**An Incremental Enhancement of MASS GUI**

**Luo Leng**

A term report

submitted in partial fulfillment of the

requirements of the degree of

Master of Science in Computer Science & Software Engineering

**University of Washington**

**March 12th, 2023**

**Project Committee:**

Prof. Munehiro Fukuda, Ph.D., Chair

Prof. Annuska Zolyomi, Ph.D.,Member

Prof. Robert Dimpsey, Ph.D., Member

## **1.Project Overview**

This project primarily focuses on the significant enhancement and development of the Multi-Agent Spatial Simulation (MASS) Graphical User Interface (GUI). The current GUI includes features such as interactive program execution, data structure visualization, and computational rollback, but its scope is relatively restricted. To provide a more comprehensive visualization experience for all MASS users, incremental enhancements are necessary. Currently, the process of visualizing data structures is disjointed. DsLab students and developers are required to manually import data from the MASS application to Cytoscape, a procedure that involves several complex steps. My project aims to streamline this process, improve user interaction, and provide additional functionalities to help MASS users gain a deeper understanding of their simulations. The sections 1.1 - 1.4 briefly discuss the current tools that I intend to use and enhance.

### **1.1 MASS**

MASS is a parallel-computing library designed for large-scale agent-based microsimulations and data science applications. It can support a broad range of simulations like wave dissemination, molecular dynamics, social networks and artificial life scenarios[1].

At the heart of MASS's parallelization model are two core components: Places and Agents[2]. Places represent elements statically allocated across various computing threads and nodes, while Agents are entities that can move across different processes. The library provides a set of APIs that abstracts thread and process management, including spawning, mapping, execution, communication, and termination.

### **1.2 InMASS**

Interactive MASS (InMASS) allows line-by-line MASS simulations through a command-line interface. This is achieved by wrapping Java's JShell interface with additional support classes for functioning in MASS' distributed environment. Figure 1 shows that InMASS is initialized from JShell and ready to accept MASS library functions. It also allows simulations to be checkpointed and rolled back without stopping and recompiling the program[3].

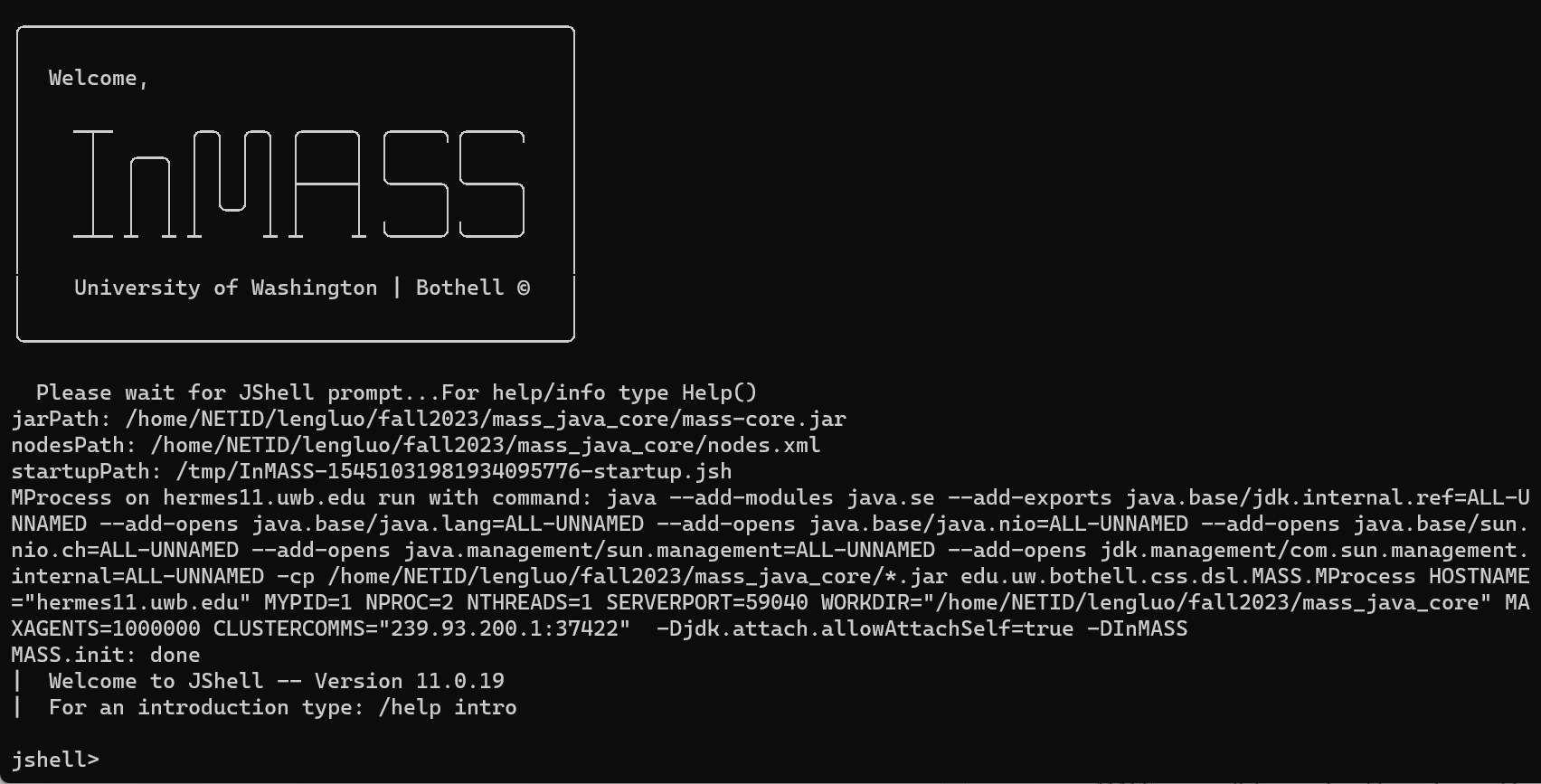


Figure 1: InMASS application

### **1.3 Cytoscape**

Cytoscape is an open-source Java platform used by data scientists and academics from a range of fields for the display and analysis of scientific graphs including biological networks[4]. It offers a user-friendly interface for data import/export and visualization while supporting a variety of data sources and file types. Users can adjust network appearances and highlight particular patterns thanks to customization capabilities. It is adaptable and expandable using plugins, which are controlled by the App Manager and created by users or outside developers[5]. Figure 2 displays the InMASS user panel as integrated within Cytoscape. On the application's left side, the InMASS user panel is featured, enabling users to link Cytoscape with Dslab machines running InMASS applications and to configure the visualization layout. On the application's right side, the visualization of agents and places from the InMASS application is presented.

