Disease Outbreaks Team

Abdulaziz Alhawas
Jean-Paul Labadie
Jordan Marshall
Luis Valenzuela

Introduction

For this project we will be using Java, JavaFX, D3.js, IntelliJ IDEA, and Apache Maven. Java will provide the core functionality of the application, supporting the GUI by allowing us to connect to remote information centers, allowing integration of different application components, generating XML instruction sets for the NASP pipeline, etc. The GUI will be built to run locally with JavaFX, both of which are well known GUI frameworks for Java. The JavaScript framework D3.js will be used for providing visualizations of the data output from a finished NASP job.

The Northern Arizona SNP Pipeline is a tool used by our sponsor, Tgen, to investigate fungal and bacterial genomes for the purpose of tracking disease outbreaks. The pipeline is optimized to run in a computing cluster, which are typically accessed remotely. Currently, users interact with a command-line program, which tries to emulate a ‘wizard’ style, asking the user for the locations of various files, and options to include when running the process. This command-line program generates a formatted XML document, which is then passed to the computing cluster. The NASP tool uses the information in the XML to run the process.

Our GUI will replace the command-line tool entirely, providing a robust and modular interface which improves the user experience. This goal is well within our abilities, as the command line tool is sufficiently abstracted from the NASP job process. Because the NASP pipeline is dependent solely on the XML, our GUI is completely independent of the other technologies used in NASP. The core requirement of the project is then to provide a user interface which can generate this XML and pass it to NASP pipeline to begin a job. Features of the project will build upon this, including providing a visual representation of job completion by interacting with the job manager, and visualizing the phylogenetic trees which result from the job once it has finished.

Technology Overview

For this project we have considered using Java Swing, JavaFX, or PyQT. These are all GUI framework toolkits, Swing and FX pertain to Java, whereas PyQT is leveraged through Python. All of these technologies fulfill the need to create a GUI which is the main task for our project.

PyQT is a set of Python bindings which contains over one thousand classes. These classes can be used to implement GUI widgets, databases, text editors, xml parsers, and SVG support. PyQT brings together C++ cross-platform application framework QT and integrates in with Python. Since the code provided by our sponsor is already in Python this was our first choice. However, after concluding that we were all more fluent in Java we moved on to the next two choices.

Swing is a Java API to provide a graphical user interface for Java projects. Swing was originally created to make a more sophisticated set of GUI components than the earlier tools. It supports a pluggable look and feel that is more user friendly, and it also provides several advanced components to the user in addition to the basics.

JavaFX is a software platform for creating and delivering desktop applications, as well as rich internet applications (RIAs) that can run across a wide variety of devices. JavaFX is intended to replace Swing as the standard GUI library for Java SE, but both will be included for the foreseeable future. JavaFX is ultimately what our group has decided to go with.

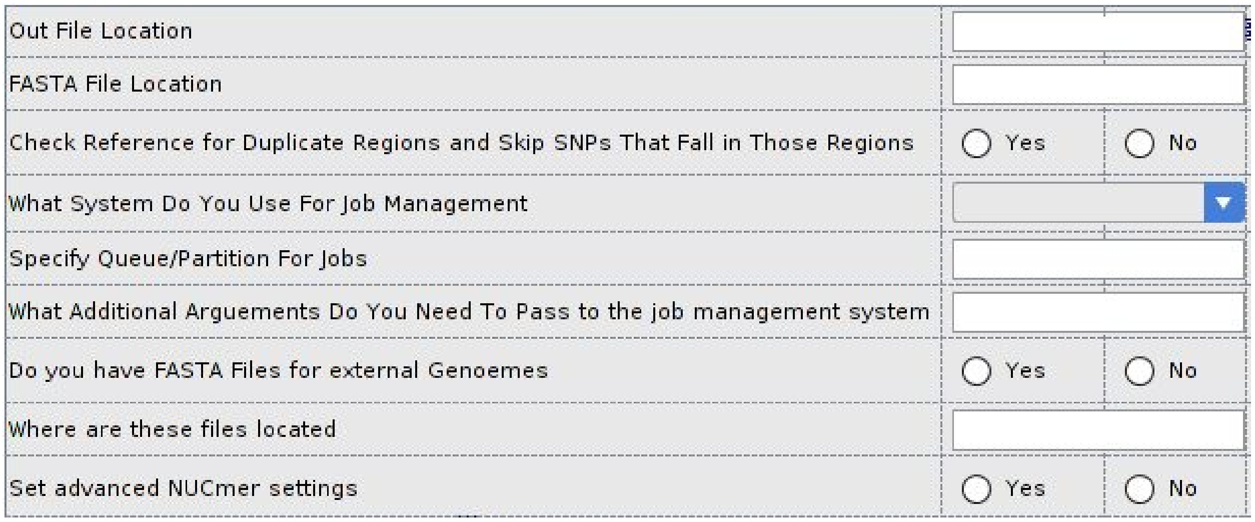
Technology Integration

Out if all the technologies we considered, we decided to use Java with JavaFX as the GUI framework. We also intend to use D3.js for visualizations as an additional feature.

These technologies fulfill our requirements and will seamlessly integrate with our particular environment. Obviously, a Java-based application will run on any major operating system. Furthermore, since the NASP tool is abstracted behind its XML input, no difficulties will be encountered in interfacing with it. It is well within the capabilities of Java to generate XML and interact with a remote machine and its process managers. Serving the visualizations generated by any D3.js scripts is likewise well-handled by our chosen technologies. JavaFX is designed for both local GUI implementation and serving rich web-content, and thus can serve the content generated by D3.js.

Proof of Feasibility

The following is an extremely early draft of a part of the system that we will eventually implement in its entirety. This shows that javaFX has what we need to produce the interface for the command line tool that is currently being used. After the interface is complete we will then move on to integrating it with the tool. That integration will include the generation of the XML file that NASP requires based on the input that a user enters into the interface. As far as the visualization is concerned we have not had the opportunity to test whether D3.js produces a visualization that we like but based on what we have seen we are confident that it should work for what need it to do.



Out File Location		
FASTA File Location		
Check Reference for Duplicate Regions and Skip SNPs That Fall in Those Regions	◯ Yes	◯ No
What System Do You Use For Job Management	▼	
Specify Queue/Partition For Jobs		
What Additional Arguements Do You Need To Pass to the job management system		
Do you have FASTA Files for external Genoemes	◯ Yes	◯ No
Where are these files located		
Set advanced NUCmer settings	◯ Yes	◯ No