Computational Biophysics: Algorithms to Applications (CS61060)

Spring 2022-23

Term Project Report

Topic**: Fast multiple similar genetic sequence alignment based on the center star strategy**

Group Members:

Rahul Mandal (20CS30039)

Rohit Ranjan (20CS30066)

Burra Nithish (20CS10018)

Table of contents

Introduction

Sequence alignment is considered the ‘Holy Grail’ problem in computational biology and is of vital importance for molecular function prediction. The widely used databases PFAM (Robert et al., 2010) and RFAM (Gardner et al., 2011) are constructed based on multiple sequence alignment (MSA). Molecular function prediction sometimes depends on evolutionary information (Liu et al., 2014). MSA is also required for evolutionary tree reconstruction. Most of the available phylogenetic tree construction software tools require previously aligned sequences as input. When addressing the evolutionary analysis of bacterial and viral genomes, large-scale similar DNA and RNA sequences often prevent these MSA tools from functioning (Wang et al., 2013). The evolution of viruses is rapid, and massive viral DNA sequences often appear in phylogenetic reconstructions. Therefore, it is necessary to improve the scalable capacity of MSA tools.

Problem Statement

Q). Multiple sequence alignment (MSA) is important work, but bottlenecks arise in the massive MSA of homologous DNA (deoxyribonucleic acid), RNA or genome sequences. Try to implement trie trees to accelerate the center star MSA strategy. The expected time complexity will be decreased to linear time from square time. The algorithm for center star strategy is also discussed below.

Algorithm 1. Improved Centre Star Algorithm Based on Trie Trees

Input: n DNA Sequences, $S_1, S_2, \ldots S_n$
Output: n aligned DNA Sequences $S'_1, S'_2, \ldots, S'_n$
1. **For** each DNA Sequence, S_i
2. Partition S_i into k segments $\{S_{i1}, S_{i2}, \ldots S_{ik}\}$ with equal lengths;
3. Construct trie tree $\mathcal{T}_i$ for the segments set $S_i = \{S_{i1}, S_{i2}, \ldots S_{ik}\}$
4. **for** j from 1 to n, $j \neq i$
5. search $\mathcal{T}_i$ in S_j, and set m_j as the segment appearance times; record all of the appearances in $\mathcal{A}_{ij}$
6. end **for**
7. calculate $m_i = \sum_{j=1, j\neq i}^{n} m_{ij}$
8. end **For**
9. $m^* = argmax_{i=1,2,\ldots,n} m_i$, set S_{m^*} as the centre star sequence
10. **For** each i from 1 to n, $i \neq m^*$
11. Partition S_i and S_{m^*} according $\mathcal{A}_{im^*}$, align the mismatched regions and obtain the pairwise alignment; record all of the positions of inserted spaces in $\mathcal{P}_{im^*}$ and $\mathcal{P}_{m^*i}$.
12. end **For**
13. **For** i from 1 to n, $i \neq m^*$
14. sum $\mathcal{P}_{m^*i}$ to $\mathcal{P}_{m^*}$
15. end **For**
16. obtain the final result, S'_{m^*}, according to $\mathcal{P}_{m^*}$
17. **For** i from 1 to n, $i \neq m^*$
18. compare $\mathcal{P}_{m^*i}$ with $\mathcal{P}_{m^*}$, and update $\mathcal{P}_{im^*}$, then obtain the final result, S'_i
19. end **For**

Existing Work and Literature Review

Zou et.al. - HAlign: Fast multiple similar DNA/RNA sequence alignment based on the centre star strategy

The authors develop two software tools to address the MSA problem. The first employed trie trees to accelerate the centre star MSA strategy. Our problem statement uses this proposed algorithm. The expected time complexity was decreased to linear time from square time. Another major contribution of the paper was incorporating parallelism using Hadoop platform to address large-scale data.