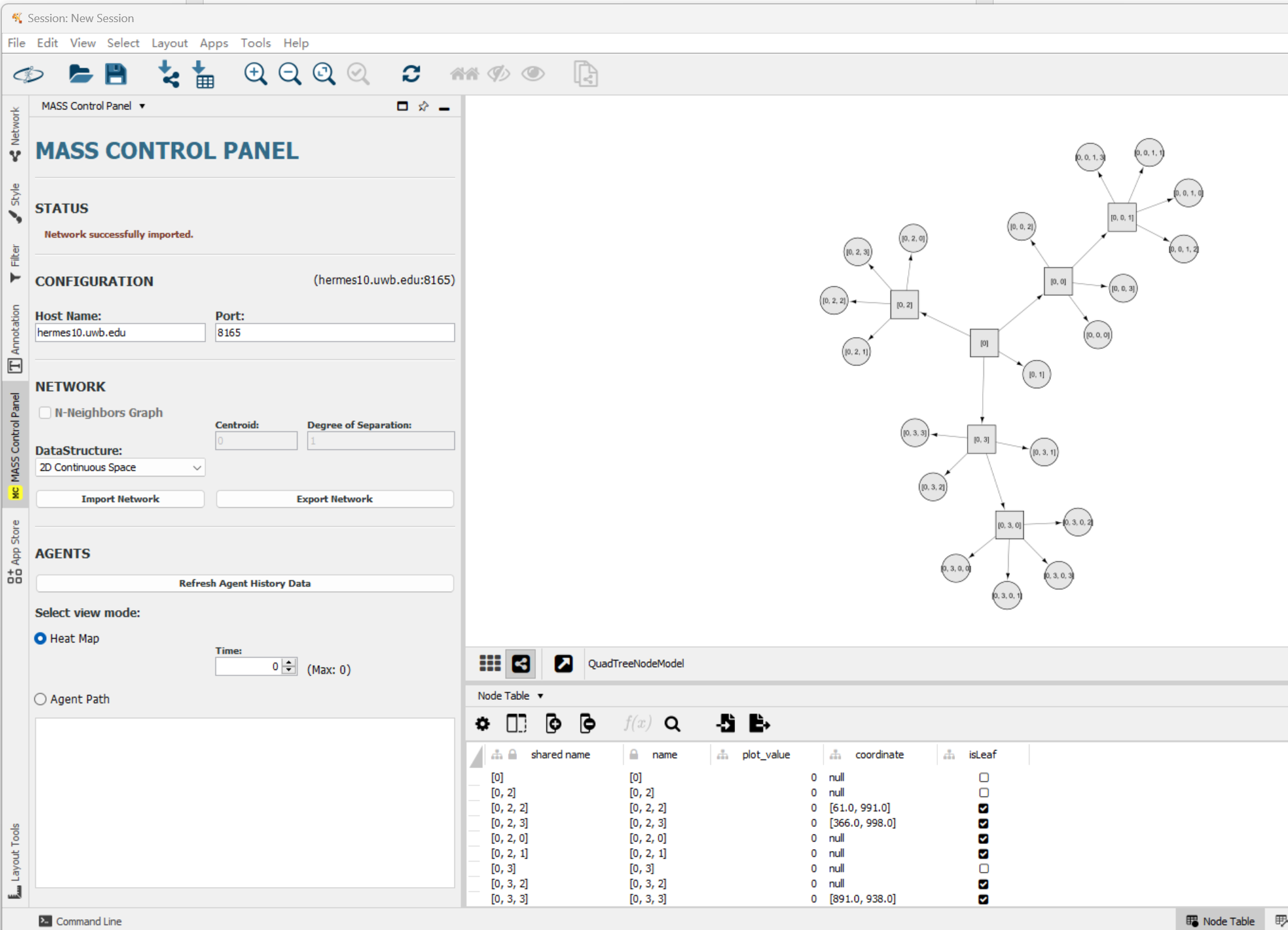


Figure 2: MASS-Cytoscape

### **1.4 Web menu**

The WEB GUI serves as an essential tool for users to monitor their cluster's performance in real-time. It includes a cluster monitoring panel for tracking the status of each machine, with indicators to show if a machine is running or has been terminated unexpectedly. The interface also allows users to monitor the functional calls made to the cluster, an extension of the checkpoint/rollback feature. This feature provides the user with a view of all calls along with their function IDs, aiding in identifying the specific steps for potential rollbacks. The rollback operation is also simplified for a streamlined user experience within InMASS[5]. Figure 4 showcases the primary interface of the WEB GUI, offering essential insights into cluster performance for users. The interface's upper area features a cluster monitoring panel, providing real-time updates on each cluster machine's status. The panel below presents the functional calls made by users to the cluster. This functionality builds upon the checkpoint/rollback feature, allowing users to see all made calls along with their associated function IDs.

