Introduction

• DNA sequencing is faster and easier than ever, and much research involves analysis of relationships between different parts of genetic sequences.

• These relationships may be of a variety of types depending on the goal of the research, for instance, finding identical or very similar blocks of genes within an entire genome to attempt to investigate evolutionary relationships between species.

• Visualization has been important to the interpretation of the results from the beginning, due to the scale of the data involved.[1]

• Because of the extremely large size of the data sets involved, visualization of the data in an interactive way is computationally challenging.

Existing Visualization Tools

• Many tools have been developed to visualize genetic sequence and comparison visualization applications (see Table 1).

• Different methods of visually representing the data have developed over time. Initially, two sequences being analyzed were visualized as the two axes of a scatter plot, with dots indicating related regions.

• The type of visualization in widest use today presents sequences as colored blocks on the circumference of a circle, with links across the circle indicating related regions.

• Circos[6] is the current standard for this type of visualization, but lacks interactivity and cannot produce visualizations of data beyond a modest size.

• MzBee[7] and Gremlin[8] have been developed more recently and include various interactive features, but Gremlin displays data in a linear arrangement (and also has yet to be released since [8] was published in 2010) and MizBee lacks Circos' flexibility in terms of input data type.

Table 1: Trends over time in development of visual presentation and interactive capability of genomic comparison visualization applications.

<table>
<thead>
<tr>
<th>Name</th>
<th>Year</th>
<th>Dot Plot</th>
<th>Linear</th>
<th>Circular</th>
<th>3D</th>
<th>Interactive</th>
</tr>
</thead>
<tbody>
<tr>
<td>MzBee[7]</td>
<td>2009</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Gremlin[8]</td>
<td>2010</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CMap3D[9]</td>
<td>2010</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>New Tool</td>
<td>2013</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Development Process

• Based on my survey of existing software and related literature a set of basic requirements was identified for the first implementation of a new interactive visualization tool.

Basic Requirements

• Colour of links is determined by their weight (Fig. 1).

• Users can limit view of links to those with weights within a specified range (Fig. 2).

• Users can limit displayed links to those within specified sections of the genome (Fig. 3).

Additional Features

• After the first requirements were implemented, in consultation with Andrian Yang and Leslie Burnett regarding their use of this tool to aid in their research, the ability to set the maximum opacity of the links (compare Figs. 1 & 2) and several labelling features were added to the program.

Implementation Details

Link Bundling

• To conserve memory and computation during rendering, links are stored in bundles.

• Links are stored in the same bundle if they meet certain criteria, the main being that if they were rendered separately, they would be overlapping at both ends, and the difference between their weights is below a certain threshold.

Data Structure

• Bundles are stored in a three-dimensional quadtree.

• Bundles are rendered rather than individual links.

• Utilizing this type of structure allows range queries to be performed on the links for filtering O(d log n) time[10] where d is the number of links in the query range.

Discussion and Future Work

Evaluating This Tool

• This stage of development has focused on creating a program which can be used interactively with data at a very large scale.

• Initially loading and processing the data in Fig. 1 took 17.82 seconds on our machine. Fig. 2 took 14.31 seconds.

• Direct comparison of running time against Circos is challenging because Circos only produces static images. In any case, Circos was not able to render the data seen in Fig. 1 at all.

Further Development

• Changes to how and when data is rendered should address the issues with interactivity because of the size of the data.

• The development process has been iterative, and should continue with more focus on refining the tool and extending its features and range of uses.