Please upload your solutions on Gradescope. You can use \LaTeX or a word document to write up your answers, but we prefer you use \LaTeX. You may scan hand-written work or images for parts of solutions only if they are extremely clean and legible. Please ensure that your name does not appear anywhere in your handin.

**Problem 0: Readings**

Read the following sections of the Barton textbook chapter, which can be found on the assignments page next to HW4. The material will help lay a foundation for Problems 4 and 5.

(a) *For Distance Methods, Corrections Are Essential to Convert Measures of Similarity to Evolutionary Distances*, pgs. 27 - 29

(b) *UPGMA and Neighbor-Joining Methods*, pgs. 22 - 26

**Problem 1: Burrows-Wheeler transform**

With the Winter Olympics fast approaching, many Olympians are receiving countless packages filled with mail from adoring fans! However, these letters are often very long and are vast in number, so a computational biologist, who happens to be an Olympics fanatic as well, seeks to help shorten the letters and save storage space.

(a) Task: What two properties of the Burrows-Wheeler transform make the algorithm well-suited for use in data compression?

(b) Task: Apply the Burrows-Wheeler transform to the string $AGCAGCTAG\$. Show your work.

(c) Task: Apply the inverse Burrows-Wheeler transform to the string $TGGCAA\$. Show your work.

(d) Suppose you are given the following string which is the output of the Burrows-Wheeler transform: $TTT\$GATTTCAA$. One of the following three options is the original string. Task: Without actually doing the inverse transform, determine which of the following is the only option that could possibly be the original string. Explain your answer.

1. $CTTATTGT\$TCC\$
2. $ATTATTAG\$GCTT\$
3. $TTGACCTT\$GT\$T\$
Problem 2: Building and using suffix trees

The Olympic cauldron is about to be lit! But first, the final runner carrying the torch needs help navigating the long and convoluted hallways of the stadium. However, despite the magical atmosphere of the Olympics, in order to reach the cauldron the runner must use something more complex than a silly “abracadabra” spell. Instead, they must find all the suffixes of the word, and they’re enlisting your help to do so!

(a) Task: Construct the expanded suffix tree $T_x$ for the word $x = \text{ABRACADABRA}$. Build the tree one suffix at a time, starting with $\text{suf}_1$. Show the first four partial trees in your construction as well as the complete suffix tree $T_x$.

(b) Task: Show how to use your suffix tree to return the starting indices of all occurrences of the string $\text{ABR}$ in the text $x$. List all of these indices. In general, what is the big-$O$ time complexity of returning the starting indices of all occurrences of a pattern $p$ in a text $t$? Explain.

(c) Task: Show how to use your suffix tree to determine whether the string $\text{ACADEMY}$ occurs in $x$. Does it occur in $x$? In general, what is the big-$O$ time complexity of determining whether a pattern $p$ occurs in a text $t$? Explain.

(d) Bonus: Construct the compacted suffix tree and the position suffix tree for the word $x = \text{ABRACADABRA}$. Explain why position suffix trees can be stored in $O(n)$ space, whereas compact suffix trees require $O(n^2)$ space.

Problem 3: Longest shared substring

Consider the following problem.

<table>
<thead>
<tr>
<th>LONGEST SHARED SUBSTRING PROBLEM</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Input:</strong> Strings $u$ and $v$.</td>
</tr>
<tr>
<td><strong>Output:</strong> The longest substring that occurs in both $u$ and $v$.</td>
</tr>
</tbody>
</table>

In class, it was briefly mentioned that we could solve this problem by concatenating the two strings and constructing a joint suffix tree for the resulting string. In this problem, we expand upon this idea.

We begin by appending two different end-of-string characters to $u$ and $v$, giving us $u\#$ and $v\$. We then build the suffix tree for the concatenation of these two strings, $u\#v\$. We shall color the leaves of this tree according to the following rule: if a leaf is labeled by the starting position of a suffix starting in $u$, color it blue; if a leaf is labeled by the starting position of a suffix starting in $v$, color it red. Note that all the leaves in the tree will be colored either red or blue according to this procedure.
Next we color the internal nodes of the tree according to the following rules:

- A node is colored blue (resp. red) if all the leaves in the subtree rooted at that node are blue (resp. red).
- A node is colored purple if the subtree rooted at that node contains both blue and red leaves.