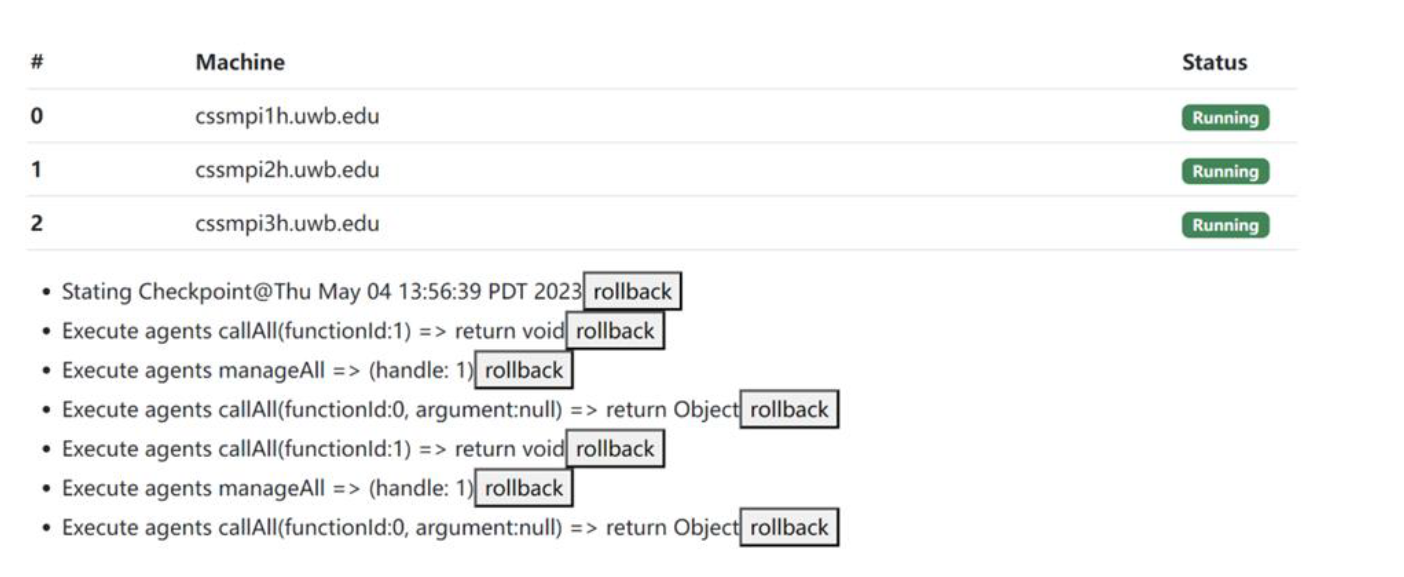


Figure 3: screenshot of current Web GUI

## **2.Goal**

My project aims to enhance and enrich the functionalities within the MASS framework. The five key objectives include:

1. **Enhanced Agent Visualization:** Enrich the visualization of agents within the MASS framework. Instead of representing agents as simple dots on a vertex, the goal is to visualize them with their IDs. It is also necessary to visualize agents within a quad tree-based 2D space to see their interactions and distribution. This enhancement will help users better understand how agents travel between places.
2. **Octree / 3D Space Visualization:** Allow MASS to construct an octree for 3D space representation and then project this visualization in Cytoscape. This improvement introduces templates and insights for a data structure not present in the current application.
3. **Graph/Tree Construction and Agent Population with Cytoscape:** Allow users to interactively construct graphs/trees, populate agents over these data structures, and transfer them to MASS directly within the Cytoscape interface. This approach can make simulations more intuitive and customizable.
4. **MASS Computation Directed from Cytoscape:** Allow users to initiate agent migration/computation and implement agent-based algorithms directly within the interface. This feature should streamline the MASS computation process.
5. **Integrated GUI:** The goal is to integrate InMASS, Cytoscape, MASS, and the web menu into one single website. Such integration will simplify the workflow and enhance usability.

## **3. Quarter’s Achievement**

I've made progress and key developments in my work with MASS and related applications. The following outlines my major achievements

1. I enhanced the usability of mass-cytoscape, ensuring compatibility with both Windows and MacOS platforms.
2. I reimplemented the InMASS benchmark application for the closest pair of points problem using smart agents. Vishnu introduced smart agents and places, where smart agents facilitate automated agent navigation, allowing users to focus on application-specific actions without manually defining agent movements. This change improved performance in finding the closest pair of points within a Quadtree by leveraging automated agent movement, complemented by smart places designed to support this navigation.
3. I improved the quadtree agent visualization in MASS-Cytoscape. Previously, the quadtree visualization inaccurately represented agents' coordinates in 2D space. By extending Cytoscape plugins to accurately calculate and display each agent's coordinates, I fixed this issue.
4. Despite significant modifications by previous students, there was no conclusive evidence of performance enhancements from using multiple machines. My benchmarking results demonstrate the benefits of distributed computation in the current InMASS application. I also noted a limitation where the application crashes beyond four machines, a detail I've documented as a starting point for future investigation. Figure 4 showcases the benchmarking results for the quadtree algorithm's performance in identifying the closest pair of points. The benchmark compares three scenarios: one using 32,769 points distributed across three machines, another with the same number of points but across two machines, and a final scenario using a single machine. The outcomes reveal that using three machines is 2.1 times more efficient than using just one, while using two machines is 1.98 times more efficient than a single machine. The minor efficiency improvement of using three machines over two is caused by the relationship between the map's size and the number of points.