Aho Corasick algorithm for Pattern Searching using Trie Trees

Proposed by Alfred Aho and Margaret Corasick in 1975, the Aho–Corasick algorithm is a string-searching algorithm that utilizes Trie Trees data structure. Let there be a set of strings with the total length m (sum of all lengths). The Aho-Corasick algorithm constructs a trie with some additional links, and then constructs a finite state machine (automaton) in O(m*k) time, where k is the size of the used alphabet. This automaton contains failure links where the longest suffix of current state which happens to be a proper prefix of another word in Trie is utilized. Final time complexity is O (n + m + z), where ‘n’ is the length of the text, ‘m’ is the length of keywords, and ‘z’ is the number of matches in text.

Methodology and Code Flow

1. Sequences in the input are broken into segments of size SEGMENT_LENGTH; and respective Trie trees are constructed with these segments.
2. Each sequence is now searched using the Aho-Corasick algorithm in the trie trees of other sequences and occurrences are used to score the trie tree sequence.
3. We now choose the centre sequence as the sequence with the highest calculated score.
4. Aho-Corasick is again used to match all the sequences with the chosen centre sequence and finding the indices of matched segments. At this stage we have all sequences partially aligned with the centre sequence because we have found matching segments. Between each of these matching segments, the sequence is still unaligned.
5. Call the normal dynamic programming based global alignment solution for these unmatched sequences.
6. Finally insert blanks into centre sequences from each of the instances to globally align them. Output results.

File structure:

- msa_star.py -> Implementation of DP based MSA using centre star strategy for benchmarking.
- msa_star_trie.py -> Implementation of the current problem statement as described above.
- aho_corasick.py -> Implementation of Trie tree and utility algorithms for pattern match
- extseq.py and extdbn.py -> Code for preparing dataset in seq and dbn format for input.

Datasets

The datasets used were the RNA sequences of distinct species like

- Archaea
- Bacteria
- Mitochondria
- Prokaryotes
- Plastids

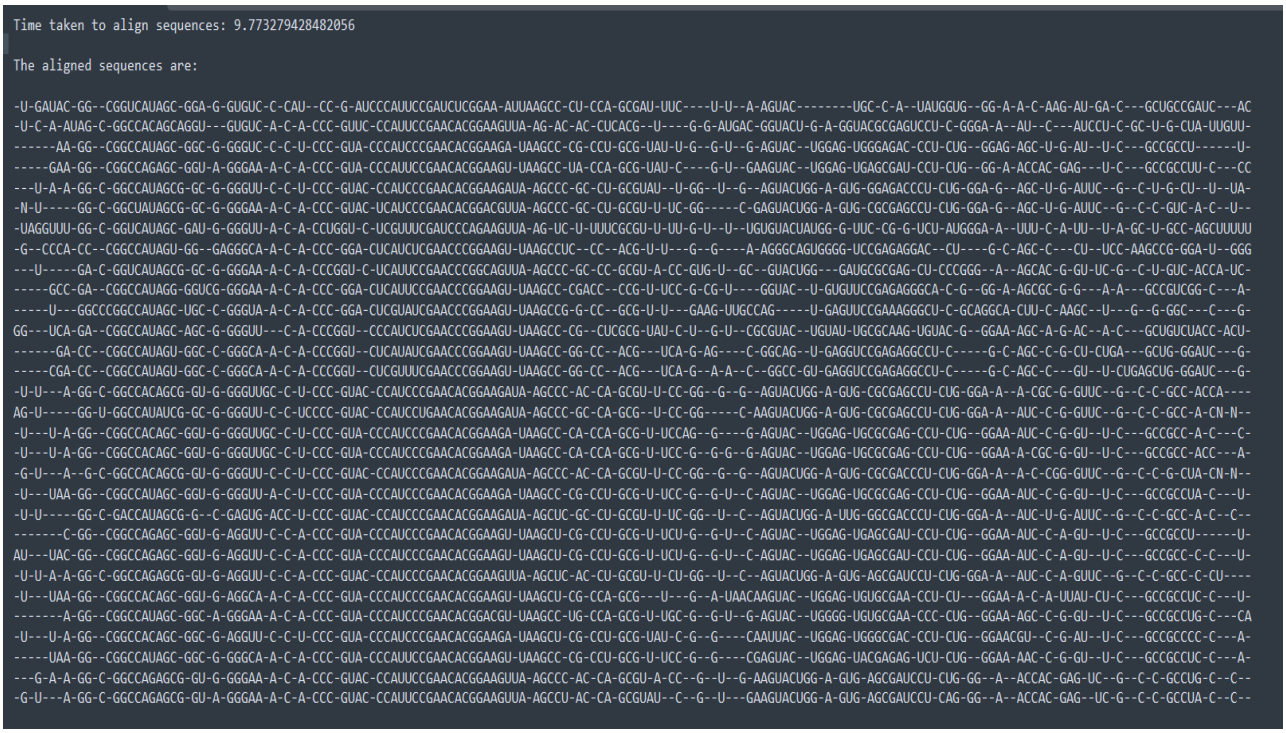
The sequences were stored in the .seq files for each species.

