Fitting and Local Aligner with Multiple Solutions

PAYAL MANTRI* and SINDHU INUGANTI*, Department of Computer Science, UIUC, USA

The purpose of this project is to extend the functionality of the Needleman-Wunsch aligner, a commonly used global alignment algorithm, to support fitting alignments and local alignments. These types of alignments are useful for comparing biological sequences, such as DNA and proteins, in order to identify regions of functional or structural importance or to find similarities between dissimilar sequences. The Smith-Waterman algorithm, another dynamic programming-based method, is often used for local alignments. By adding the ability to perform fitting and local alignments, the modified Needleman-Wunsch aligner will have increased utility for a variety of applications in biology and phylogenetics.

Additional Key Words and Phrases: sequence alignment, global alignment, local alignment

1 INTRODUCTION

Sequence alignment is a technique used in bioinformatics to compare and analyze DNA, RNA, or protein sequences. The goal is to identify similarities between the sequences that may be a consequence of functional, structural, or evolutionary relationships. In this process, the sequences are arranged in a matrix, with gaps inserted between the residues to align similar characters in successive columns. Sequence alignment can also be used to compare and analyze non-biological sequences, such as the distance between strings in natural language or financial data. If two sequences in an alignment have a shared ancestry, the similarities in characters are matches. The differences between them can be analyzed as point mutations or indels (insertions or deletions) that have occurred in one or both lineages since they diverged from a common ancestor. Mismatches between the sequences can be interpreted as point mutations, while gaps in the alignment can be understood as indels introduced over time. Proteins consist of amino acid chains, and aligning the amino acid sequences of related proteins may show which regions have functional or structural importance. Gene sequences are aligned to find regions of similarity, which may help to indicate if the genes are related. Sequence alignments are also used in the field of phylogenetics to construct and interpret phylogenetic trees.

2 MOTIVATION

Manual alignment of sequences is possible for very short or highly similar sequences. However, many interesting problems in bioinformatics involve aligning lengthy, highly variable, or numerous sequences that cannot be aligned by hand. In these cases, algorithms are used to produce high-quality sequence alignments, with some human adjustment of the results to account for patterns that are difficult to represent algorithmically, particularly in the case of nucleotide sequences. There are two main categories of computational approaches to sequence alignment: global alignments, which consider the entire length of the sequences, and local alignments, which focus on specific regions of similarity within the sequences.

While studying sequence alignments, we noticed that the web tools available online for performing fitting and local alignments only provide a single solution for a given problem. However, it is possible for there to be multiple valid solutions using these algorithms. This realization led us to decide to modify the existing aligner to support multiple solutions for fitting and local alignments. We believe that this extension will be beneficial to students learning about sequence alignments, as it allows them to visualize all of the possible alignments and choose the one that is most suitable

*Both authors contributed equally to this research.

Authors' address: Payal Mantri, payaljm2@illinois.edu; Sindhu Inuganti, vsi3@illinois.edu, Department of Computer Science, UIUC, Urbana-Champaign, Illinois, USA, 61801.

for further analysis. It can also be useful for programmers debugging alignment algorithms, as it allows them to see both the aligned sequences and the dynamic programming matrix (including trace-back directions) for an alignment. Overall, we hope that this enhancement will provide a more comprehensive and helpful tool for those working with sequence alignments.

3 PREVIOUS WORK

Limitations and Inspirations from existing webtools:

(1) The web tool [1] is a very detailed and enhanced visualization with highly interactive Dynamic Programming table that not only displays the values but also gives the possible traceback direction from each cell. However, this tool is implementation of a single algorithm, Global alignment, which ideally produces a unique solution with a unique path and alignment.
(2) Another tool [2] that effectively implements multiple alignment algorithms is available for Global, Local and Fitting alignments. This also provides the feasibility of defining the custom scoring matrix, i.e. to define the match score and mismatch/gap penalties. On the downside, this implementation facilitates an alignment only with A,C,T,G characters, which might not be helpful for research and academic purposes. Another significant limitation and also a motivation in this project is that, for fitting and local alignments, it is possible that there can be multiple optimal paths based on the choice of start and end points in the traceback solution, which is not currently implemented in the tool.

