VOL.2 NO.2September 2013 - ISSN 2233 – 1859

Operations Research Society in Bosnia and Herzegovina

Southeast Europe Journal of Soft Computing

Available online:www.scjournal.com.ba

Multiple Methods for Genome Filtering

Alma Husagic Selman

International University of Sarajevo, Faculty of Engineering and Natural Sciences, Hrasnicka Cesta 15, Ilidža71210 Sarajevo, Bosnia and Herzegovina

Article Info

Article history:
Received 17 Sep.2013
Received in revised form 17 Oct 2013

Keywords:
Filtering algorithms, classification of filtering algorithms

Abstract

Filters are fast algorithms, which help to preprocess DNA sequences in order to reduce the time and complexity of approximate motif search. Multiple filtering methods exist, and this paper classifies the filtering algorithms based on their approach, numerical analysis or digital signal processing, and it briefly reviews both classes of filters. The paper also reflects on filters currently used in popular software for genomic processing.

1. INTRODUCTION

Genome represents an essential instrument needed for understanding the biology of an organism, and as such studying genome and its DNA sequence represents one of the main problems in bioinformatics, or more precisely in genomics. The length of the DNA, which, for human genome, counts 3.2x109 base pairs in length, makes the genome search very complex and expensive problem.

Genome constitutes of complete set of DNA, including all the genes. In short, it carries complete hereditary information of an organism. DNA is a fundamental substrate of a genome, and it represents a chain or sequence of nucleotides. And nucleotide is a base joined to a 2-deoxyribose sugar molecule that forms part of the backbone of the double helix (Figure 1) (Anastassiou, 2001). Four DNA nucleotides exist, namely Adenine (A), Cytosine (C), Guanine (G) and Thymine (T) [Buhler Thesis]. For a computer scientist, DNA sequence represents a very long string of alternating characters from the space of four alphabets that correspond to four respective nucleotides {A, C, G, T}.

Common problem in genomics is known as motif search. Motif search considers the discovery of signals or motifs in the set of DNA sequences. DNA sequences are interesting segments cut at certain positions from complete DNA. As the DNA sequences might be very long (hundreds or thousands of nucleotides), problem of motif search becomes computationally very expensive (Raphael et. al. 2004). Adding to this the fact that the motifs, eitherdue to substitutions or indels, may be subject of mutation, motif search becomes even more complex.

Motif search may involve exact or approximate search. In exact motif search, algorithm scans through all DNA sequences in order to find the exact match. The complexity of the algorithms is generally directly proportional to the length of the search sequences, and the size of the motif (Jones &Pevzner, 2004). These algorithms can be used only when the search motif is known exactly, however, when it comes to the search for possible mutations in the motif, the method cannot be used.

The alternative method is known as approximate method search and it allows k number of errors to occur in the match (Jones &Pevzner, 2004). Approximate motif search algorithm scans through the DNA sequences looking for a match, which deviates from the motif k number of characters. Based on different algorithms, number of errors in the match may vary. However the complexity of the algorithms is still directly related to the length of the DNA sequences and the length of the motif (Navarro, 2001).

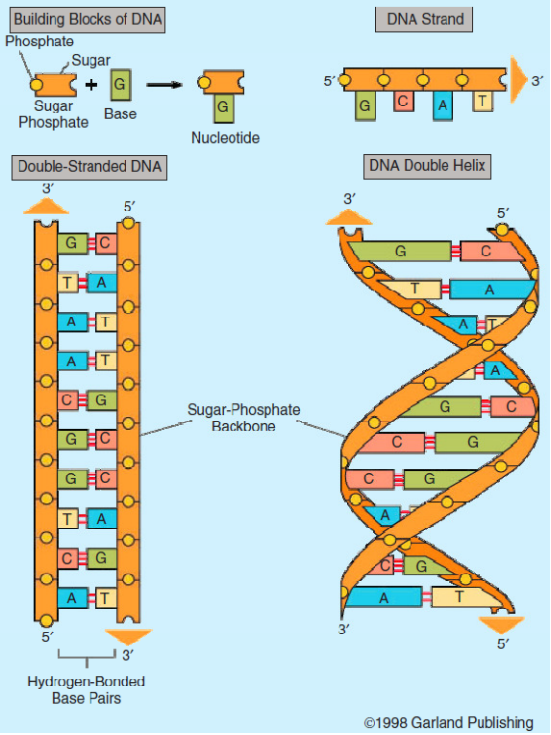

Figure 1: The structure of DNA and nucleotide (Anastassiou, 2001)