Results for some species are attached below.

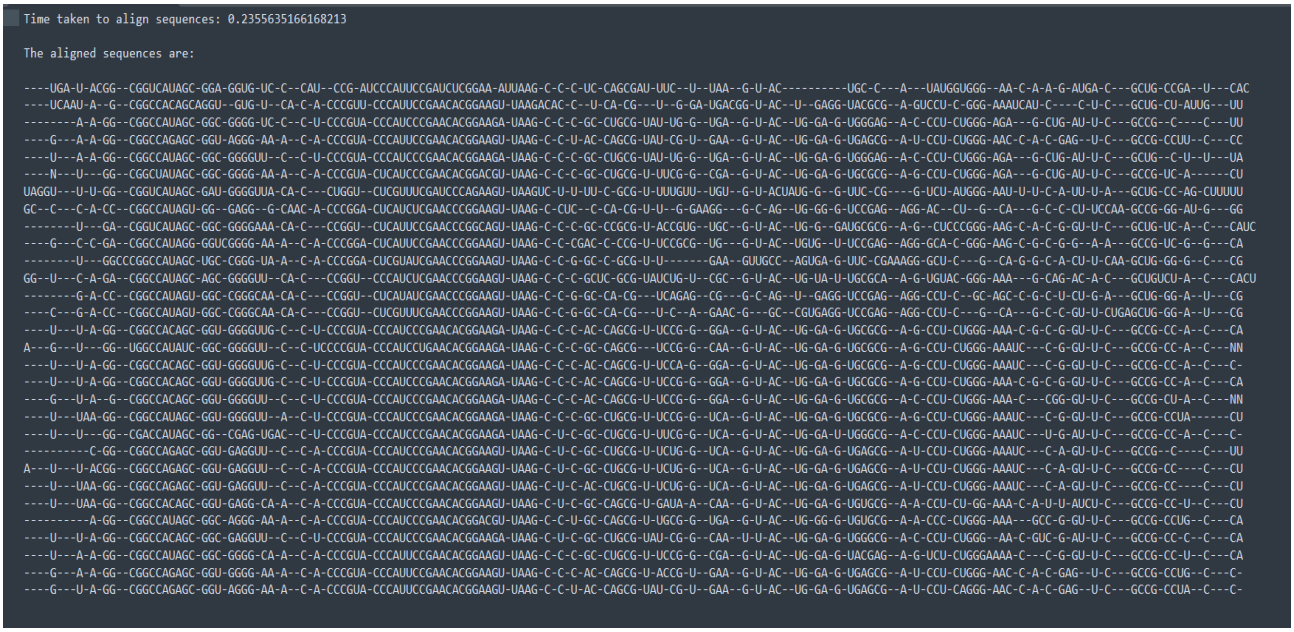
Results

Following are the results of sequence alignment

1. 30 Sequences at a time