Gene Sequencing Project

Emily Elzinga

October 27, 2022

1 Complexity

Explain the time and space complexity of your algorithm by showing and summing up the complexity of each subsection of your code.

- **The unrestricted algorithm runs in O(nm) time and space.** The unrestricted algorithm employs two nested for loops that run k and l times, where k is the length of the string arranged vertically and l is the length of the string arranged horizontally.

- **The banded algorithm runs in O(kn) time and space.** As in the unrestricted algorithm, the banded algorithm contains two for loops, the outside of which runs the length of the string on the vertical side of the table. Or, if the length of that string is considerably larger than the length of the other string, the length of the other string is used instead to allow the diagonal to be defined everywhere. The second for loop runs seven times, as was predefined in the project specifications.

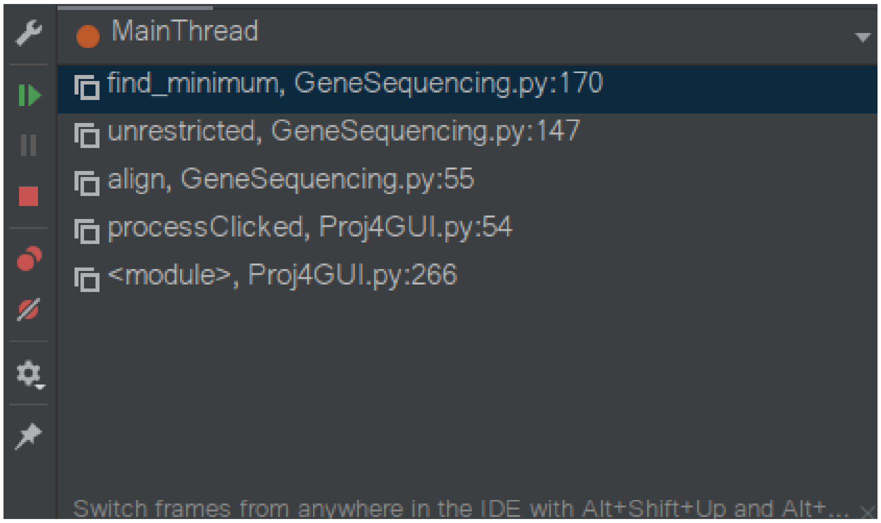
  There are also the pointer matrices which were created alongside the scoring matrices and adds no time complexity to the scoring algorithm itself, but there is another outside while loop needed to iterate through the pointer array and find the final alignments. This makes the time and space complexity for the unrestricted and banded algorithms 2mn and 2kn, respectively. These amount to O(mn) and O(kn).

2 Algorithm Explanation

(10 points) Write a paragraph that explains how your alignment extraction algorithm works, including the backtrace The first step in both the unrestricted and banded algorithms was to initialize the matrices being used. In the unrestricted algorithm, I made the matrix using a list of columns rather than rows, which probably wasn't the best way since matrix notation lists the row first when describing a cell. I made it work with the simple test cases though, and didn't want to try fixing what wasn't broken. However, my values for the longer sequences are slightly off, and I am not sure why.

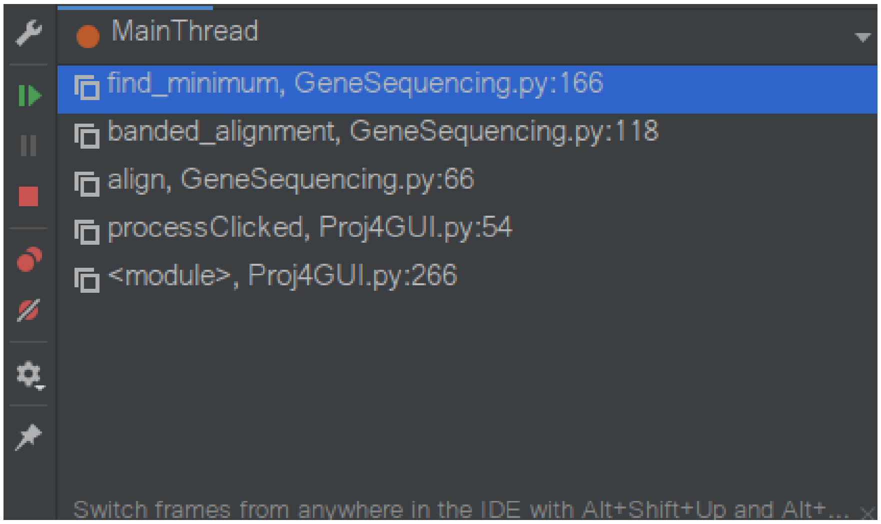
Below are shown the stack traces for the deepest levels of the two algorithms. The "unrestricted" and "banded_alignment" methods are concerned with building and inserting values into the score and pointer matrices. These contain the cost values and their pointers, respectively. If the letters at the current cell does not give a match, the "find_minimum" method is called to find the min of the alternatives.