The complexity of the motif search process may be reduced using filtering (Buhler, 2001). Filtering helps in finding those segments of the sequences which might contain interesting similarity to the searched motif. In such a way, the search size reduces to the fraction of the initial size (Buhler, 2001). The filtering is based on the fact that it is easier to find and eliminate the motifs which do not match the conditions, making motif search area smaller for the approximate or exact search algorithms. Filters are fast algorithms since they search for the necessary, but not sufficient conditions in order to pass the string (Baeza-Yates & Navarro, 2001).

The output of the filters is a rough estimate of the possible matches (Menon, 2007). This rough estimate still must be processed using approximate or exact match search, but it greatly reduces the processing time of the later algorithms.

There are two main approaches in designing filters (Anastassiou, 2001). First approach is traditional one, where DNA sequence is considered as stream of characters and is processed based on numerical methods. Examples of such filters are Superimposed Automata, Counting Filters,Random or Uniform Projection Filtering etc. (Raphael et al, 2004, Buhler &Tompa, 2002).
The other approach uses the fact that the genomic sequence can be viewed as digital, and as such digital signal processing (DSP) approaches to filtering can be applied. Huge number of DSP filters exists, and incorporating them in software for genomic search may greatly improve the software performance (Anastassiou, 2001).

This paper reviews these both approaches to filtering, and reflects on their performance and software usage. The paper is organized as follows: Section 2 discusses the basic concepts of both filtering approaches, Section 3 talks about the filter types and popular filters, Section 4 compares the filters and Section 5 concludes the paper.

2. BASIC CONCEPTS

This section represents the important concepts needed to understand the background behind the functionality of filters. As mentioned in the previous section, two approaches exist in filtering, traditional numerical analysis approach and DSP approach.

2.1 Numerical Analysis Approach

Suppose we are given two sequences, text S of size n, and a pattern P of size q. The substring matching problem with k-mismatches (Pevzner& Waterman, 1995) requires scanning of the initial text sequence S, in order to search for a motif from the pattern P. The motif can be represented through a sub-pattern L of size l, generally called l-mere. l-mere controls the size of the text window, i.e. the size of a substring that is to be checked with given l-mere (Pevzner& Waterman, 1995). In each iteration, the text window is moved one character. The total number of substrings can be calculates as *(n-l+1)* (Jones &Pevzner, 2004).

Conditions for filtering based on numerical analysis method are grounded on the following two Lemmas:

Lemma 1: If text S matches pattern P with k errors, and pattern $P = (P_1, P_2 \ldots P_j)$ (concatenation of sub patterns), then S includes a segment that matches at least one of the $P_{i's}$, with *k/j* errors.

For Proof of Lemma 1, refer to (Baeza-Yates & Navarro, 1999)

Lemma 2: If there are $i<j$, such that *edit distance* $ed(S_{i\ldots j}, P) \leq k$, then $S_{j-q+1\ldots j}$includes at least m-k characters of P. (Ukkonen, 1992, Baeza-Yates & Navarro, 2001)

For Proof of Lemma 2, refer to (Ukkonen 1992)

Lemma 1 shows that a large search problem can be divided into multiple smaller problems, i.e. given a large search pattern and a large text sequence, it is possible to divide the search pattern in smaller parts, and then search the text sequence for these smaller parts of the pattern.

Lemma 2 shows that, based on finding sufficient number of pattern characters in a text sequence window (i.e. one l-

mere-sized text subsequence), it is possible to design a filter for approximate search, so as to properly separate the probable string matches (Baeza-Yates & Navarro, 2001)

Most of the current software used either for longer or shorter DNA sequence search are based on numerical analysis. Examples are MEGABLAST, SSAHA and BLAST-like alignment tool for longer DNA sequence search, and SOAP, MAQ and SHRIMP for shorter DNA sequence search.