At this point, you are strongly encouraged to run through this procedure on a pair of simple example strings, so that you can better understand the logic and purpose of this coloring scheme. You will not be graded on whether or not you create an example, so feel free to share your example with your classmates and TAs on Piazza.

For all tasks in this problem, assume that we are using **expanded** suffix trees.

(a) Task: Explain how every path ending in a purple node in the suffix tree of \( u \# v \) spells out a substring shared by \( u \) and \( v \).

(b) Task: Explain how every path ending in a blue (resp. red) node in the suffix tree of \( u \# v \) spells out a substring that appears in \( u \) but not in \( v \) (resp. \( v \) but not in \( u \)).

Given these two facts, it should be clear how we find the longest shared substring of \( u \) and \( v \): We need only examine the strings spelled by paths that lead to purple nodes; the longest such string is precisely the solution to the Longest Shared Substring Problem.

The above approach makes use of a single suffix tree to solve the Longest Shared Substring Problem. It is also possible to solve this problem using two suffix trees, one for each of \( u \) and \( v \).

(c) Task: Describe an algorithm that solves the Longest Shared Substring Problem using the two suffix trees \( T_u \) and \( T_v \).

(d) Task: Compare (in as much detail as possible) the worst case space- and time-efficiency of the algorithm that uses a single suffix tree for \( u \# v \) with the worst case space- and time-efficiency of the algorithm you designed in (c) above.

**Problem 4: Jukes-Cantor Distance**

(a) To get the feel of how to calculate the Jukes-Cantor distance, consider two biological sequences that have accumulated mutations over time. Below are snapshots of the two sequences at 4 different points in evolutionary time:

<table>
<thead>
<tr>
<th>Sequence 1</th>
<th>Sequence 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>t = 1</td>
<td>AGGTCA</td>
</tr>
<tr>
<td>t = 2</td>
<td>AGCTCA</td>
</tr>
<tr>
<td>t = 3</td>
<td>AGCTCA</td>
</tr>
<tr>
<td>t = 4</td>
<td>AACTCA</td>
</tr>
</tbody>
</table>

Task: For each timepoint \( t \), calculate the fraction of sites that are different between the two sequences, \( \lambda_t \). Then use this value to calculate the Jukes-Cantor distance \( D_t \) between each sequence.
Task: Compute $\lambda$ as above. What happens when you try to use this value to calculate the Jukes-Cantor distance?

(c) Task: Propose a correction to the Jukes-Cantor distance that avoids the issue above. What result do you get when you calculate your new distance at $t = 5$? Explain in two or three sentences the potential benefits and drawbacks of your new distance. It may be helpful to read up a little on the Infinite Sites Model.

**Problem 5: Neighbor Joining**

While watching the Winter Olympics on TV, the CS181 TAs sit in awe as their favorite figure skater, Japan’s Yuzuru Hanyu, stuns with his quadruple toe loop-triple Axel sequence during the men’s singles skating event. Wanting to understand the progression of famous figure skaters over the decades, one of the TAs decides to build an evolutionary tree for various icons, including Sonja Henie, Kristi Yamaguchi, Evgeni Plushenko, and Richard Button, based on their active years in international competition.

<table>
<thead>
<tr>
<th></th>
<th>Sonja Henie</th>
<th>Richard Button</th>
<th>Kristi Yamaguchi</th>
<th>Yuzuru Hanyu</th>
<th>Evgeni Plushenko</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sonja Henie</td>
<td></td>
<td>135</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Richard Button</td>
<td>159</td>
<td>24</td>
<td>213</td>
<td>54</td>
<td></td>
</tr>
<tr>
<td>Kristi Yamaguchi</td>
<td></td>
<td>78</td>
<td>177</td>
<td>42</td>
<td>18</td>
</tr>
<tr>
<td>Yuzuru Hanyu</td>
<td>213</td>
<td>78</td>
<td>177</td>
<td>42</td>
<td>18</td>
</tr>
<tr>
<td>Evgeni Plushenko</td>
<td></td>
<td>24</td>
<td></td>
<td>18</td>
<td>36</td>
</tr>
</tbody>
</table>

Task: Use the distance matrix above and the neighbor joining algorithm to construct an evolutionary tree for the figure skaters above. In a sentence or two, comment on Evgeni Plushenko’s place in the evolution of world-class figure skaters.

