Nate Evans

Christina Zheng & Shannon McWeeney

BMI650: Algorithms

10/12/18

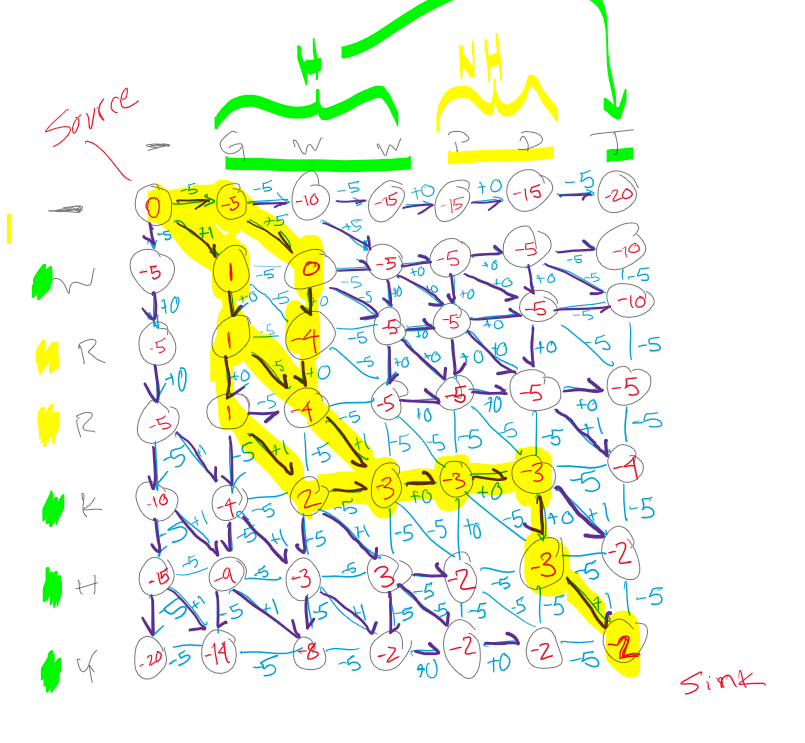
HW#2

**Attempt 1**

First attempt, I used circles to represent nodes, red numbers to represent the node value and blue lines to represent edges, if it was a trace back edge I used purple. Systematically, this worked fairly well although the nodes / edges were difficult and time intensive to draw cleanly. The error with this method did not use a gap penalty of -2, rather I treated an indel as non-hydrophobic, thus the majority of indels were penalized with -5.

As it is currently represented, there are several global alignments tied for the top score: The top most route (visually) is aligned as:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| G | W | - | - | W | P | D | - | T |
| - | W | R | R | K | - | - | H | Y |

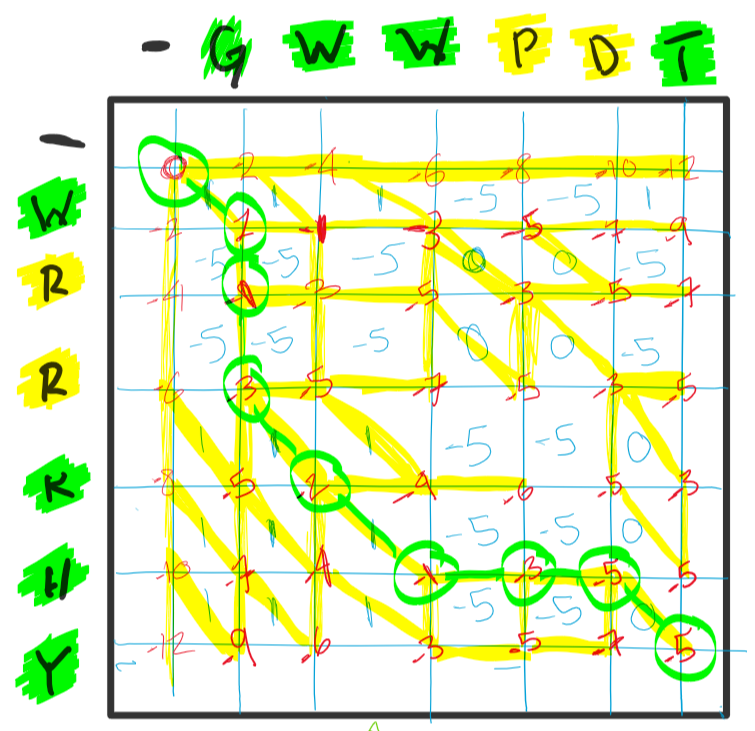


**Attempt 2**

This attempt I created a grid where blue lines represent indel transitions and the blue value or square represented diagonal (match/mis-match) transitions, line intersections represented nodes and the red values at each intersection was node weights. This method was much easier to set up and draw and systematically it was straightforward to fill out all the transition scores (blue numbers, line transitions represent -2 gap penalty) but I found the node representation less intuitive. However, in the end, it was my own negligence that caused this to fail, misrepresented the W-W match score as 1 which propagated through the whole grid.