Many of these algorithms can detect single nucleotide changes. However they require a hash table to be built for each pattern or reference. The hash table then needs to be searched for alignment matches.

2.2 DSP Approach

According to Tuqan&Rushdi (2008), most of the existing signal-processing techniques depend on spectral analysis of the DNA sequence using the short-time discrete Fourier transform (ST-DFT). Like before, DNA sequence is denoted as a string of characters, where each character denotes a nucleotide. Given DNA sequence of length N, each character within the sequence is assigned a numerical value. This mapping may not be unique, but it must have an underlying biological interpretation (Tuqan&Rushdi, 1008), and it should not alter the structure of the DNA sequence under study.

Voss representation, although not the best, is one of the most popular mapping methods (Arniker& Kwan, 2012).Voss representation uses the DNA sequence to generate four binary indicator sequences xI(n), I ϵ{A, C, G, T}, one for each nucleotide. Binary indicator sequences are generated by setting 1 in the case of presence of the nucleotide and 0 in the case of its absence (Tuqan&Rushdi, 2008, Voss, 1992). Table 1 shows Voss representation for the genome subsequence from Figure 1.

Table 1: Vos representation of DNA sequence

String:	G	T	A	A	C	G	G	T	C	A
$x_A(n)$	0	0	1	1	0	0	0	0	0	1
$x_C(n)$	0	0	0	0	1	0	0	0	1	0
$x_G(n)$	1	0	0	0	0	1	1	0	0	0
$x_T(n)$	0	1	0	0	0	0	0	1	0	0

For pure DNA character strings, the binary indicator sequences $x_i(n)$, provide a four-dimensional representation of the frequency spectrum of the character string.

Using procedure explained in (Anastassiou, 2001) contributions of all four characters can be combined in the following quantity:

$$S[k] = |x_A[k]|^2 + |x_C[k]|^2 + |x_G[k]|^2 + |x_T[k]|^2$$

$$k = 0, 1, \ldots, N-1$$

$S[k]$ is a measure of the total spectral content of the DNA string at frequency k.

The dimensionality may be reduced to three, as it is really necessary to know only the three binary indicator sequences and the fourth one can be directly derived.

Once the spectrum has been defined, it is possible to apply other relevant DSP methods for processing of the spectrum. This includes the filtering as well. Detailed description of filtering of the DNA spectrum can be found in (Tuqan&Rushdi, 2008)

3. FILTERING ALGORITHMS

Filtering algorithms help in reducing the size of the approximate or exact search problem by initially scanning through the sequence and eliminating part of the string which contain no similarities to the searched motif. The filtering algorithms are generally fast, but their performances vary. This section gives a brief overview of existing types of filters based on numerical analysis and DSP approaches, and it briefly discusses filters used in popular genomic software.

3.1 Algorithms based on Numerical Analysis Approach

Filtering algorithms based on numerical analysis may be divided in four different groups (Baeza-Yates & Navarro, 2001):

Filters based on Bit Parallelism: Bit parallelism approach uses the advantage of parallelism that is possible to achieve on the bit level inside the computer word. In such a way, the number of operations may be cut down by a factor of width of a computer word. The algorithm requires generation of the non-deterministic finite automata (NFA) that is used in pattern matching. NFA contains rows, which represent error levels, and columns, which represent pattern characters. Operations upon adding character to the pattern include shifts (to the right, down and diagonal), and or operation. After the pattern passes NFA, the matching results mush be stored, and this is done using mask b[], which is built for each pattern. The algorithm is linear and can be used to search for multiple patterns.

Filters based on Superimposed Automata: this is extension of the bit parallelism approach in such that, in order to search r patterns, P1…Pr, with at most k errors, it is possible to build the automation for each pattern, and then superimpose all the automata. The superimposition is implemented using bitwise and on all the tables. The result is stored in a single b[] table, and table match at position i matches with i-th character of any of the patterns. Following this, the automation is built in similar way like in bit parallelism approach. After the string is passed

through the automata, it needs to be verified, and for verification methods, refer to (Baeza-Yates & Navarro, 2001). The algorithm runs linearly.