Once the matrices are filled, the "find_alignment" method is called to iterate through the pointer matrix and get the alignment.



```
MainThread
find_minimum, GeneSequencing.py:170
unrestricted, GeneSequencing.py:147
align, GeneSequencing.py:55
processClicked, Proj4GUI.py:54
<module>, Proj4GUI.py:266
Switch frames from anywhere in the IDE with Alt+Shift+Up and Alt+...
```

Figure 1 – Stack trace of the unrestricted algorithm to the deepest level

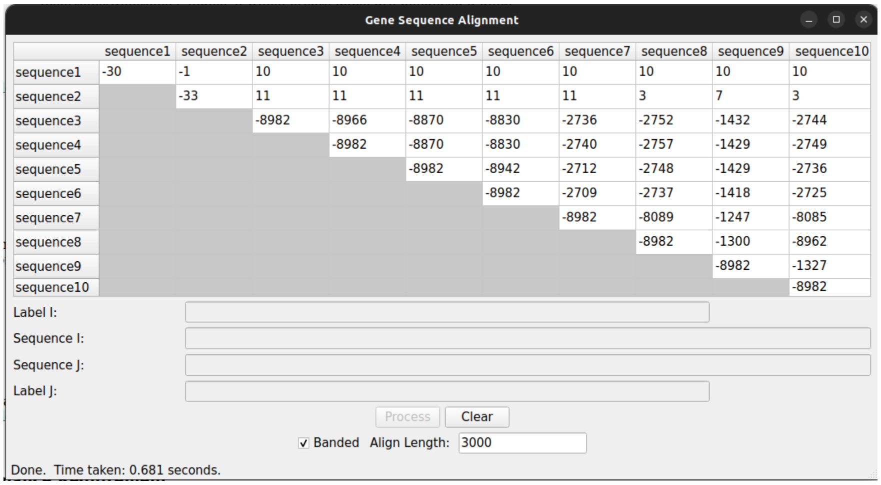


```
MainThread
find_minimum, GeneSequencing.py:166
banded_alignment, GeneSequencing.py:118
align, GeneSequencing.py:66
processClicked, Proj4GUI.py:54
<module>, Proj4GUI.py:266
Switch frames from anywhere in the IDE with Alt+Shift+Up and Alt+...
```

Figure 2 – Stack trace of the banded algorithm to the deepest level

3 Results

[20 points] Include a "results" section showing both a screen-shot of your 10x10 score matrix for the unrestricted algorithm with align length k = 1000 and a screen-shot of your 10x10 score matrix for the banded algorithm with align length k = 3000.



	sequence1	sequence2	sequence3	sequence4	sequence5	sequence6	sequence7	sequence8	sequence9	sequence10
sequence1	-30	-1	10	10	10	10	10	10	10	10
sequence2		-33	11	11	11	11	11	3	7	3
sequence3			-8982	-8966	-8870	-8830	-2736	-2752	-1432	-2744
sequence4				-8982	-8870	-8830	-2740	-2757	-1429	-2749
sequence5					-8982	-8942	-2712	-2748	-1429	-2736
sequence6						-8982	-2709	-2737	-1418	-2725
sequence7							-8982	-8089	-1247	-8085
sequence8								-8982	-1300	-8962
sequence9									-8982	-1327
sequence10										-8982

Label I:

Sequence I:

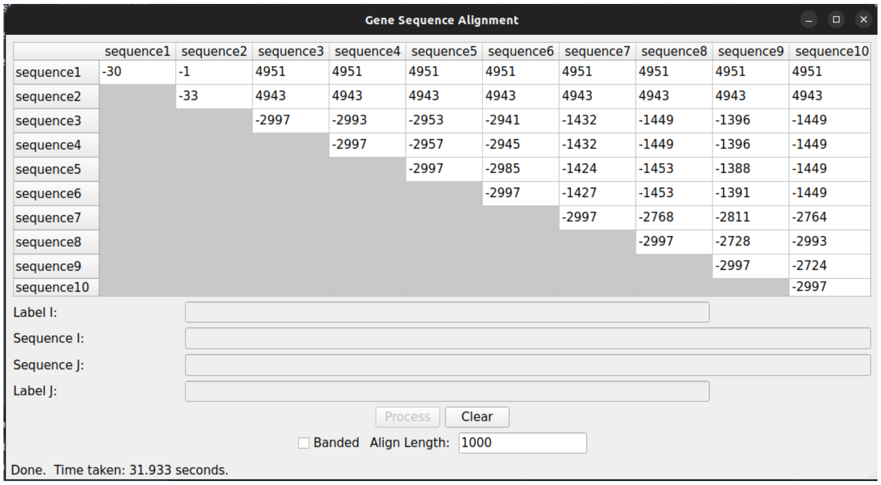
Sequence J:

Label J:

☑ Banded Align Length: 3000

Done. Time taken: 0.681 seconds.