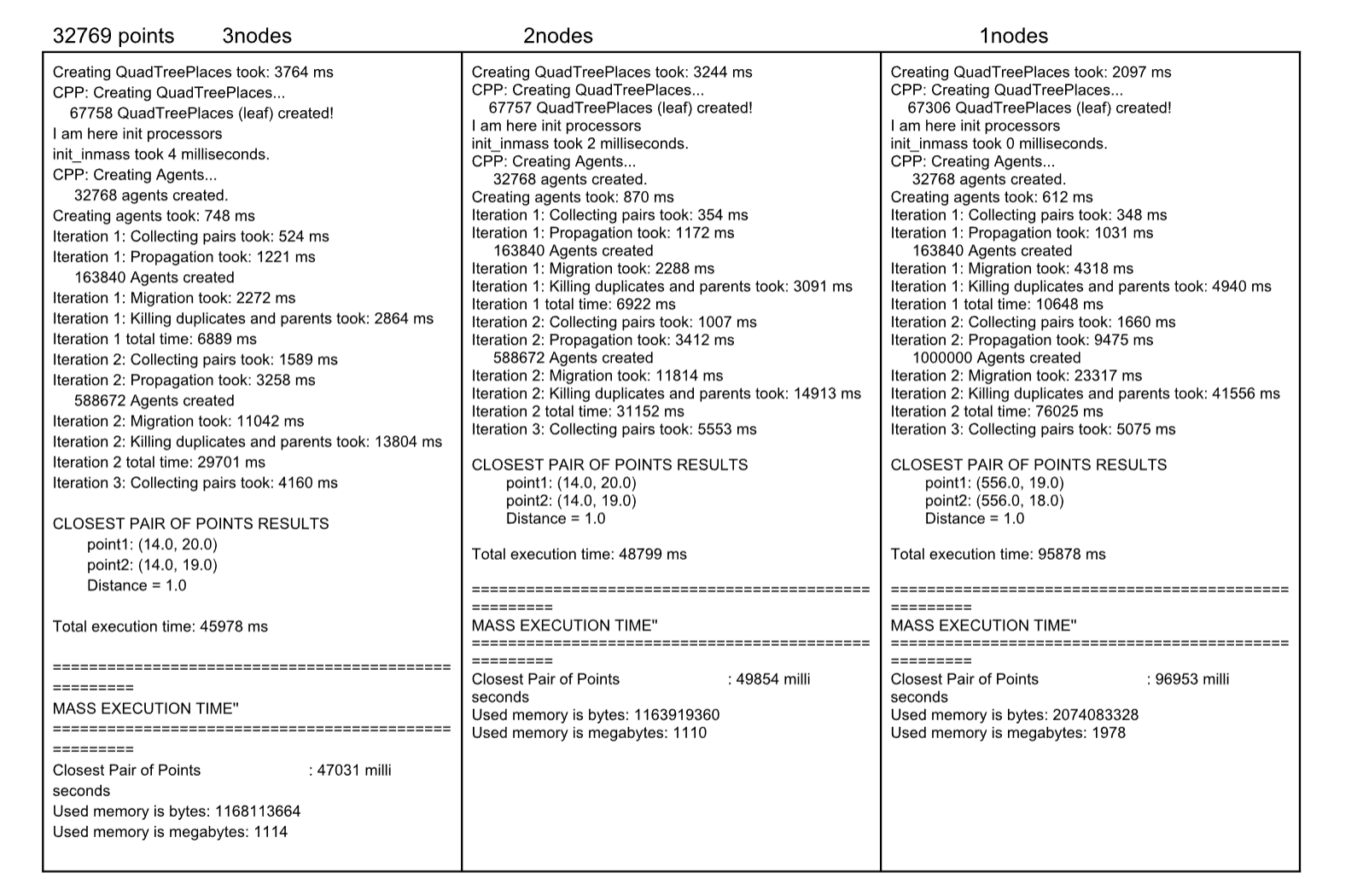


Figure 4: Quadtree CPP performance evaluation

1. I experimented with Cytoscape.js and finalized the design for user interface integration. Initially, I proposed an all-in-one web application for performance monitoring and data visualization, requiring a different technical stack. After familiarizing myself with Cytoscape’s OSGI framework, I recognized its potential for serving as a comprehensive user interface through its modular, plugin-based architecture. Table 1 provides a summary of the plugins created to facilitate InMASS simulation visualization in Cytoscape.

|  |  |
| --- | --- |
| Name of Plugin | Usage |
| 3D-Visulization | Renders agents and places in 3D space |
| export-network | Sends Cytoscape networks to MASS |
| import-agent | Retrieves and store agent history information |
| import-network | Simplifies in-memory network from MASS and rebuilds in Cytoscape |
| mass-agents | Creates InMASS-Control Panel and integrates other plugins |
| mass-cli | Provides a standard command line interface in Cytoscape |

Table 1: List of Cytoscape Plugins and Their Usage

1. I implemented 3D space visualization in MASS-Cytoscape using OpenGL for rendering. OpenGL’s open-standard specification facilitates 2D and 3D graphics programming without the need for direct installation. However, the absence of OpenGL drivers in the latest MacBooks presents a performance challenge due to rendering at the translation layer. Addressing this to better support MacOS users in DSLab remains a priority. For visualization, green cubes indicate a node within the octree, with heatmaps visualizing agent density. At the simulation's end, selecting alive agents shows their movement paths through a coordinate list, although accurately calculating these in 3D space presents challenges I aim to overcome in future developments, which will allow users to see the complete agent path in 3D space. Figure 5 displays 5x5x5 cube, illustrating the 3D visualization of an octree data structure. Each small cube with in the 5x5x5 structure represents a distinct coordinate point in 3D space.

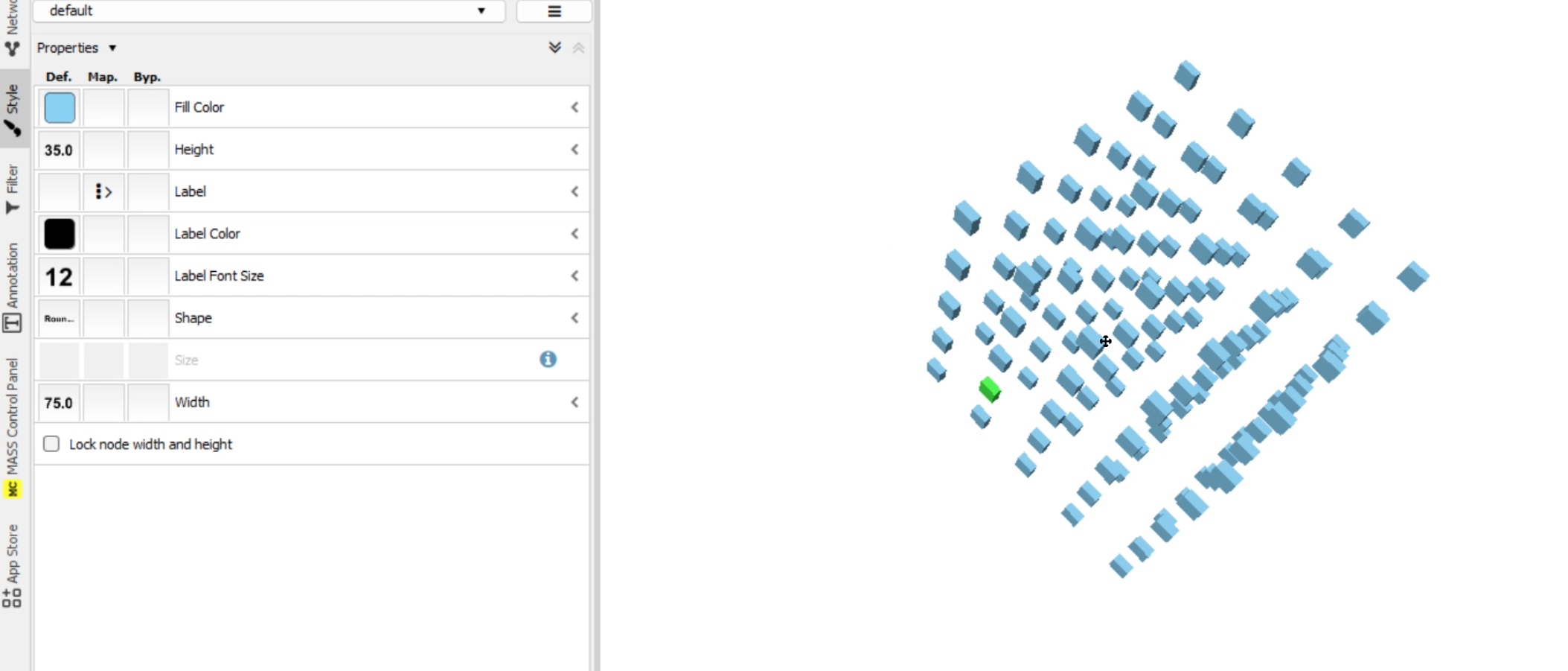


Figure 5: 3D space visualization in Cytoscape

1. I integrated a Command-Line Interface (CLI) into Cytoscape, enabling users to connect to lab machines using Cytoscape's built-in terminal. Cytoscape comes with a built-in command-line interface that lets users execute Cytoscape-specific commands. I expanded this plugin into a standard command-line interface, allowing users to access lab machines directly. Figure 6 presents the overall components of the Cytoscape CLI plugin



Figure 6: current Cytoscape CLI plugin

1. I integrated the web GUI into Cytoscape using its built-in browser. The previous setup was limited, allowing users to only monitor the machine hosting the web server. This posed a significant issue for applications running on lab machines, which lack a graphical display device for viewing the web GUI. To address this, I developed a Cytoscape plugin that establishes an SSH tunnel between my laptop and the lab machines. This tunnel is designed to forward requests from a designated local port on my laptop to the port where the web service is active on the school machine. Consequently, the web server processes any incoming requests and returns responses, including HTML, CSS, and JavaScript, through the tunnel.This built in browser will then renders the web application’s frontend on my screen. Figure 7 demonstrates how the web GUI appears when viewed through Cytoscape’s integrated browser.