Filters based on Exact Searching: is based on Lemma presented in Section 2. It assumes that, if k errors are allowed, it is possible to divide the pattern into k+1 pieces, and use the exact matching algorithm on all the pieces. At least one of the pieces must be present with no errors in each occurrence of the partially matching pattern. Algorithm is linear. It is possible to apply it on multiple pattern search, but in that case the algorithm performance is logarithmic.

Counting Filters: Counting Filters are based on counting the frequency of occurrence of common characters between the pattern and the window. The filter is based on Lemma 2, and it checks the n-letters long window. The algorithm keeps a table A[], where the number of occurrences of each character within the pattern is tracked. For example, when character 'G' is met, the number in A[G] is incremented. To advance the window, the next character in the text, just right of the window is added to the window, and the first character is removed from the window. Then the count is recalculated for the new-coming character.

This algorithm is linear, and number of operations per character is very small. It is also possible to keep many counters in parallel, if the string is divided into multiple substrings. Thus, given the M substrings. The algorithm will need to generate M tables MA[], which will pack all the A[] tables for each substring.

3.2 Algorithms based on DSP Approach

DSP approach to genome filtering is new, and not much work is done on DSP filtering for genomic sequences. However, DSP represents the future for genomic filtering, as the digital approach implemented in DSP is possible to match the developments in current technology.

Since spectrograms are generally used for DSP approach in genomics (Anastassiou, 2001, Tuqan&Rushdi, 2008), it is possible to use any kind of filters that deal with spectrograms, such as Finite Impulse Response (FIR), IIR, DFT filter, or combination of those. For detailed description of these filters, refer to (Oppenheim & Schafer, 2009, Crochiere&Rabiner, 1983).

According to Stranneheim et al (2010), Bloom filter provides very good performance when it comes to the DNA sequence search. In their work, Stranneheim et al proved that implementing software with Bloom filter outperforms the existing genome softwares by factor of 20 or 30, depending on the size of the search sequence.

Since a Bloom filter can be used to determine if an element belongs to a reference set or not, it is suitable for the approximate motif search problem. It is implemented using a bitmap and k-associated hash functions. To check if an element will be found in the filter, the k-hash functions are applied to the element and the resulting positions are looked up in the bitmap. If all bits are 1, the element is most likely part of the set; otherwise, the element is definitely not in the set. The algorithm is linear, and it is independent of the number of elements already in the set.

Following we discuss the filters used in popular genomic software.

3.3 Filters used in popular Software

Current software for genomic processing are designed either to process the longer or shorter DNA sequences. For processing longer DNA sequences popular software such as MEGABLAST (Zhang et al., 2000), and for shorter DNA sequences, the common software are SOAP (Li et al., 2008b) and SHRIMP (Rumble et al., 2009).

MEGABLAST is using greedy approach in DNA motif search. Since a greedy algorithm is faster than dynamic programming, and the speed is considered a cost, Zhang et al. (2000) came to realize and implement a new greedy alignment algorithm with good performance. The algorithm relies on the differences between the patterns, instead of their the similarities. Any greedy algorithm is based on the distances between the strings, so is this algorithm, but rather than initializing the distance identity to very large number, , this algorithm initializes it to very small number. In the short, string search (alignment) is measured by counting the number of character differences, i.e., the number of columns that do not align identical nucleotides. The distance between the strings is defined as the minimum number of differences in any alignments of those strings.

Although the MEGABLAST is not directly using the filter, the concept of the greedy algorithm is similar to filtering, since according to Baeza-Yates & Navarro, (2001), filtering is based on the fact that it is easier to locate and remove the motifs which do not match the conditions, than those that do.
SOAP is short oligonucleotide alignment program which converts sequences into numeric data type using 2-bits-per-base encoding. The processed sequences are checked with exclusive-OR comparison with the reference sequence and result is stored. The value is then checked to see how many bases are different. This way of preprocessing is similar to bit parallelism filters. The algorithm is very fast, and it outputs the identical alignments.

SHRiMP - the SHort Read Mapping Package is a set of algorithms and methods used to map short reads of a genome. According to Rumble et al (2009) "The method is based on a fast read mapping technique, separate thorough alignment methods for regular letter-space as well as AB SOLiD (color-space) reads, and a statistical model for false positive hits.". The algorithm uses Bloom filter which is discussed in previous section.