**Problem 6: PyMOL Visualization**

In Homework 1, you identified the protein sequence of the reference spike (S) gene of the SARS-CoV-2 coronavirus. The spike glycoprotein is the major surface protein that SARS-CoV-2 uses to bind to a human cell receptor. When the spike protein binds to this human cell receptor, the viral membrane can adhere to the human cell membrane, allowing the viral genome to enter and infect human cells. For this reason, the spike glycoprotein is a key target for vaccines and therapeutic antibodies. In this question, you will get a better understanding of the 3D structural representation of the SARS-CoV-2 S protein.

To complete this question, you will need to download a molecular visualization software called PyMOL. You can download a free trial of the software from this link. You can interact with the software via the
command interpreter in the top half of the interface. The bottom window is the Viewer screen, where you can interact, rotate, and manipulate the model. In this problem, we will walk you through what commands you will need to generate the model.

Specifically, we will be visualizing 6VSB, the viral protein structure 2019-nCoV spike glycoprotein. Follow the command line steps below and complete the corresponding tasks.

**Step 1. Setup**

PyMOL> fetch 6vsb
PyMOL> hide all
PyMOL> show cartoon, 6vsb
PyMOL> color cyan, chain A
PyMOL> color red, chain B
PyMOL> color green, chain C

**Step 2. Zoom In**

PyMOL> remove chain A
PyMOL> remove chain B
PyMOL> show surface, 6vsb
PyMOL> color cyan, chain C
PyMOL> set transparency=0.5
PyMOL> hide surface

**Task:** How is the cartoon ribbon structure of 6vsb from Step 1 different from the Connolly surface representation in Step 2? Give an example of what unique information each schematic representation provides.

**Step 3. Select Protein Domain**

PyMOL> set seq_view, 1
PyMOL> select rbd, (resi 333-528)
PyMOL> color white, rbd
PyMOL> hide all
PyMOL> show cartoon, rbd
PyMOL> zoom rbd

**Step 4. Select Active Site**

PyMOL> select cysteines, resn cys
PyMOL> color red, cysteines
PyMOL> show sticks, cysteines
PyMOL> select active_I, (resi 336-361)
Step 5. Save and Output Picture

File> Save Session As
PyMOL> bg_color white
PyMOL> ray
File>Export Image As

Task: Attach a saved image of the protein surface from Step 4. What does each color identify? Record the protein sequence for the first active site residue range (336-361).

6crz represents SARS spike glycoprotein. In this final step, we will use PyMOL to align this protein with 6vsb.

Step 6. Alignment

PyMOL> fetch 6vsb
PyMOL> fetch 6crz
PyMOL> hide all
PyMOL> show cartoon
PyMOL> remove chain A
PyMOL> remove chain B
PyMOL> align 6vsb, 6crz
PyMOL> select 6vsb_rbd, (resi 333-528) and not 6crz
PyMOL> select 6crz_rbd, (resi 303-490) and not 6vsb
PyMOL> align 6vsb_rbd, 6crz_rbd

Task: How is structural alignment different from sequence alignment? Describe a possible benefit to using structural alignment to align known proteins.

Problem 7: HIV Phylogenetics

The phylogenetics of a virus can be very helpful for uncovering epidemiological, immunological, and evolutionary processes. In this problem, we will investigate the phylogenetics of HIV along with some
related ethical considerations described in this paper.

a. Why is the molecular epidemiology of HIV particularly important to learn about? What characteristic of HIV makes phylogenetics a useful tool for studying the virus?

b. Identify and explain the two main mechanisms described in the paper for how the privacy of HIV phylogenetic data can be compromised. What are the two main risks of privacy loss identified in the paper? How are these risks related?

c. What is the “black box” method of data storage? What are potential advantages and drawbacks of this solution?

**Problem 8: Genealogy**

GEDmatch is a DNA comparison and analysis website that aggregates user-uploaded DNA files. It was used by law enforcement to identify the Golden State Killer in 2018 after decades of investigation. GEDmatch’s databases can be queried without a warrant and have played a role in solving various other cases.

Read through GEDmatch’s Terms of Service and Privacy Policy.

a. How may this service affect individuals who are not direct customers? How can these effects be mitigated, if possible?

b. Do these terms raise any concerns to you? What is your perception of GEDmatch as a company and why? Cite specific details to support your claim.

c. How may GEDmatch benefit from providing users with its services?