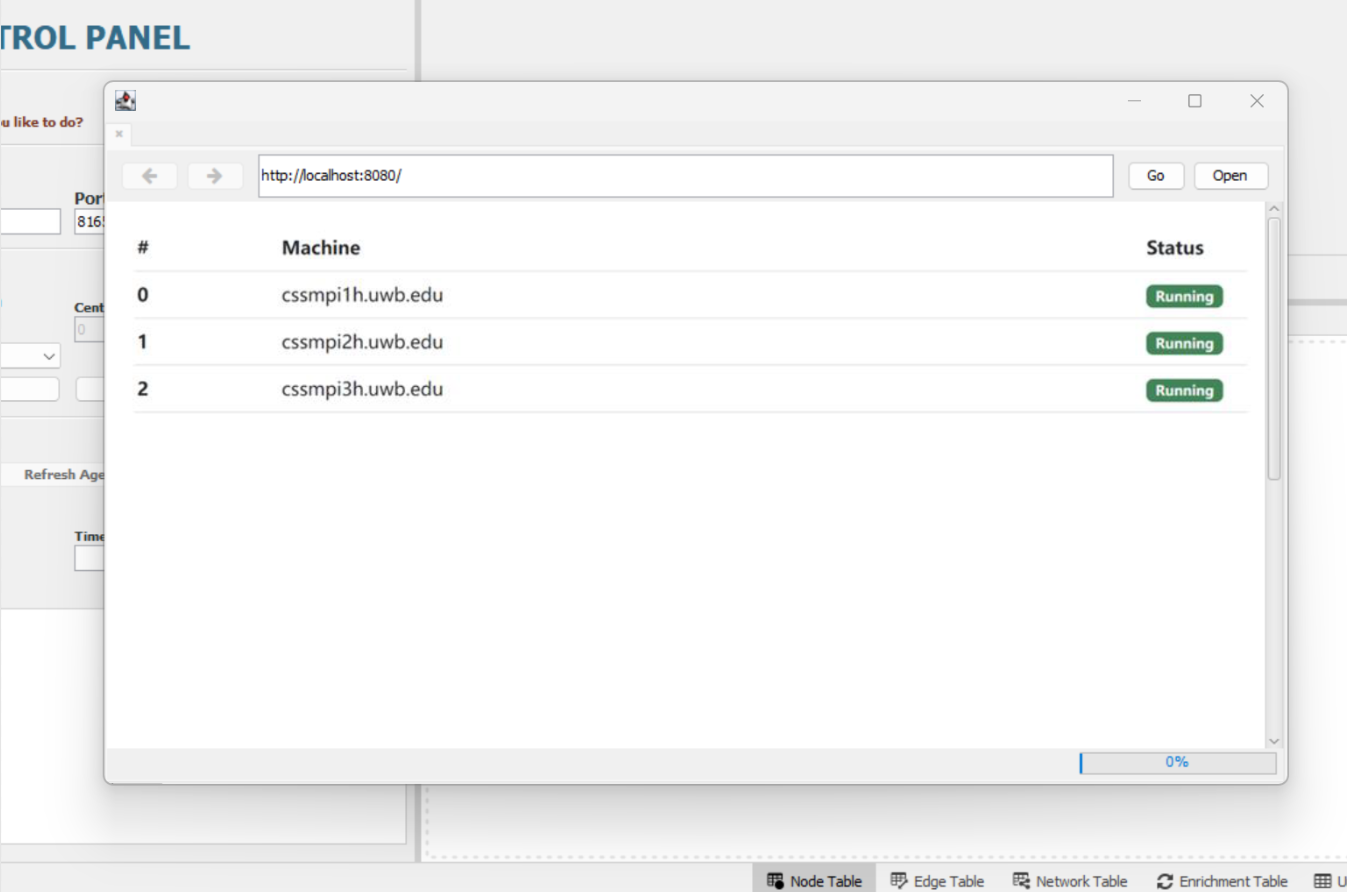


Figure7: integrated Web GUI with cytoscape’s built-in browser

## **4. Plan for Spring 2024**

This plan outlines the objectives and tasks for Spring 2024, focusing on wrapping up all features including mass-cytoscape, the inMASS benchmark application, and the web GUI. Additionally, I will design and conduct user tests and questionnaires for DSLab members for my integrated application. Besides the tasks mentioned above, I will complete my capstone whitepaper as partial fulfillment of the graduation requirements.

|  |  |  |
| --- | --- | --- |
| **Quarter** | **Weeks** | **Tasks** |
| Spring 2024 | 1 | * Present my draft of whitepaper to Prof. Fukuda for review. This draft will include sections on the introduction, description of existing similar applications, description of previous students' work, solution overview, detailed implementation, evaluation of my implementation, conclusion, and references * Discuss with Prof. Zolyomi and Prof. Dimpsey to arrange the schedule for the final defense * Incrementally improve my project |
| 2 | * Incrementally improve my project * Prepare necessary materials for conducting the user tests, including test data file, instruction, questionnaires, and compiled program |
| 3 | * Deliver user tests to Dslab members at the beginning of the week * Collect feedback and make necessary changes to the project |
| 4-10 | * Prepare for whitepaper and final defense |

## **5. Future work**

This section of the document addresses issues identified during this quarter that were not initially included in my project plan. Consequently, I will outline them here for the benefit of future students who may wish to address them, or I might work on them myself should time permit.

* The current Web GUI is incomplete and lacks support for displaying function calls such as callAll or manageALL, which are crucial for the inMASS benchmark application when it operates across multiple machines. Implementing this functionality is necessary within both Mprocess and MassBase. Additionally, the capability to rollback user operations appears to be unfinished.
* Enhancing the code structure of the Mass-Java-core’s InMASS branch is important. Nasser's implementation was removed due to maintenance challenges, yet remnants of his code still persist in the repository. Furthermore, Daniel's contributions are entirely absent from the project. At present, the only method to access his work involves reviewing the commit history.