for Species Archaea with Normal MSA algorithm



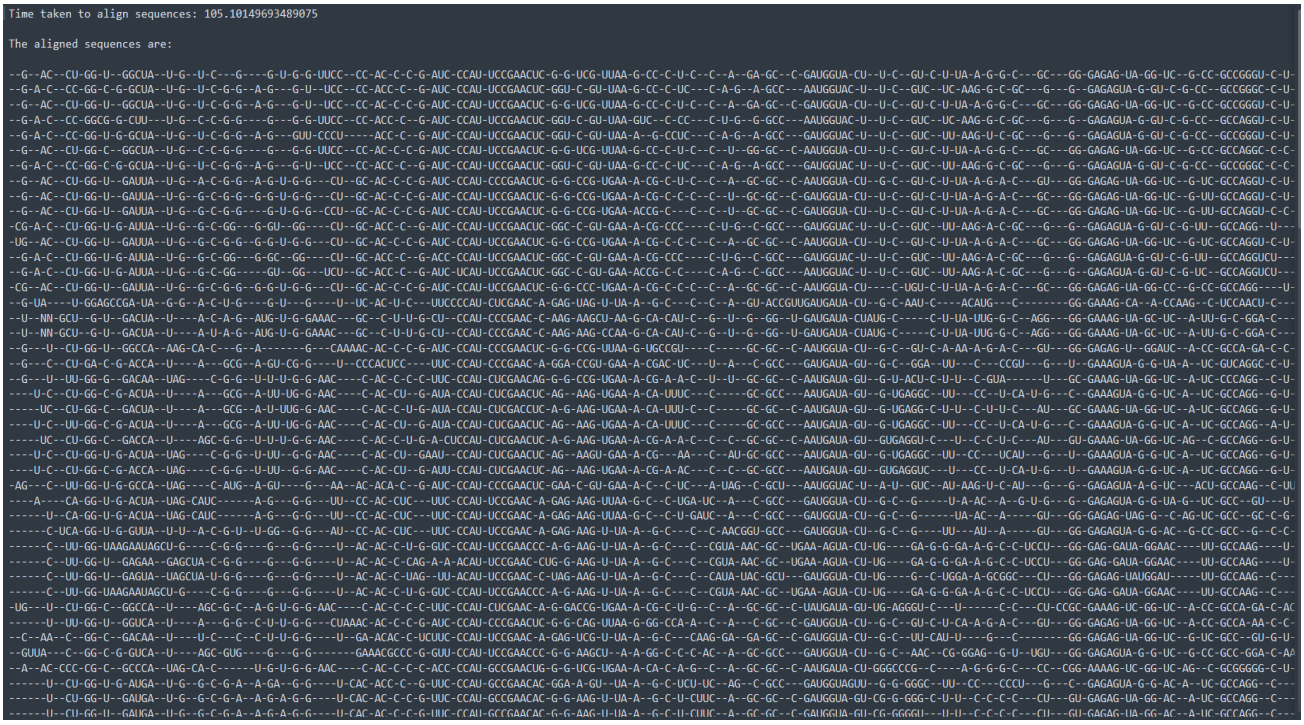
30 Sequences at a time for Species Archaea with MSA using trie tree algorithm