Were my scoring system to have been filled out correctly, this method would have worked well. As it currently stands, the represented global alignment is:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| G | - | - | W | W | P | D | T |
| W | R | R | K | H | - | - | Y |



**Attempt 3 / Program Check**

My latest attempt started out as a method of checking my work (above) by writing a program to calculate the edge and node weights of the described system. In the matrix below…

0 1 2 3 i

0 \*0\* |-2|\*-2\*|-2|...

1 -2 |-5| -2 |-2|...

2 \*-2\*|-2|\*-4\*|-2|...

j

nodes - i & j are even

edges - all other times

indel when i=odd, j=even

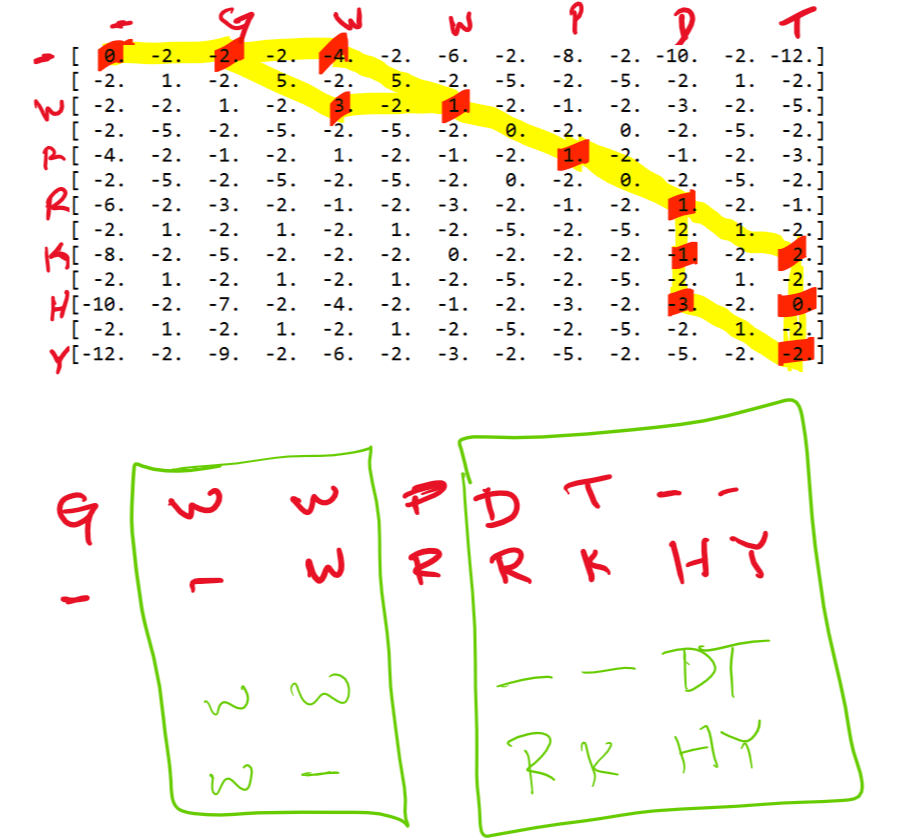
indel when i=even, j=odd

diag when i=odd, j=od

This program does not backtrack or provide sequence alignment output, and I manually backtracked the alignment. This however could be done with limited changes to the current code, which I have attached as a .py file. Apologies for the lack of commenting, it was done in a rush.

Global alignment has 4 equal scoring global alignments but to simplify the alignment representation, the divergent components are shown below, rather then showing 4 separate alignments.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| G | W | W | P | D | T | - | - |
| - | - | W | R | R | K | H | Y |
|  | W | W |  | - | - | D | T |
|  | W | - |  | R | K | H | Y |



The alignment of the sequences makes sense given the scoring metric H: The W-W match is favored while hydrophobic – non-hydrophobic were avoided (G-). Given the relative simplicity of this scoring system and the shortness of these sequences, it is easy to see how this problem becomes unrealistic for human computation very quickly. In fact, human computation is arguably unreliable even at this level, I did it wrong twice before using a computer!