## **6. Summary**

During this term, I have made significant steps in enhancing the mass-cytoscape project, focusing on improving compatibility, functionality, and user experience. Notable achievements include ensuring mass-cytoscape operates seamlessly on both Windows and MacOS platforms, reimplementing the InMASS benchmark application with smart agents to improve performance, and enhancing quadtree agent visualization for accuracy.

In addition to these advancements, I streamlined the process for users to utilize the applications, making it more intuitive and user-friendly. I also conducted benchmarking to validate the benefits of distributed computation and documented a critical limitation for future investigation. Further advancements were made in user interface integration, leveraging the OSGI framework for a comprehensive, modular design. I introduced 3D space visualization using OpenGL, despite facing performance challenges on MacOS, and expanded user control by integrating a Command-Line Interface for direct lab machine access. Additionally, I overcame previous limitations by integrating the web GUI into Cytoscape, enabling effective cluster monitoring and interaction.

Looking ahead to Spring 2024, my focus will shift to finalizing features, conducting user tests, and completing my capstone whitepaper as part of the graduation requirements. Documented issues such as the incomplete web GUI and the need for code structure enhancement in the Mass-Java-core’s InMASS branch have been outlined for future attention, marking a path for continued improvement and exploration.

## **Appendix**

Repo URL:

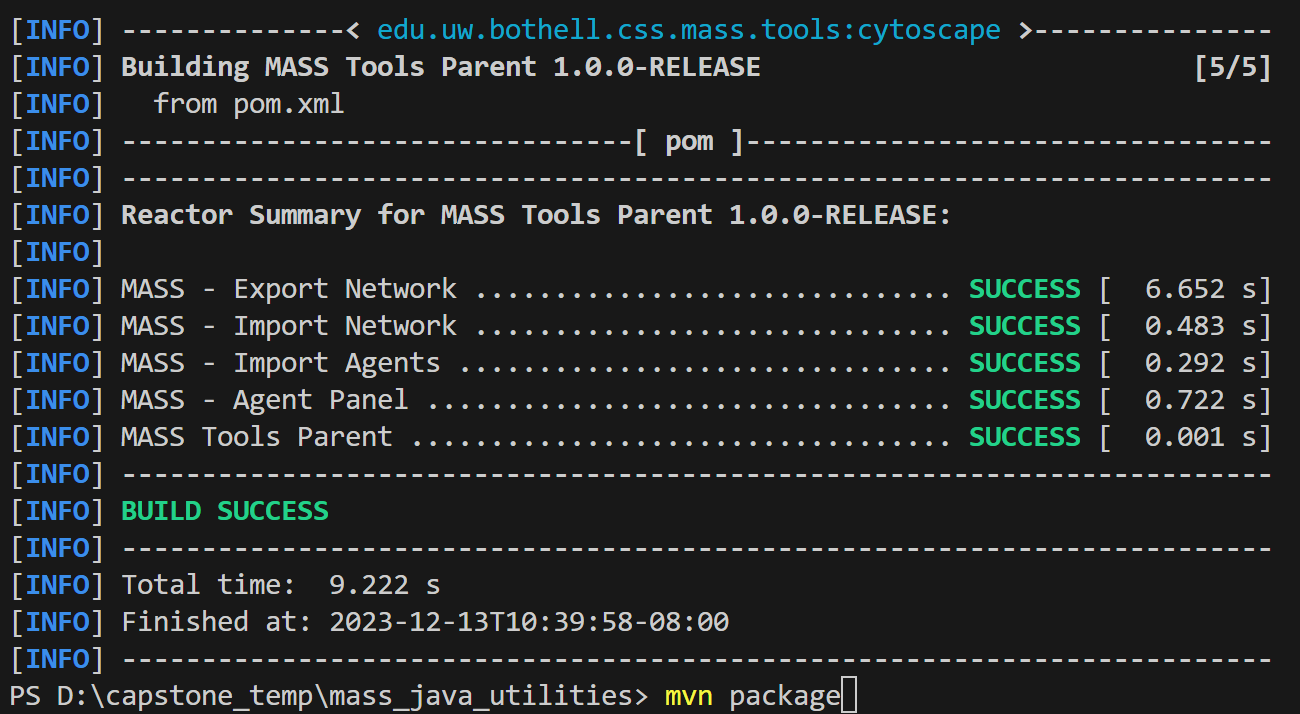
**Mass-cytoscape**: <https://bitbucket.org/luoleng/mass-cytoscape/src/master/>

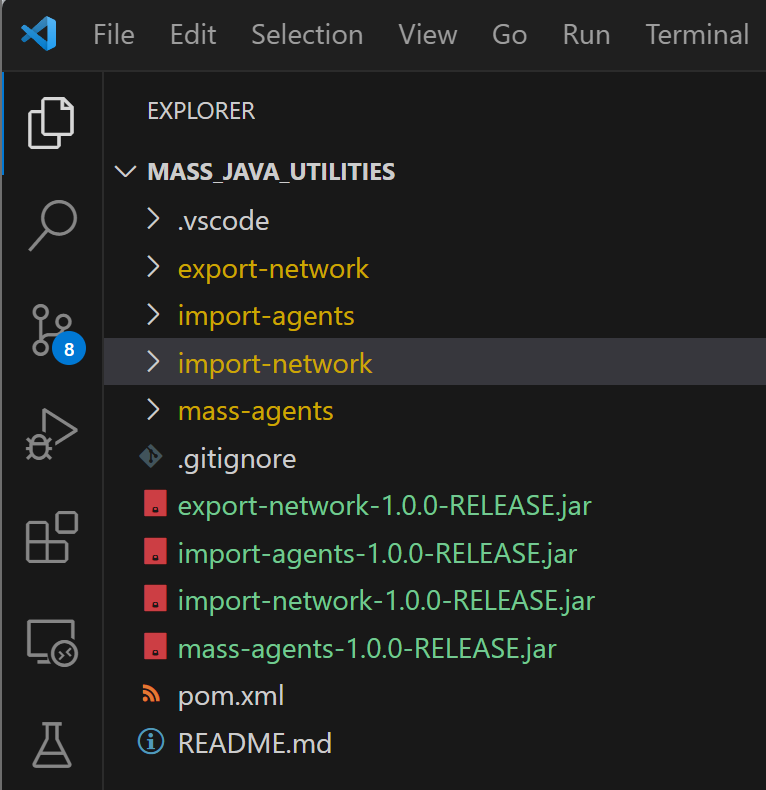
**Mass-java-core**: <https://bitbucket.org/mass_library_developers/mass_java_core/src/c6706f414e256111223baebe11253f4a11b05a73/?at=Luo%2FinMASS>