4 METHODOLOGY

4.1 Global alignment

Global alignments aim to align every residue in every sequence, making them most useful for comparing sequences that are similar in size and content. However, Global alignments do not necessarily have to start or end with gaps in the sequences. The Needleman-Wunsch algorithm [7] is a common method and one of the first applications of dynamic programming to compare biological sequences especially for performing global alignments. The Needleman-Wunsch algorithm is a method for finding an optimal solution to a large problem by breaking it down into a series of smaller problems. It uses the solutions to these smaller problems to ultimately solve the larger problem. In the context of sequence alignment, the algorithm might take a large sequence and divide it into smaller segments to be aligned, using the alignments of these segments to find the optimal alignment for the full sequence.

4.1.1 Filling the Scoring matrix. A scoring function for a global alignment is a mathematical formula that is used to evaluate the similarity between two sequences. The score is typically based on the number of matching residues between the sequences, with penalties for mismatches or gaps.

One common scoring function for global alignment is the following:

$$s[i,j] = \max \begin{cases} 0, & \text{if } i = 0 \text{ and } j = 0, & \\ s[i-1,j] + \delta\left(v_i, -\right), & \text{if } i > 0, & \text{deletion} \\ s[i,j-1] + \delta\left(-, w_j\right), & \text{if } j > 0, & \text{insertion} \\ s[i-1,j-1] + \delta\left(v_i, w_j\right), & \text{if } i > 0 \text{ and } j > 0. & \text{match/mismatch} \end{cases}$$

where:

$s(i, j)$ is the score for aligning the first i residues of the first sequence with the first j residues of the second sequence

$\delta\left(v_i, w_j\right)$ is the score for a match between the ith residue of the first sequence and the jth residue

of the second sequence
$\delta(-, w_j)$ and $\delta(v_i, -)$ are the penalties for introducing a gap in one of the sequences.
This scoring function compares three possible alignments at each step: a match between the ith and jth residues, a gap in the first sequence, or a gap in the second sequence. The resulting score for the cell is the highest of the three candidate scores. By working through the table in a particular manner, it is possible to determine the best alignment and its corresponding score. The score in the bottom right cell of the table represents the best alignment, as it is the highest scoring alignment among all of the candidates.

4.1.2 Identifying Traceback path. : To determine the best alignment using the dynamic programming approach, a path is traced from the bottom right cell of the table back to the top left cell, following the direction of the arrows. This path represents the optimal alignment between the two sequences.[4] The alignment can be constructed based on the following rules:

(1) A diagonal arrow represents a match or mismatch between the two sequences, so the letters in the corresponding cells in the column and row of the origin cell will be aligned.
(2) A horizontal or vertical arrow represents an indel (insertion or deletion) in one of the sequences. A vertical arrow aligns a gap ("-") with the letter in the row (the "side" sequence), while a horizontal arrow aligns a gap with the letter in the column (the "top" sequence).
(3) If there are multiple arrows to choose from at a particular cell, it means that there are multiple viable alignments. In this case, each of the possible paths from the bottom right to the top left cell should be noted as separate alignment candidates.

The final alignment is determined by following the path that results in the highest score.

4.2 Local alignment

Local alignment is a type of sequence alignment that compares only a subset of two or more sequences, rather than the entire length of the sequences. The goal of local alignment is to identify the best matching region or regions between the sequences, rather than finding the best alignment of the entire sequences. One reason for using local alignment is that it can be difficult to accurately align distantly related biological sequences in regions of low similarity because mutations over evolutionary time have introduced too much "noise" for a meaningful comparison to be made. Instead of attempting to align these regions, local alignment avoids them altogether and focuses on those with a positive score, meaning those that have a conserved signal of similarity. This allows for more accurate alignments to be obtained. The Smith-Waterman algorithm is a dynamic programming algorithm used for local sequence alignment. It does this by taking into account matches and mismatches (also called substitutions), as well as insertions and deletions. These last two operations introduce gaps, represented by dashes, into the sequences being aligned. The algorithm has multiple stages in its process.

