seem to follow a simple rule.

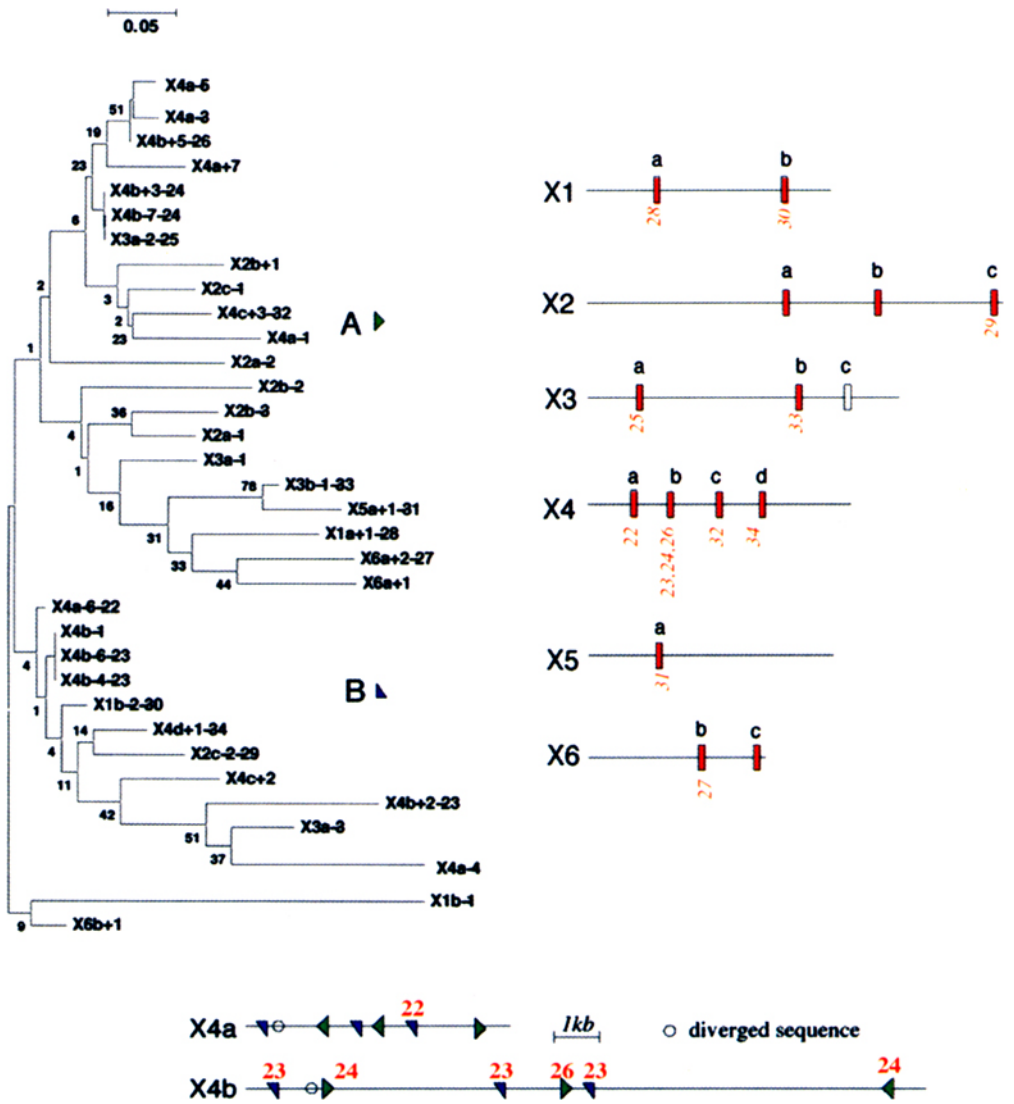

Fig. 1 The type-I ncRNAs from *Dictyostelium discoideum*. **Top Left**: The phylogenetic tree (neighborjoining method) suggests that there are two major subgroups, labeled as **A** and **B**. Leaf labels refer to the positions of the corresponding occurrences within the genome; for instance, X4a-5 refers to the fifth member within cluster *a* in Chromosome 4 (see the middle part of the figure); plus (+) or minus (−) indicates the occurrences in the 5′ or 3′ direction. **Top Right**: The type-I ncRNAs that appear in clusters on all chromosomes. The clusters are labeled by lower case letters, and the italic numbers below the clusters indicate the DdR- numbers of the expressed RNAs from the experimental survey by Aspegren *et al* (*11*). **Bottom**: The organization of the two largest clusters *a* and *b* located at Chromosome 4. Note that type A and type B copies alternate. The other type-I ncRNA clusters consist of no more than three sequences.