Figure 3 – Output of code for banded algorithm and k=3000



	sequence1	sequence2	sequence3	sequence4	sequence5	sequence6	sequence7	sequence8	sequence9	sequence10
sequence1	-30	-1	4951	4951	4951	4951	4951	4951	4951	4951
sequence2		-33	4943	4943	4943	4943	4943	4943	4943	4943
sequence3			-2997	-2993	-2953	-2941	-1432	-1449	-1396	-1449
sequence4				-2997	-2957	-2945	-1432	-1449	-1396	-1449
sequence5					-2997	-2985	-1424	-1453	-1388	-1449
sequence6						-2997	-1427	-1453	-1391	-1449
sequence7							-2997	-2768	-2811	-2764
sequence8								-2997	-2728	-2993
sequence9									-2997	-2724
sequence10										-2997

Label I:

Sequence I:

Sequence J:

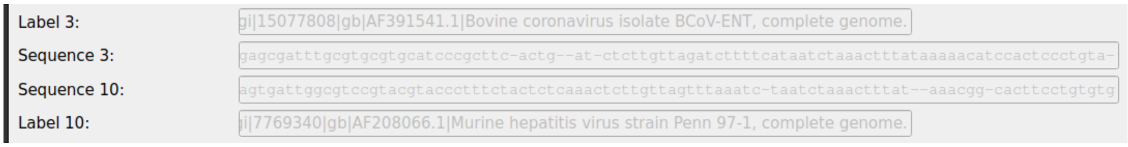
Label J:

☐ Banded Align Length: 1000

Done. Time taken: 31.933 seconds.

Figure 4 – Output of code for unrestricted algorithm and k=1000

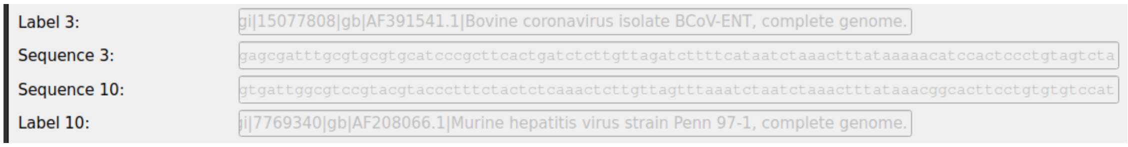
[10 points] Include in the "results" section the extracted alignment for the first 100 characters of sequences 3 and 10 (counting from 1), computed using the unrestricted algorithm with k = 1000. Display the sequences in a side-by-side fashion in such a way that matches, substitutions, and insertions/deletions are clearly discernible as shown above in the To Do section. **Shown below in Figure 5.**



```
Label 3:        gi|15077808|gb|AF391541.1|Bovine coronavirus isolate BCoV-ENT, complete genome.
Sequence 3:     gagcgatttgcgtgcgtgcatcccgcttc-actg--at-ctcttgttagatcttttcataatctaaactttataaaaacatccactccctgta-
Sequence 10:    agtgattggcgtccgtacgtaccctttctactctcaaactcttgttagtttaaatc-taatctaaactttat--aaacgg-cacttcctgtgtg
Label 10:       gi|7769340|gb|AF208066.1|Murine hepatitis virus strain Penn 97-1, complete genome.
```

Figure 5 – Alignment for code for unrestricted algorithm and k=1000

Also include the extracted alignment for the same pair of sequences when computed using the banded algorithm and k = 3000 **Shown below in Figure 6**.



```
Label 3:        gi|15077808|gb|AF391541.1|Bovine coronavirus isolate BCoV-ENT, complete genome.
Sequence 3:     gagcgatttgcgtgcgtgcatcccgcttcactgatctcttgttagatcttttcataatctaaactttataaaaacatccactccctgtagtcta
Sequence 10:    gtgattggcgtccgtacgtaccctttctactctcaaactcttgttagtttaaatctaatctaaactttataaacggcacttcctgtgtgtccat
Label 10:       gi|7769340|gb|AF208066.1|Murine hepatitis virus strain Penn 97-1, complete genome.
```

Figure 6 – Alignment for code for banded algorithm and k=3000

[30 points] Attach your commented source code for both your unrestricted and banded algorithms.