Assignment 4: Structures in Biology - 1D sequences

Due Thursday February 26, 5pm.

One of the most fundamental data type in bioinformatics is a sequence of characters, whether it be DNA, RNA, or amino acids. Each has a finite alphabet, possibly with some similarity measure or grouping. Sequences can be short (dozens or hundreds of elements), long (thousands or tens of thousands), or huge (millions or billions) in length. Visualization has been used to study individual sequences, usually using color to convey the value of elements, but a far more common use is to visualize relationships between two or more sequences. A particularly powerful mechanism is to compute a good alignment of the sequences (using insertions, deletions, and substitutions) and displaying this alignment in such a way as to emphasize where the two sequences overlap.

A common way of generating an optimal alignment of two sequences is by using a class of algorithms known as dynamic programming. This algorithm basically computes an alignment score that propagates from the start of the two sequences to the end, penalizing the alignment for mis-matches and adding gaps and rewarding it when the corresponding elements match. In some applications there may be degrees of match; for our purposes, you can consider matching of two elements to be a binary result (1 = match, 0 = mismatch). For the first phase of this project you need to implement the dynamic programming algorithm on two genetic sequences of your choosing. The input would be the 2 sequences plus the scores to be applied for matches, mismatches, and an inserted gap. The output will be an alignment, where for every position in one sequence there is either an index to the corresponding position in the other sequence or some indicator of a gap (e.g., a -1). An excellent tutorial on this algorithm can be found at http://www.avatar.se/molbioinfo2001/dynprog/dynamic.html.

Once you have a good alignment, you want to generate a visualization of the sequences and their alignment. This can be done in a purely textual manner (yes, a visualization can consist of just text, with additional information embedded in the positions), a purely graphical manner, or a combination of the two. To simplify things, you can limit the size of your sequences to less than a couple thousand to make it easier to fit onto the screen. You should look at how various bioinformatics programs display their alignments for inspiration.

These are the minimum requirements.

Remember that in the assignments part of your grade depends on going beyond the minimum requirements. There are many opportunities to move beyond the basics. For example:

1. Deal with the issue of scale. If your sequences are very long, to avoid scrolling you need to find ways of compressing things. Even if you are only

showing graphics (e.g., the color of a dot or small rectangle), you will run out of space eventually. One solution is to compress the alignments so that each dot or small rectangle represents the degree of match between two small subsequences, rather than individual sequence elements. Again, looking at existing systems may provide other ideas.

2. Expand your system beyond two sequences. For example, for 3 sequences you could make a 3-D scoring array and follow the dynamic programming algorithm with a higher dimension. Another would be to find regions of strong homology (similarity) between pairs of sequences and use one or more of these as foundations for partial alignments of the elements before and after the strong matches. Multiple sequence alignment is still an area of research, especially for large numbers of long sequences. Efficiency becomes a big issue.

Don't restrict yourself to these examples. The goal is to develop and implement your own ideas.

Resources

Matt Ward's course page has several links to sequence datasets. Feel free to search elsewhere (Prof. Ryder's slides, for example).

Requirements

1. Generate an alignment of (at least) two sequences.
2. Display a visualization of the sequences and their alignment.
3. The data should be nucleotide or amino acid sequences. FASTA format files might be the easiest to work with.
4. Include the datasets you use with your submission so I can run the program.

In your readme.md:

1. Describe the design of the visualization - what was considered for representing insertions, deletions, and substitutions? What happens if the sequences are longer than the display size?
2. Identify the source of the code if you start with code that you found.
3. For the free-form (beyond the requirements) component:
    - Describe the biological significance of the additions you make.
    - Describe the technical significance of the additions you make.
4. Provide instructions on running your program.

Turning in the project

Submit the following:

- All of the source code and data needed to run the program.
- A readme.md file containing the elements listed in the Requirements section

The source code must run on a Mac or Linux machine. (Note– don't worry about exporting Mac/Linux applications. The .pde files are fine.)

Submissions should be made using myWPI. Either:

1. (Recommended) Add text with a link to your GitHub repo, or
2. Zip your files and README and upload to MyWPI

Grading

Each homework assignment is graded on a 100 point scale:

- 85 points will be based on whether the program fulfills the minimum requirements.
- 15 points will be based the additions you made:
  - 7.5 Bio-related additions
  - 7.5 Technical additions

Total – 100

(0 will be assigned if the code can't be compiled or run.)