4. CONCLUSION

Filtering methods help reduce the space for string search or string alignment problem. Multiple algorithms for filtering exist, and they can be generally divided based on their mathematical approach – numerical analysis approach and DSP approach based filters. Traditionally numerical analysis approache filters were used, however, with the advancement in the computational devices and speed, genomic processing is shifting to DSP space. As such, DSP filter have recently shown very good performance, outrunning the traditional filter by the factor of 20. This paper gave a brief overview of existing filtering methods and briefly reflected on three popular softwaresfor genomic processing.

REFERENCES

Anastassiou D., "Genomic Signal Processing", IEEE Signal Processing Magazine, 2001.

Arniker S., Kwan H. K. "Advanced Numerical Representation of DNA Sequences", 2012 International Conference on Bioscience, Biochemistry and Bioinformatic, IPCBEE vol.3 1(2012) © (2012)IACSIT Press, Singapoore

Baeza-Yates R., Navarro G. "Faster Approximate String Matching", Algorithmica, 23 (2): 127-158, 1999.

Baeza-Yates R., Navarro G. "New and Fast Filters for Multiple String Matching", Random Structures and Algorithms (RSA), 2001.

Buhler J. "Search Algorithms for Biosequences Using Random Projection", Doctoral Thesis, 2001.

Buhler J., Tompa M. "Finding Motifs Using Random Projections", Journal of Computational Biology, Vol. 9, No. 2, 2002. pp. 225–242.

Crochiere R. E. &Rabiner L. R. "Multirate Digital Signal Processing", Prentice Hall, 1983. ISBN-10: 0136051626.

Jones N. C., Pevzner P. A. "An Introduction to Bioinformatics Algorithms". Computational Molecular Biology, A Bradford Book, The MIT Press, Cambridge, Massachusetts, London, England, 2004.

Kent,W.J. (2002) BLAT—the BLAST-like alignment tool. Genome Res., 12, 656–664.

Melsted P., Pritchard J. K. "Efficient counting of k-mers in DNA sequences using a bloom filter", BMC Bioinformatics, 2011.

Menon A. K. "Random projections and applications to dimensionality reduction", Unpublished Thesis, 2007.

Navarro G. "A Guided Tour to Approximate String Matching". ACM Computing Surveys, Vol. 33, No. 1, March 2001, pp. 31–88.

Ning,Z. et al. (2001) SSAHA: a fast search method for large DNA databases. Genome Res., 11, 1725–1729.

Oppenheim A. V. & Schafer R. W. "Discrete-Time Signal Processing", PreTeX, Inc. Oppenheim book, Pearson, 2009.

Pevzner P. A. & Waterman M. S. "Multiple Filtration and Approximate Pattern Matching" Algorithmica Springer, 1995.

Raphael B., Liu L. T., & Varghese G. "A Uniform Projection Method for Motif Discovery in DNA Sequences", IEEE Transactions on Computational Biology and Bioinformatics, Vol. 1, No. 2, April-June 2004.

Rumble,S.M. et al. (2009) SHRiMP: accurate mapping of short color-space reads. PLoSComput. Biol., 5, e1000386

Sharma R. &Kaushik A. "Exon identification using Filters", International Journal of Enginering Research and Applications (IJERA), ISSN: 2248-9622, Vol. 2, Issue 1, Jan-Feb 2012, pp. 295-298. 2012.

Stranneheim H., Kaller M. Allender T., Andersson B. , Arvestad L., Lundeberg J. "Classification of DNA sequences using Bloom filters", Bioinformatics, Vol 26, No. 13, 2010. Oxford University Press, 2010.

Tuqan J. &Rushdi A. "A DSP Approach for Finding the Codon Bias in DNA Sequences", IEEE Journal of Selected Topics in Signal Processing, Vol. 2, No. 3, June 2008.

Ukkonen E. Approximate string matching with q-grams and maximal matches. Theoretical Computer Science, 1:191-211, 1992.

Voss R. F. "Evolution of Long-range Fractal Correlations and 1/f noise in DNA base sequences". Physical Review Letters, 1992, 68(25):3805-3808.

Zhang,Z. et al. (2000) A greedy algorithm for aligning DNA sequences. J. Comput.Biol., 7, 203–214.