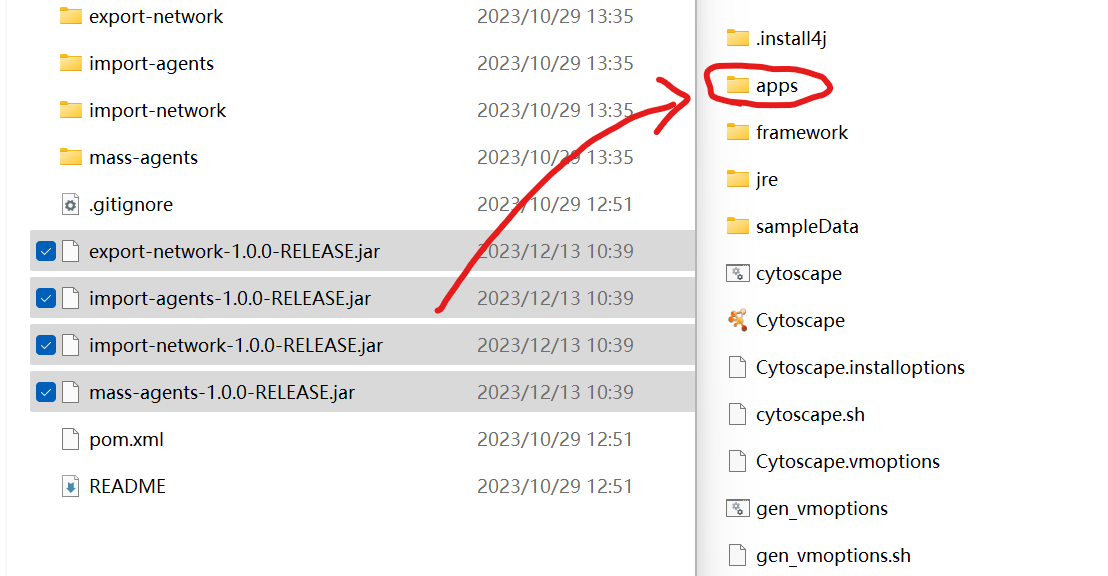
A1. Cytoscape Installation

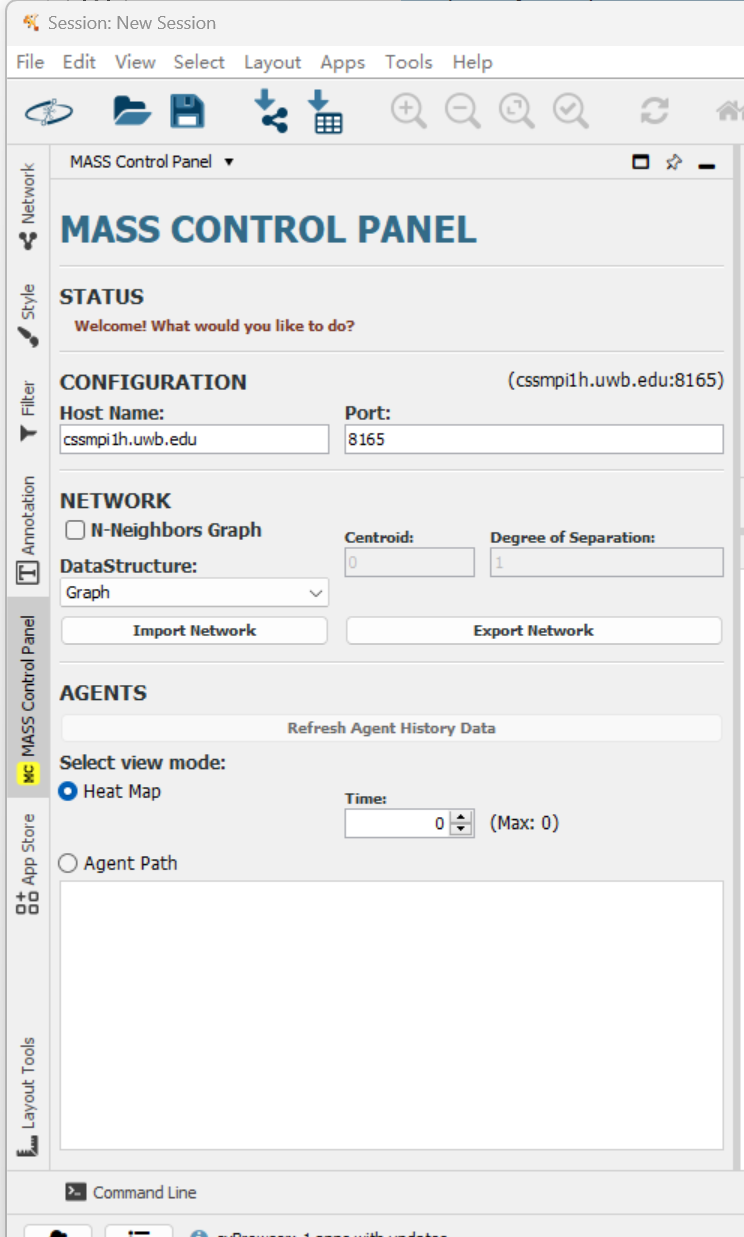
1. Download and install Cytoscape:<https://cytoscape.org/download.html>
2. Download JDK 11 and set JAVA\_HOME environment
3. we use Cytoscape version 3.10.1 which supports both Windows and Mac with M chip

A2. Mass-Cytoscape Installation

1. Download MASS-Cytoscape plugins from Bitbucket. Please download the mass\_java\_utilities repo from Bitbucket. This project is based on [master branch](https://bitbucket.org/luoleng/mass-cytoscape/src/master/)
2. Compile and install MASS-Cytoscape plugins
   1. Use IDEA or terminal to compile the utilities package: mvn package
   2. Run command to copy jar files from each component to root
      1. For Windows: Get-ChildItem -Path .\export-network\target\\*.jar, .\import-agents\target\\*.jar, .\import-network\target\\*.jar, .\mass-agents\target\\*.jar -Recurse | Copy-Item -Destination .
      2. For Mac: find ./export-network/target ./import-agents/target ./import-network/target ./mass-agents/target -type f -name "\*.jar" -exec cp {} . \;



* + 1. Move Jar files to Cytoscape app folder
    2. Reopen Cytoscape. The MASS plugins have been installed in Cytoscape. MASS control panel can be found



