Programming Project 03 - Likelihood Protein Identification via k-mer Genetic Sequence Assembly

Group Projects

These projects serve as a way for you to learn how to build a programmatic tool in a group setting. Often in this field, large packages and software are built in a collaborative effort, and knowing how to effectively communicate ideas, tasks, and workflow is an essential skill. Here, you will work as part of a group and demonstrate your ability to compose a cohesive (and working) tool comprised of components created by others *and* yourself.

Deadline

February 07, 2022 - 12:00/noon (UTC+1:00)

Submission Guidelines

1. Clone your repository to your local workstation/computer if you have not done so
2. Structure and work on your submission.
3. Commit your work to the git repository.
4. Create a git tag.
    - Tag name must be equivalent to ”GroupProject”.
    - Tag can be made via the command line or using the GitLab GUI
5. Be sure to include a PDF of your presentation in your repository once it is finished

Package Requirements

- All code comprising the backend portion (i.e. the code responsible for downloading, parsing, and formatting the information) must be compiled as an installable Python package
- The package must contain the following:
    - A working CLI (see CLI Requirements below)
    - A clear and descriptive `README.md` that details what the package is, what it does, how it can be used, and examples of how to use the CLI
    - The necessary dependencies so that the package works immediately upon installation in a new virtual environment
    - Working unit tests that test at least 70% of the code in the package

CLI Requirements

- Within the python package described in the package requirements section, there must also be a working command line interface (CLI)
- CLI methods must contain proper documentation that is accessible via the command line
- CLI method documentation should contain:
    - Explanations of the arguments and options involved with the method
    - Brief description of what the method is used for

Use of External Libraries

In general, one can make use of an external library or package that can aid in accomplishing a small subtask, such as a combinatorial problem, interface with an API, etc., but you cannot use a library or package capable of solving **all** of your tasks. You are of course allowed to use modified code from your previous individual assignments (including that of PLAB1) where

applicable. If you do choose to use an external resource to perform part of one of your tasks, it must be properly explained in the presentation. If you have any questions or concerns about whether a particular resource is allowed, please feel free to ask via email or issue.

General Remarks

- The tasks are purposely written in such as manner as to require you, as a group, to figure out what tools are needed, what information needs to be gathered, and what resources should be used
- All code-based work is to be done in GitLab
- Use GitLab Issues to track and assign individual tasks and required work
- The software package (backend code) and web application (frontend) can be stored in separate folders in the root directory of your repository as shown here (you can rename these folders as you please):

```
├── frontend
└── project_package
```

Grading (10 pts):

Task	1	2	3	4	5
Points	3	1	2	3	1

Likelihood Protein Identification via k-mer Genetic Sequence Assembly - Introduction

Molecular biologists and biochemists often utilize strands of DNA to perform experiments. This requires knowing the exact genetic sequence of the DNA they plan to use or modify. In order to sequence a large strand of DNA, one must first break it into smaller fragments, determine the base-pairs of these fragments, and reconstruct the original sequence from the individual reads. This is known as *shotgun sequencing*, and though newer techniques have been developed, this method of sequencing large strands of DNA revolutionized the field and made it possible to sequence the human genome.

Once the fragments have been read, they must be reconstructed or assembled. *De novo* sequence assembly is a technique for assembling shorter fragments of DNA into a longer one through the use of multiple alignments. This is in contrast to *reference mapping*, in which scientists use reference genomes for mapping their strands of genetic sequences to the proper location in its genome. Reference mapping is significantly faster as there is a template to which each strand can be compared against. However, before there were reference genomes, one had to use *de novo* sequence assembly, and it is still used when the DNA's organism is unknown. In this project, you will develop a pipeline for assembling fixed-length DNA strand reads into a complete sequence and predict which protein the completed strand may represent.

Aims

1. **Perform *de novo* sequence assembly** on a set of fixed-length DNA strands (k-mers)
2. **Transcribe and translate** the assembled DNA sequence
3. **Predict** which protein the assembled DNA strand may represent
4. **Create a frontend** which allows one to upload a series of DNA strand reads and obtain a list protein predictions

Tasks

Task 1 - *Strands, Assemble!* (3 pts)

- Given a list of DNA strands, implement an algorithm for performing multiple sequence alignment
- Use the multiple sequence alignment algorithm to reconstruct the genetic sequence of the used DNA strands

Task 2 - *Central Dogma* (1 pt)

- Using the compiled genetic sequence from Task 1, transcribe it to its corresponding mRNA sequence
- Predict possible amino acid sequences from the generated mRNA sequence
  - This may require checking the open reading frames (ORFs) and some exon splicing

Task 3 - *Protein Prediction* (2 pts)

- Assemble a list of possible proteins that the amino acid sequences generated in Task 2 may represent
  - This may require using tools available from UniProt, EBI, NCBI, BLAST, or elsewhere

Task 4 - *GUI* (3 pts)

- Construct a web interface that allows one to upload a file of DNA strands (such as a FASTQ file) and get a list of possible proteins. Your interface should include the following features:
  - An upload button that allows one to upload a file of DNA strands
  - A collapsable button that when pressed will show the resulting assembled DNA sequence
  - A collapsable button that when pressed will show the resulting mRNA sequence
  - A collapsable button that when pressed will show a table of possible amino acid sequences and the proteins they represent
  - A button that allows one to download this table of amino acid sequences/proteins as an Excel file or CSV

Task 5 - *Containerize the Application* (1 pt)

- Create a `Dockerfile` in your project's root directory that successfully compiles your application into a Docker image
  - Should bundle your backend and frontend code together
  - you may choose any base image
  - All members of group should be in the image metadata using `LABEL`
  - The `ENTRYPOINT` should start your web application and the web app should be accessible using port mappings