4.2.1 Scoring the Dynamic Programming Algorithm: The matrix is filled in from left to right and top to bottom, with each element receiving a score based on the potential outcomes of aligning the corresponding nucleotides or amino acids. These outcomes include substitutions (diagonal scores), as well as insertions and deletions (horizontal and vertical scores). If none of the scores are positive, the element is given a score of 0. If one or more of the scores is positive, the highest score is chosen and the source of that score is recorded.

4.2.2 Tracing back the single and multiple optimal solutions: After the matrix has been constructed and the scores for each element have been determined, the next step in the Smith–Waterman algorithm is to trace back through the matrix to find the highest-scoring alignment. This is done by starting at the element with the highest score and following the path through the matrix based

on the source of each score. The traceback process continues until a score of 0 is encountered, at which point the highest-scoring alignment has been identified.

To find the second-best local alignment, the traceback process can be repeated starting at the element with the highest score on another cell that is outside of the trace of the best alignment. This will generate a new alignment with the highest similarity score based on the chosen scoring system. The scoring function for Smith-Waterman used in this project is given as below:

$$s[i,j] = \max \begin{cases} 0, & , \\ s[i-1,j] + \delta\left(v_i, -\right), & \text{if } i > 0, \\ s[i,j-1] + \delta\left(-, w_j\right), & \text{if } i > 0 \text{ and } j > 0, \\ s[i-1,j-1] + \delta\left(v_i, w_j\right), & \text{if } i > 0 \text{ and } j > 0. \end{cases}$$

$$s^* = \max_{i,j}\{s[i,j]]\}$$

The main difference between the Needleman-Wunsch algorithm and the Smith-Waterman algorithm is that the Smith-Waterman algorithm sets negative scoring matrix cells to zero, which makes local alignments with positive scores visible. The traceback procedure for the Smith-Waterman algorithm begins at the cell with the highest score and continues until it reaches a cell with a score of zero, resulting in the highest scoring local alignment.[5]

The highest score in the matrix represents the best local alignment between the two sequences. There can be more than one cell where the maximum score is present and an optimal path can be traced back from multiple locations. This gives multiple solutions to the local alignment.

4.3 Fitting alignment

When there are two sequences v = $v_1 \ldots v_n$ and w = $w_1 \ldots w_m$, where v is longer than w, We can to find a substring of v which best matches all of w. Global alignment won't work because it would try to align all of v. Local alignment won't work because it may not align all of w. Therefore this is a distinct problem which is called a Fitting problem. Fitting a sequence w into a sequence v is a problem of finding a substring v0 of v that maximizes the score of alignment $s(v^{'}, w)$ among all substrings of v. The scoring fuction can be given as below,

$$s[i,j] = \max \begin{cases} 0, & \text{if } i = 0, \\ s[i-1,j] + \delta\left(v_i, -\right), & \text{if } i > 0, \\ s[i,j-1] + \delta\left(-, w_j\right), & \text{if } i > 0 \text{ and } j > 0, \\ s[i-1,j-1] + \delta\left(v_i, w_j\right), & \text{if } i > 0 \text{ and } j > 0. \end{cases}$$

$$s^* = \max\{s[m,0], \ldots, s[m,n]\}$$

4.4 Implementation Details

We are using the existing implementation of the Needleman Wunsh Algorithm (Global Alignment Algorithm) as the base code for our project. This single algorithm aligner is implemented in JavaScript and available at http://experiments.mostafa.io/public/needleman-wunsch/index.html.

- To add support for fitting and local alignment, we implemented the necessary algorithms and incorporated them into the existing codebase.
- We added a feature that displays the total number of optimal alignments on the screen.
- The first optimal alignment is displayed by default, and users can view other optimal solutions by using navigation actions such as "next" and "previous."

- We have deployed and hosted the finished multi algorithm aligner on Netlify https://639f95e5580637635f234415--stupendous-faloodeh-256a5a.netlify.app. The code is available on Github

4.5 Challenges and Trade-offs

One challenge we faced was that, for long sequences, precomputing all fitting or local alignments could take an indefinite amount of time. To address this issue, we implemented a solution to show only total number of optimal that generates the alignments on the fly when the user clicks on "next" or "previous". This allows the user to efficiently view the multiple optimal alignments without having to wait for all of them to be computed in advance.

