BioInfo Paper — Topic I

Falk Schubert

1 Introduction

In the last years genomic databases grow at a high rate. More and more organisms are being sequenced producing lots of data for further analysis. As part of many questions that come up, comparing sequences with each other is very common task. The alignment of sequences reveals information about the relationship of the input and thus helps to answer questions about their function, evolutionary and structural relation. Since many of these newly retrieved sequences are quite large, fast and efficient methods are required.

2 History

Aligning two sequences has been a long known problem. As early as 1878 Lewis Carrol stated a problem involving sequence alignment. [2] 1950 Hamming developed the sequence theory with the coding theories.[4] A systematic approach was then taken by Levenstein in 1966.[7] As a pioneer for a computational solution of the alignment problem Dayhoff developed first sophisticated methods in 1968.[3] After this fundamental work many other methods have been proposed and are being used in different applications. A following table gives a short timeline of the work:

1970	Needleman-Wunsch Algorithm for global alignment
1977	Gilbert and Sanger developed methods for sequencing DNA initiating large-scale sequencing projects
1981	Smith-Waterman Algorithm for local alignment
1985	FastP Algorithm
1988	FastA Algorithm
1990	BLAST Algorithm

3 Alignment Methods

There a different categories of aligning sequences. For each of these alignment types there are several algorithms that can be used. The basic classes of alignment are: *pairwise alignment, multiple alignment* and *self alignment.*

The first type can be divided into *global alignment* (to look at sequences globally related by common ancestry) and *local alignment* (to look at related sequence segments). In general a pairwise alignment as shown in figure 1 tries to find the best match between two sequences while minimizing penalties for opening gaps, extending gaps and replacing characters in one of the sequences. From a computational point of view there a several ways to implement these methods.

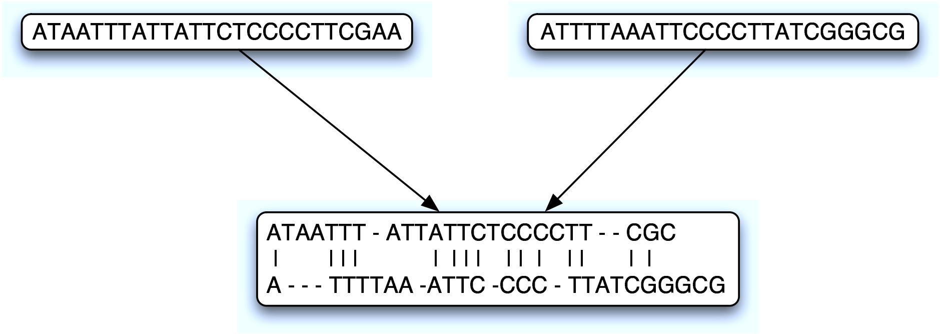
ATAATTTATTATTCTCCCCTTCGAA
ATTTTAAATTCCCCTTATCGGGCG
ATAATTT - ATTATTCTCCCCTT - - CGC
I III I III III II I I
A - - - TTTTAA -ATTC -CCC - TTATCGGGCG

Fig. 1: aligning two sequences – the dash represents a gap whereas a vertical bar shows a match; at positions missing a vertical bar, a substition has occured

1. A dot matrix can be used to create a dotplot. This shows all possible matches of all sequence characters. An investigator can evaluate these plots according to some expertise.

2. The *Needleman-Wunsch Algorithm* [8] builds on an interesting key observation: any prefix of an optimal alignment is also optimal for aligning just the prefix. A *dynamic programming* approach can be used to compute the overall alignment from optimal subalignments. This increases the speed by allowing to store some partial solutions. Although this method is already quite fast, for real-life sequences even faster methods are required. Furthermore this algorithm is used for global alignment, meaning it computes the best overall score for the complete alignnment of two given sequences.

3. Like the previous algorithm the *Smith-Waterman Algorithm* [10] uses *dynamic programming* for the same reason for alignment. However the focus here lays on a local alignment. This means it trys to find the highes local similarity of two sequences.

4. Another key observation of alignment is, that the symbol pattern of an alignment can be derived by statistical analysis from previous patterns. The *WABA Algorithm* [6] uses *Hidden Markov Models* based on that property.

Theses fundamental algorithms are still not fast enough. One can show that the dynamic programming approaches have a time complexity of $O(m + n)$ and a space complexity of $O(m \cdot n)$ – where m and n denote the length of the two input sequences. Further improvments have been developed over the years that increase the speed and reduced the required space of these algorithms. Some of these extensions are:

1. *Hirschberg's Linear Space Alignment*[5] by Miller and Myer: improves time efficency to $O(\min(m, n)$
2. *k-difference Method*: finds the best alignment with at most k character substitutions and improves time efficency to $O(k \cot m)$
3. *Aproximate Matching Algorithms*: these algorithms do not compute the absolute maximum, but a solution that is close to the optimal one; this results in great speed increases

A problem arises when querying short sequences against databases with many large sequences. Even using improved methods a fast answer can only be computed by *heuristic sequence searches.* Two famous algorithms doing this are *FastA* and *BLAST*. They are actually the choice when trying to align small sequences to a database containing genome of an organism.

Last but not least the *Super Pairwise Alignment* algorithm implemented in *DIALIGN2* [9] increases speed by a factor of 15 compared to *BLAST* or *FastA*. Instead looking at each character of the sequence it takes segments of a sequence for alignment.

Besides the *pairwise alignment, multiple alignment* is being used to extract information about the relationship between many sequences. This way evolutionary history or family relations can be revealed. However the complexity of this alignment type is $O(2^k n^k)$ and it has been identified as a *NP-hard problem.* The *multiple alignment* can among many applications be used for protein classification or identification of conserved regions.

References

[1] L. Carrol. A new puzzle. *Vanity Fair*, 1879.

[2] M. Dayhoff and R. Eck. Atlas of protein sequences and structures. *National Biomedical Res. Foundation*, 1968.

[3] R. Hamming. Error detecting and error correcting codes. *Bell System Tech*, 1950.

[4] D. Hirschberg. Algorithm for the longest common subsequence problem. *J. ACM.*, 1997.

[5] W. Kent and A. Zahler. Conservation, regulation, synteny and introns in a large-scale c. briggae-c.elegans genome alignment. *Genome Res.*, 2000.

[6] V. Levenstein. Binary code capable of correcting deflections, insertions and reveals. *Soviet Phys. Dokl*, 1966.

[7] S. Needleman and C. Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *J. Mol. Biol.*, 1970.

[8] J. Y. A. Shen, S. Yang and P. Hwang. Super pairwise alignment. *J. Comp. Biol.*, 2002.

[9] W. M. Smith, T.F. and W. Fitch. Comparative biosequence metrics. *J. Mol. Evol.*, 1981.

[10] Y. Zhang. Sequence alignment methods. *Computer Science, University of Minnesota*, 2002.