In order to evaluate the performance of the algorithm underlying Fragrep, we used the query

```
0 0    TRGCNNAGYGG 1
0 100  GGTTCGANTCC 1
0 100  GGTTCGANTCC 1
```

derived from the vault RNA A-, B1-, and B2-box consensus structures in Kickhoefer *et al* (*7*) to scan the whole human genome. The query consisted of three fragments, each of which was 11 nucleotides long. Scanning all chromosomes of the human genome took less than 10 minutes on a standard desktop computer with a 2 GHz processor and 1 GB main memory. Further results from scanning the human as well as the mouse, rat, and dog genomes are listed in Table 1.

Table 1 Surveys of Mammalian Genomes for vault RNA Candidates

Genome	Size (Mb)	Runtime (mm:ss)	No. of matches
Homo sapiens	2,980	9:24	14
Mus musculus	2,561	7:36	35
Rattus norvegicus	2,640	8:33	44
Canis familiaris	2,454	7:55	768

In order to obtain an estimation of the influence of high gap-length tolerance on the running time, we modified the above query to allow for gap lengths up to 20,000, resulting in an increase of the running time by less than five folds. Note that this increase in fault tolerance also increased the number of matching subsequences to the order of hundreds of thousands, so that the significance of such highly fault tolerant queries is already limited from a practical point of view. In spite of this undue fault tolerance, the running time remains within acceptable bounds.

These examples demonstrate that Fragrep can be used for systematic surveys of eukaryotic genomes. The application of standard multiple alignment tools such as ClustalW or Dialign to a relatively small set of representatives of an ncRNA class can be used to determine conserved sequence patterns, which can be turned into Fragrep queries in a straight-forward manner. The Fragrep tool can then be employed to find additional members of the ncRNA family in related genomes. This approach yields significant matches where other sequence search tools such as BLAST fail to report useful results, while the structure based approaches such as INFERNAL (*4*) are too costly. Naturally, Fragrep is not limited to ncRNA detection; the search for specific constellations of transcription factor binding sites is another potential application. Furthermore, the approach could be easily adapted to searching peptide motifs in protein databases.

Acknowledgements

This work was supported by the Bioinformatics Initiative (BIZ-6/1-2) of the German Research Association (DFG).

References

1. Lowe, T.M. and Eddy, S.R. 1997. tRNAscan-SE: a program for improved detection of transfer RNA genes in genomic sequence. *Nucleic Acids Res.* 25: 955-964.
2. Macke, T.J., *et al.* 2001. RNAMotif, an RNA secondary structure definition and search algorithm. *Nucleic Acids Res.* 29: 4724-4735.
3. Gautheret, D. and Lambert, A. 2001. Direct RNA motif definition and identification from multiple sequence alignments using secondary structure profiles. *J. Mol. Biol.* 313: 1003-1011.
4. Eddy, S.R. 2002. A memory-efficient dynamic programming algorithm for optimal alignment of a sequence to an RNA secondary structure. *BMC Bioinformatics* 3: 18.
5. Farris, A.D., *et al.* 1999. Conserved features of Y RNAs revealed by automated phylogenetic secondary structure analysis. *Nucleic Acids Res.* 27: 1070-1078.
6. Teunissen, S.W. 2000. Conserved features of Y RNAs: a comparison of experimentally derived secondary structures. *Nucleic Acids Res.* 28: 610-619.
7. Kickhoefer, V.A., *et al.* 2003. Identification of conserved vault RNA expression elements and a non-expressed mouse vault RNA gene. *Gene* 309: 65-70.
8. Segal, E. and Sharan, R. 2005. A discriminative model for identifying spatial cis-regulatory modules. *J. Comput. Biol.* 12: 822-834.
9. Jensen, S.T. and Liu, J.S. 2004. BioOptimizer: a Bayesian scoring function approach to motif discovery. *Bioinformatics* 20: 1557-1564.
10. Bi, C. and Rogan, P.K. 2004. Bipartite pattern discovery by entropy minimization-based multiple local alignment. *Nucleic Acids Res.* 32: 4979-4991.
11. Aspegren, A., *et al.* 2004. Novel non-coding RNAs in *Dictyostelium discoideum* and their expression during development. *Nucleic Acids Res.* 32: 4646-4656.
12. Kreppel, L., *et al.* 2004. dictyBase: a new *Dictyostelium discoideum* genome database. *Nucleic Acids Res.* 32: D332-333.