Allen Proulx | PA5 Report

The approach I used relies heavily on the properties of the problem. To create my longest sequence, I first find all the pairs of two numbers that are their own 2 length sequence. Then, I go through and connect all these 2 length sequences. I first create a matrix of nodes that stores all the values. When I read the numbers in I store their index from the original genome. I use their value as an index for the matrix. This makes it so the positions of each value are stored at the same column index. This allows me to traverse down the matrix without searching each row to find the positions of the value I want.

As I am creating the matrix of nodes, I also link the sequence together in original order. This was I can traverse the position matrix by value instead of position. All of this is done in O(N*M) time because it is just one pass over the whole matrix. I then traverse the first row of the matrix in original order. I go through and compare each value to every value which comes after it by compare the two position values. I keep going down the matrix until I reach the bottom, then I know it is a sequence and I add it to the list of values that the node points to. If my comparison fails, it goes to the next node. I do this for all the values in the row. So, this is O(M*N!) time at the worst case.

I then traverse the original row again, this time I go through the link to find the longest. Each node of the top row has a linked list of points to other nodes in the top row. I go through the linked list for each node and traverse the pointers until I get to a node that doesn't have any link. Then I return the updated the length value of the pointer. After I have done this for all the links of the node. I discard all but the longest link. This allows be to not have to go down the same path again. While I am traversing the original row, I keep track of the longest linked sequence. After I have traversed the whole row. I return the longest sequence found and the length. The time complexity of this part at the worst case is probably close to O(2N). This is because I traverse the whole row so N and for each linked list I stop if I have gone down the path.