# Implementation of Multiple Sequence Alignment using Genetic Algorithm


***Abstract***

Multiple Sequence Alignment (MSA) is a problem of alignment of three or more sequences. The aligning of multiple sequences has a lot of applications like construction of phylogenetic tree, prediction of protein structure and is one of the fundamental problems in Bioinformatics. Dynamic Programming is the most accurate way to align multiple sequences. However, this approach is impractical since it is computationally very expensive. A solution that is currently being considered is to use a heuristic approach like the Genetic Algorithm to solve the problem. The Genetic Algorithm is a way to solve optimization problems using techniques that mimic biological phenomenon namely selection, crossover, and mutation. In this project, we plan to collect multiple protein sequences from BaliBase benchmark database and attempt to align them using the Genetic Algorithm.


***Introduction***

Genetic algorithm belongs to the class of evolutionary algorithms. The steps of Genetic Algorithm are creating the initial population of sequences, performing crossover on them at middle position, applying mutation i.e. applying a minute percentage of insertion/deletion in the element of the sequence and finally selecting the best alignments based on a predefined score (sum of pairs) to get a better set of sequences than the initial ones. The steps are repeated definite number of times and the best score out of them is selected to get the better alignment (may not be the best alignment) of the multiple sequences. The score output from the genetic algorithm is then compared with the output of CLUSTALX 2.1 tool to check how the heuristic approach to this problem is faring against the standard tool available.

***Workflow***

The input sequences are taken from available data sets in BaliBase 3 (http://www.lbgi.fr/balibase/). Implementation steps of the Genetic Algorithm for aligning multiple protein sequences are discussed. They are broadly classified as CREATING_INITIAL_POPULATION, CROSSOVER, MUTATION, and SELECTION.

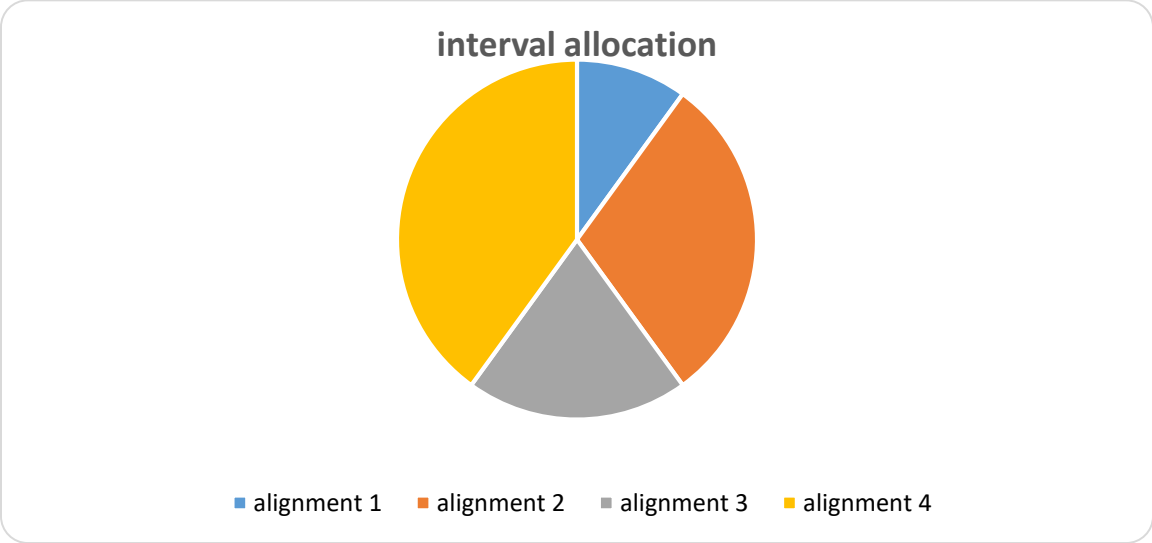
CREATING_INITIAL_POPULATION

1. Three sequences are taken as input in the code, which can be of variable lengths. If the lengths of the input sequences are not equal, then gaps are introduced at the end of the shorter sequences to make the length of the sequences equal.
2. Gap Introduction: Four iterations are run, each time with a different gap percentage. (0.2, 0.3, 0.4 and 0.5 times the length of the initial sequences).
3. The introduced gaps at random locations using the random function can lead to a large number of alignment combinations possible. Thousand alignments are selected at random and thus initial population of 1000 alignments is created.