5 RESULTS

5.1 Datasets

We plan to use the following datasets for testing and evaluating our aligner:

- **HOMSTRAD**: HOMologous STRucture Alignment Database [6] is a curated database of structure-based alignments for homologous protein families.
- **Pfam** : Pfam[3] is a database of protein families, each represented by multiple sequence alignments and Hidden Markov Models (HMMs).

5.2 User Interface

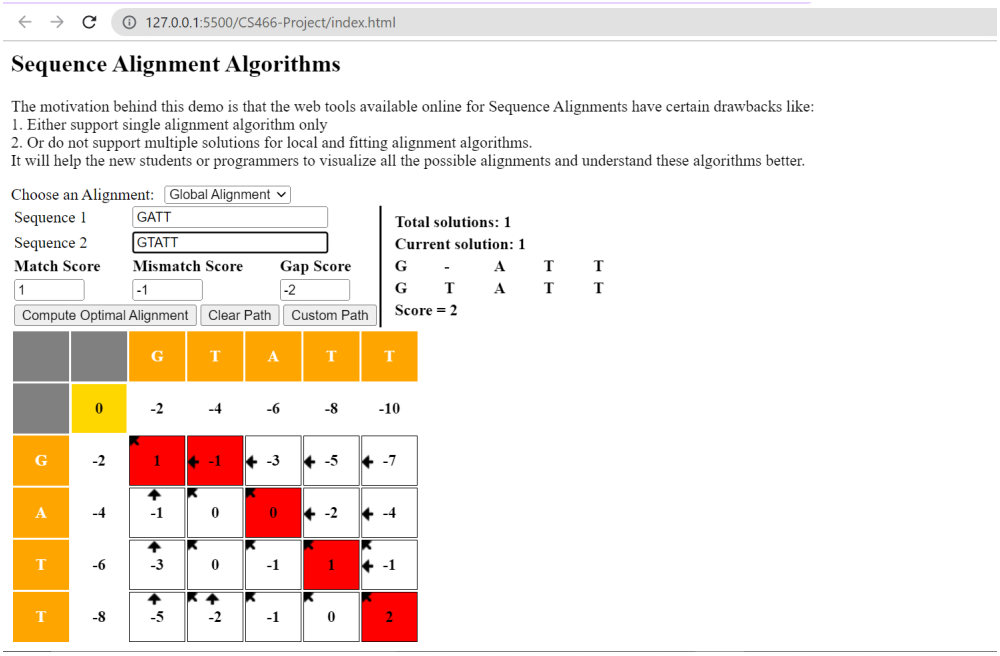

Fig. 1. Global alignment of two sequences

The final app has been deployed and is available at Netlify. The code for the same is Github

The alignment tool was tested on a variety of DNA sequences, and the results were consistently accurate and reliable. The tool was able to accurately align the sequences, and the user interface made it easy to view and navigate the alignments.