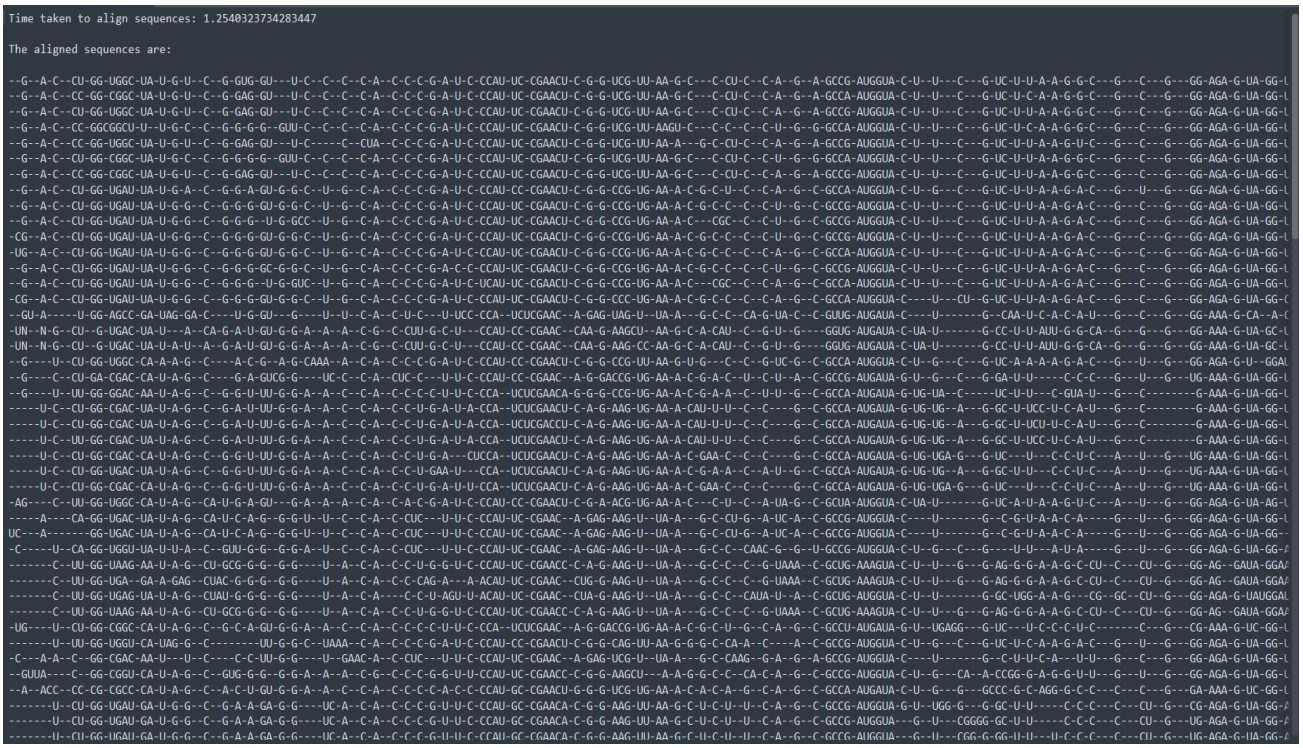
Here for this result set we can see the time difference to align the set of 30 sequences. The time taken by normal MSA is 9.773 s and for aligning the same set of 30 sequences the time taken by the MSA using trie tree is 0.2355.

We can see the improvement in time when MSA with trie trees is used.

2. 100 sequences at a time for species Bacteria using Normal MSA



100 Sequence at a time for species Bacteria with MSA with trie tree algorithm



In this result set we have taken a set of 100 sequences of species Bacteria and the time taken by normal MSA for aligning these 100 sequences is 105.101 s (about little less than 2 minutes) and the time taken by the MSA with trie tree is only 1.2540 s which is significantly smaller than the former.

Further results on all five species datasets with different segment lengths are available in the submitted repository.

Further work and Conclusion

Scalable and fast MSA is the need of the hour. While the HAlign algorithm decreases the time complexity to linear from squared time, the quality of results may vary because the algorithm uses some heuristics. Once we get the first level of matches on running Aho-Corasick with the centre sequence and any one sequence at a time, we get the indices of these matches of segments. However, the optimal arrangement of these segment matches is again a DP problem. Here we achieve fast times by using a greedy FIFO allotment which might not be optimal always. Practically however, we find the results to be suitable.

Further work should further focus on increasing the stability of the MSA algorithms in the linear time complexity space. Our experiments showed decreasing match quality as the number of sequences input was increased. In conclusion, we found the algorithm to be a great steppingstone towards linear time MSA algorithms. Working directly with code and datasets has given us a very clear image of the computations involved in sequence alignment and its several bottlenecks.