Pairwise Sequence Alignment

Antonio Almeida, UP2015058365

April 14th, 2019

1 Introduction

This report refers to the the second project of the Algorithms for Bioinformatics[1] course, with goal of implementing common algorithms for pairwise sequence alignment, namely the Needleman-Wunsch[2] and Smith-Waterman[3] algorithms. More specifically, the approach taken was to extend a previously developed Python package designed for a practical way of manipulating biological sequences, i.e., `bioseq`.

2 Implementation

The main concern taken while developing the project was the accuracy of the algorithms implementation, given the major set of edge cases inherent to them. To tackle this, a Test Driven Development[4] approach was taken, namely by starting with small, familiar sequences whose expected output was know, and then iteratively increasing the testees' sizes as the implementation became more robust. This not only improved the final quality of the implementation, but also helped tackle the unfamiliarity of the algorithms to the developer.

All the features added are part of a new class, `Alignment`, whose purpose is encapsulating features related to pairwise sequence alignment. The functions can be used as part of the `bioseq` package, by using its abstractions or biological sequences, or independently, by using standard Python strings as sequences. This was made possible by well defined implementation of class methods overloading[5].

A significant effort of the development went to structuring and refactoring the code developed during the classes to fit production software standards and adapt it to the already existing package.

Additionally, given the pedagogical nature of the project, and even though traditional visualization is not feasible for observing common "real-life" sequence alignments, utilities for visualizing the algorithms' steps through their phases were developed through a `Debug` flag on most features.

Finally, it was decided not to implement additional features developed in class as their addition to the package is trivial and didn't match the project's goal.

3 Results

All the required features were implemented.

Major features:

[1]https://sigarra.up.pt/fcup/en/ucurr_geral.ficha_uc_view?pv_ocorrencia_id=430165
[2]https://en.wikipedia.org/wiki/Needleman%E2%80%93Wunsch_algorithm
[3]https://en.wikipedia.org/wiki/Smith%E2%80%93Waterman_algorithm
[4]https://en.wikipedia.org/wiki/Test-driven_development
[5]https://docs.python.org/3/reference/datamodel.html

- Needleman-Wunsch algorithm for global sequence alignment - `global_align_multiple_solutions` and `recover_global_align_multiple_solutions`
- Smith-Waterman algorithm for local sequence alignment `local_align_multiple_solutions` and `recover_local_align_multiple_solutions`
- Both implemented algorithms calculate multiple optimal solutions
- Visualization of alignment scores between pairs in set of sequences for both global and local alignment `compare_pairwise_global_align` and `compare_pairwise_local_align`
- Helper function to run both build and recover stages of the sequence alignment algorithms - `run_global_align_mul` and `run_local_align_multiple_solutions`

Notable features:

- A `Debug` flag that helps visualize the algorithms' steps, e.g., score and trace matrixes
- Extended the package's test suite[6] to include tests for the additional functionalities
- The additional features work both with the biological sequences class from the package, i.e., `BioSeq`, and Python strings, while using the same code
- Ability to use substution matrixes based on file or through a user selected constant gap, match and mismatch values
- The recovery algorithm on both global and local versions of sequence aligment is not recursive and is pretty scalable

4 Conclusions

The main goal of the project was successfully accomplished as the author has learned the objective, methodology, and a few algorithms for Pairwise Sequence Alignment, an area of Bioinformatics previously completely unknown.

From a software point of view, the author concludes the project fits its purpose, its extensible enough to cover additional sequence alignment algorithms or general Bioinformatics features, but requires a few improvements in order to be used as a serious tool. Most improvements are trivial are satisfied by "doing things" in a more Pythonic[7] way, but greatly improve the usability and readibility of the code.

5 References

No papers were referenced. However, footnotes have been added with hyperlinks for crucial keywords

[6]https://en.wikipedia.org/wiki/Test_suite
[7]http://google.github.io/styleguide/pyguide.html