C343 Project 4 - DNA Sequence Alignment

Due 11:59pm, October 17, 2014

1 Assignment Description

Sequence Alignment is an important technique in understanding how *similar* two DNA sequences are. Applications of DNA sequence alignment range from determining gene function to finding similarity between species. Informally, alignment can be understood as writing the two sequences in rows, where the two characters in the same column are said to be aligned. If the two aligned characters are the same, we have a *match*; if the two characters are not the same, we have a *mismatch*. To try and maximize the number of matches, we can also insert gaps in either sequence. Mismatches can be interpreted as point mutations and gaps as insertion or deletion mutations. We disallow columns that consist of gaps only.

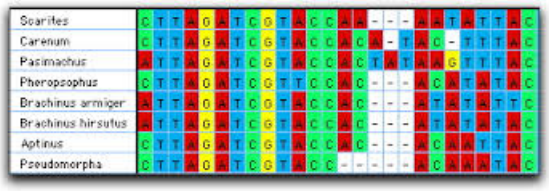

Figure 1: Alignment Example

There can be many alignments of two sequences, but the scores assigned to matches, mismatches, gaps determine which are the best alignments. An optimal solution can be found using a dynamic programming approach. Dynamic programming algorithms can be visualized by a board whose cells keep track of the best solution till that point. In bioinformatics, these are called *dot boards*.

The algorithms fills in the cells of this board with the best scores for every possible prefix of the alignment.

		A	C	A	C	A	C	T	A
	0	-1	-2	-3	-4	-5	-6	-7	-8
A	-1	2	1	0	-1	-2	-3	-4	-5
G	-2	1	0	-1	-2	-3	-4	-5	-6
C	-3	0	3	2	1	0	-1	-2	-3
A	-4	-1	2	5	4	3	2	1	0
C	-5	-2	1	4	7	6	5	4	3
A	-6	-3	0	3	6	9	8	7	6
C	-7	-4	-1	2	5	8	11	10	9
A	-8	-5	-2	1	4	7	10	9	12

Figure 2: Dot Board

2 Your Task

Your task is to complete the `seq_align` function in `seq_align.py`. This function takes three arguments:

- `s1` - the fist sequence.
- `s2` - the second sequence.
- `enable_graphics` - shows the dot board if this option is `True`.
- The function returns a pair of sequences representing the alignment.
- The score function named `s` is given to you. It takes as input two characters (from the same column) and returns the score.
- Suppose that your algorithm returns the sequences s'_1 and s'_2 of length n. The total score is computed as follows:

$$\sum_{k=0\ldots n-1} s(s'_1[k], s'_2[k])$$

  Your algorithm must find the alignment (the s'_1 and s'_2) that minimizes the total score.

You need to setup the recurrence equations for this dynamic programming algorithm, initialize the recurrence, compute the scores using the recurrence equations and finally trace the solution to produce the two aligned sequences. All of this is done in the `seq_align` function.

Here is an example of the completed dot board for aligning $X = AC$ and $Y = CA$. Here the `SPACE_PENALTY` is -1, a match is 2, and a mismatch is -2. To record our choice in each cell, we write D for delete (a gap in Y), I for insert (a gap in X), and M for either match or mismatch.

		i=0	1	2
			A	C
j=0		0	D=-1	D=-2
1	C	I=-1	M=-2	M=0
2	A	I=-2	M=0	D=-1

Therefore, a best alignment (not unique) for AC and CA is $(\_AC, CA\_)$.

3 Deliverables

Your repo's `project4` folder should contain all the files from the zip. These are the ones you need to modify:

- `seq_align.py` - containing your solution and tests
- `README.md` - where you need to explain your code

After that you can issue a pull request.

4 Testing

We have provided you with the `seq_align.py` file where you have to implement the `seq_align` function. There are also tests in `tests.txt` file which you can run by executing *run seq_align.py test*. We also encourage you to write your own test cases. A good way to do this is to use **assert** statements. So if your function must return `True` for given inputs, the test case looks like

```
assert f(i) == True
```

You can then put all your tests inside an `if` statement.

```
if __name__ == "__main__":
  # my unit tests
```

The tests will be executed when you execute this python script but not when you include this file into another script and run that. For more information look at `http://pythontesting.net/framework/unittest/unittest-introduction/`.

5 Running Your Code

As always, you should be able to execute the script in three modes:

- interactive - invoked by executing *python seq_align.py*
- test - invoked by executing *python seq_align.py test*. You can run the provided tests in this mode.
- gentests - invoked by executing *python seq_align.py gentests*. This mode helps you generate new tests.