CROSSOVER

1. In crossover, total score of all alignments (TScore) was calculated and each alignment was assigned a value equal to their respective Score/TScore value.
2. Every alignment was assigned with contiguous weighted intervals corresponding to their score value/TScore.

3. 2000 random numbers are generated and they are sorted. Each sorted random value also holds the index at which it was originally generated.
4. Each sorted random value is compared with the weighted intervals of the alignments to see which alignment it corresponds to. This 'parent' alignment is then linked with the index of the original random value.
   This is analogous to a biased roulette wheel where each alignment can be considered as a separate sector of the wheel and the area of the sector is directly proportional to the alignment score. So, when the wheel is spun, the ball has a higher probability of landing in the sector with greater area and thus the alignment with larger alignment score has higher probability of getting picked since its interval is larger.
   For example, if we have 4 alignments with score of 10,30,20,40 respectively. Then the interval allocation for roulette wheel will be:




   Now we spin the ball. So, the probability of ball landing in a slot(alignment) is proportional to the score of that alignment. Using this method, the complexity of the process is improved from $O(n^2)$ to $O(n*\log(n))$.
5. The original random values are taken sequentially and for every two random values, their corresponding linked parents are taken as the candidates for the crossover step.
6. Both the candidate parent alignments are split into half and they are matched with each other such that the first half of the first parent is combined with the second half of the second parent and vice versa which results in creation of two children.
   The gaps are handled such that when the split is made, the number of characters on both sides of the split are equal. This ensures that no overlap or separation between both halves exist.
7. Both the new child alignments are compared with each other based on their scores and the child with greater score is kept while the other is discarded.

MUTATION

1. 7% mutation probability rate is chosen as it was found to generate better results.
2. When P(0.07) = TRUE, and the hit position has a gap, the gap is deleted.
3. When P(0.07) = TRUE, and an amino acid is present in the position, a gap is introduced just after the amino acid.

4. For each alignment, the length of all the sequences of that alignment are adjusted and made equal to the longest one by introducing gaps at the end of each sequence of that alignment.

SELECTION

1. The 2000 alignments are sorted in decreasing order of their fitness score.
2. Out of the 2000 alignments, selection shortlists 1000 alignments using step 3 and 4 below to get the input alignments for the next iteration.
3. 20% of the total current population (PARENT and CHILD) having the highest fitness score measured using sum of pairs are selected directly for the next iteration.
4. From the remaining 80% of the population, 600 alignments are selected using probability function with the alignments having higher score having more probability of getting selected than with lower probability. This is achieved using probability function on the weights of each alignment based on its score in the same manner as that in crossover step.
5. If there are gaps in all the alignments for a single column in the alignment, that column is removed.

After completion of the selection process, a total of 1000 alignments are generated with the fitness scores better than the initial population. The above four function are iterated 100 times and the best score is taken as the final alignment for Genetic Algorithm to be compared with the CLUSTALX 2.1 tool for the same set of sequences.

***Results and Discussion***

The team worked on 100 protein alignments taken from the BaliBase database and ran them on CLUSTALX 2.1 and on the Genetic Algorithm program to get the SOP scores. The percentage accuracy over the CLUSTALX 2.1 is measured. It is found that for some input sequences, the genetic algorithm gives better scores than the CLUSTALX 2.1 tool. Below are the details of the fitness scores and the percentage accuracy for 12 input sequences taken from the BaliBase database.