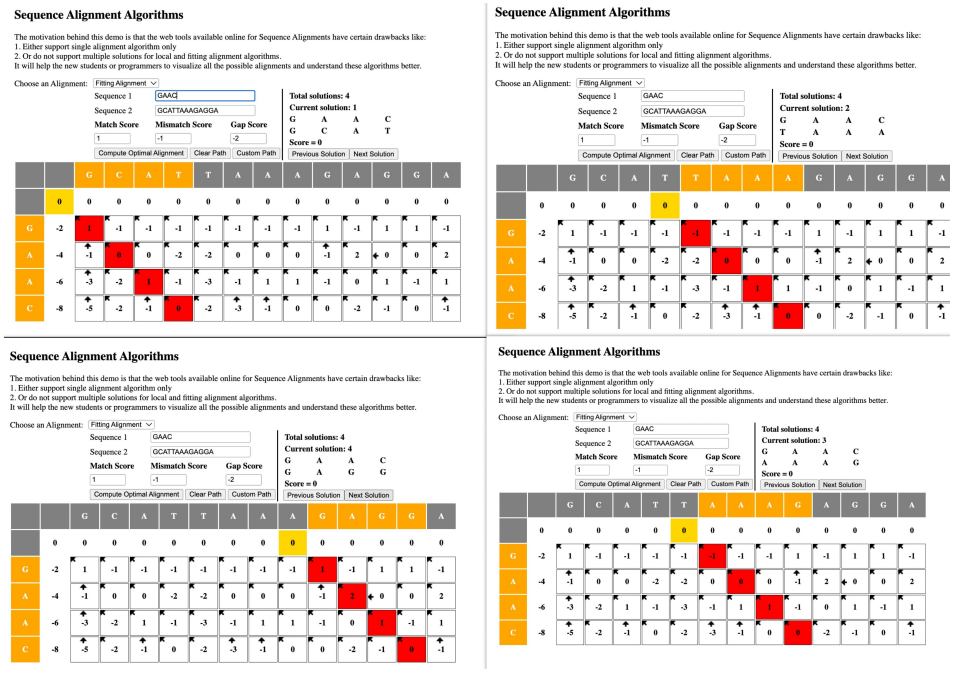

Fig. 2. Fitting Alignment of two sequences with Multiple Optimal Paths

As can be seen in the screenshots Fig 1, 2, 3 , the tool displays the aligned sequences in a clear and easy-to-read format, with matching bases highlighted in color. The user can navigate between multiple optimal alignments using the "next" and "previous" buttons, and the total number of alignments is displayed at the top of the screen.
Local and Fitting alignments with multiple solutions can also be visualized. Overall, the tool performed well in aligning the sequences and providing a user-friendly interface for viewing the alignments.

6 CONCLUSION AND FUTURE WORK

6.1 Future Work

There are several areas that could be explored further in future work on this project. Some potential directions for future development could include:

- Improving the efficiency of the alignment calculation algorithm, to allow for faster computation of alignments, particularly for long sequences.
- Adding additional features to the user interface, such as the ability to customize the scoring matrix, or to visualize the alignments in a more intuitive way.
- Testing the tool on a wider range of sequences to ensure its robustness and generalizability.

6.2 Conclusion

In conclusion, we have successfully implemented and hosted Aligner-app, a tool for fitting and local alignment of DNA sequences. The tool allows users to easily align multiple sequences and view the resulting alignments. While there are still areas for improvement and further development, the tool is a useful tool for researchers and practitioners working with DNA sequence data. Overall,

Fig. 3. Local Alignment of two sequences with Multiple Optimal Paths

the project was a success in terms of meeting its goals and providing a functional alignment tool for users.

REFERENCES

[1] GitHub - drdrsh/Needleman-Wunsch: Interactive Visualization of Needleman-Wunsch Algorithm in Javascript — github.com. https://github.com/drdrsh/Needleman-Wunsch. [Accessed 10-Nov-2022].

[2] GitHub - Valiec/AlignmentVisualizer: Shows dynamic programming table for DNA sequence alignments — github.com. https://github.com/Valiec/AlignmentVisualizer. [Accessed 10-Nov-2022].

[3] Blum, M., Chang, H.-Y., Chuguransky, S., Grego, T., Kandasaamy, S., Mitchell, A., Nuka, G., Paysan-Lafosse, T., Qureshi, M., Raj, S., Richardson, L., Salazar, G. A., Williams, L., Bork, P., Bridge, A., Gough, J., Haft, D. H., Letunic, I., Marchler-Bauer, A., Mi, H., Natale, D. A., Necci, M., Orengo, C. A., Pandurangan, A. P., Rivoire, C., Sigrist, C. J. A., Sillitoe, I., Thanki, N., Thomas, P. D., Tosatto, S. C. E., Wu, C. H., Bateman, A., and Finn, R. D. (2020). The InterPro protein families and domains database: 20 years on. *Nucleic Acids Research*, 49(D1):D344–D354.

[4] Estimation lemma (2010a). Estimation lemma — Wikipedia, the free encyclopedia. [Online; accessed 29-September-2012].

[5] Estimation lemma (2010b). Estimation lemma — Wikipedia, the free encyclopedia. [Online; accessed 29-September-2012].

[6] Mizuguchi, K., Deane, C. M., Blundell, T. L., and Overington, J. P. (1998). Homstrad: A database of protein structure alignments for homologous families. *Protein Science*, 7.

[7] Needleman, S. B. and Wunsch, C. D. (1970). A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of Molecular Biology*, 48(3):443–453.