| # | BB Sequence Index | CLUSTALX 2.1 SOP Score | Genetic Algorithm SOP Score | Percentage Accuracy |
|---|---|---|---|---|
| 1 | BB11001 (1aab_ 1j46_A 1k99_A) | 50 | 51 | 102.00 |
| 2 | BB11002 (1abo_A 1pht_ 1ihv_A) | 29 | 24 | 82.75 |
| 3 | BB11013 (1idy_ 1hst_A 1tc3_C) | 24 | 23 | 95.83 |

| | | | | |
|---|---|---|---|---|
| 4 | BB11022 (1r69_ 1neq_ 1au7_A) | 34 | 31 | 91.17 |
| 5 | BB11025 (1tvx_A 1prt_F 1sap_) | 26 | 26 | 100.00 |
| 6 | BB12009 (1csp CSPD_HAEIN CSPF_ECOLI) | 70 | 56 | 80.00 |
| 7 | BB30025 (GLRX_HAEIN GLR3_ECOLI 1grx_) | 114 | 104 | 91.2 |
| 8 | BB20036 (THIO_RHOSH THIO_PASMU THIO_HAEIN) | 196 | 172 | 87.8 |
| 9 | BB12041 (1wad_ 1czj_ 1gm4_A) | 65 | 61 | 92.4 |
| 10 | BB11017 (1qi7_A 1dm0_A 111n_A) | 108 | 102 | 94.44 |
| 11 | BB11029 (1ycc_ 1cno_A 1a56_) | 46 | 40 | 86.95 |
| 12 | BB11021 (1aac_ 1pmy_ 1kdi_) | 52 | 49 | 94.23 |

***Workload Distribution***

Nishant Varma:

1. Study of Genetic Algorithm and its steps.
2. Implemented the CREATING_INITIAL_POPULATION function of the genetic algorithm.

3. Collected 25 alignments from BaliBase database and compared each alignment with CLUSTALX and Genetic Algorithm implemented in the project.

Adhiraj Nakhe:

1. Study of Genetic Algorithm and its steps.
2. Worked on the CROSSOVER function of the genetic algorithm.
3. Collected 25 alignments from BaliBase database and compared each alignment with CLUSTALX and Genetic Algorithm implemented in the project.

Debarshi Mitra:

1. Study of Genetic Algorithm and its steps.
2. Worked on the MUTATION function of the genetic algorithm.
3. Collected 25 alignments from BaliBase database and compared each alignment with CLUSTALX and Genetic Algorithm implemented in the project.

Vivek Goyal:

1. Study of Genetic Algorithm and its steps.
2. Worked on the SELECTION function of the genetic algorithm.
3. Collected 25 alignments from BaliBase database and compared each alignment with CLUSTALX and Genetic Algorithm implemented in the project.

After implementing the functions using Java, the team worked together to integrate the code, debugged all errors, and tested the code for multiple protein sequences to get the desired output alignment and scores.

## ***Conclusion***

Alignment of multiple protein sequences taken from the BaliBase database is successfully implemented using genetic algorithm. The results show more than 85% accuracy for all the sequences tested when compared to CLUSTALX 2.1 tool output. In a few cases, the genetic algorithm even gives better score than the CLUSTALX 2.1. The algorithm implemented leaves scope for improvisation in future to make the algorithm more dynamic. The number of sequences can be taken as a variable and not restricted to a fixed size as with the algorithm above.

## ***References***


- http://www.tcoffee.org/Publications/Ps_pdf/saga_paper.pdf
- https://deepblue.lib.umich.edu/bitstream/handle/2027.42/46947/10994_2005_Article_422926.pdf
- SAGA: Sequence Alignment by Genetic Algorithm, C. Notredame, D.G. Higgins, Nucleic Acid Research, Vol. 24, 1515-1524, 1996.
- https://www.researchgate.net/publication/220885709_A_Standard_GA_Approach_to_Native_Protein_Conformation_Prediction.
- "IEEE Xplore document - progressive alignment method using genetic algorithm for multiple sequence alignment," 2016. [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=6151111.
- "IEEE Xplore document - multiple sequence alignment using genetic algorithm and simulated annealing," 2016. [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=1307828.
- http://www.ch.embnet.org/software/ClustalW.html