A3. InMASS Compilation

1. Download the MASS library. Download the mass\_java\_core package from BitBucket. In this project, we finished the work using InMASS branch.
2. Login to the virtual machine
3. Clone the repo
4. find the mass\_jave\_core package, and compile the package: mvn -DskipTests package
5. Then copy jar file to the root: cp ./target/mass-core.jar .
6. Make sure that SSH keys are generated: ssh -keygen -t rsa -f ~/.ssh/id\_rsa and cat ~/.ssh/id\_rsa.pub >> ~/.ssh/authorized\_keys
7. Try to login different machines without being asked to provide password:

ssh machine\_you\_are\_using.wub.edu

1. Setup nodes.xml before running InMASS

<nodes>

<node>

<master>true</master>

<hostname>{machine}.uwb.edu</hostname>

<masshome>/home/NETID/{your NET ID}/mass\_java\_core</masshome>

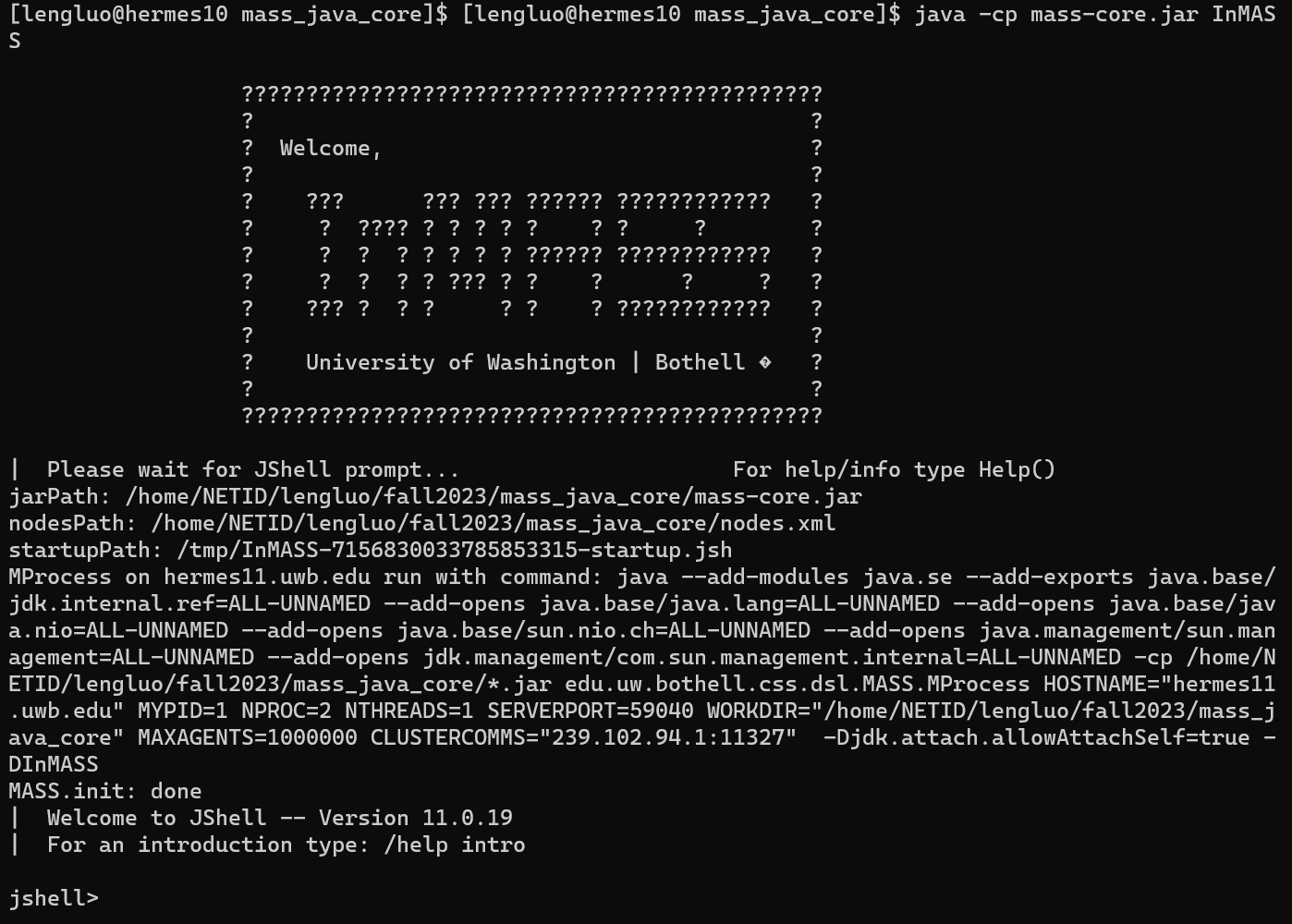
<username>{your NET ID}</username>

<privatekey>/home/NETID/{your NET ID}/.ssh/id\_rsa</privatekey>

<port>{your assigned port}</port>

</node>

</nodes>

1. Run InMASS platform using: java -cp mass-core.jar InMASS
2. Run MASS benchmark application in InMASS:
3. /open /home/NETID/**your net id**/mass\_java\_core/applicationsInMASS/cpp/main.jsh
4. /open /home/NETID/**your net id**/applicationsInMASS/triangle/CountTrianglesGraphMASS.jsh

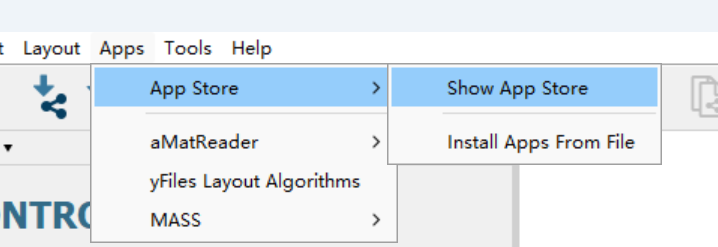
Use /exit to quit

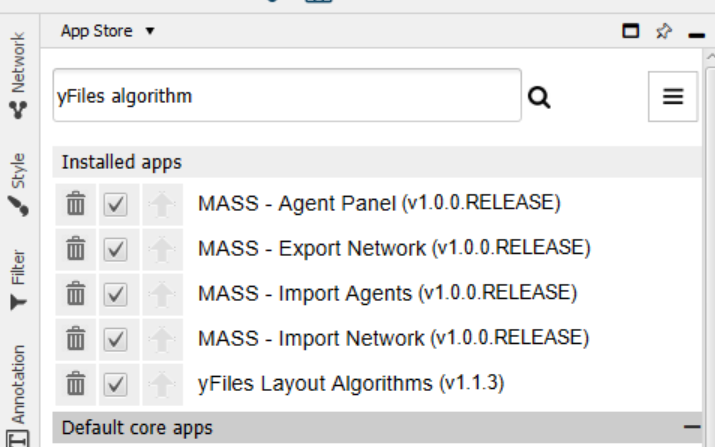
1. Now open Cytoscape and import network

Change Host Name

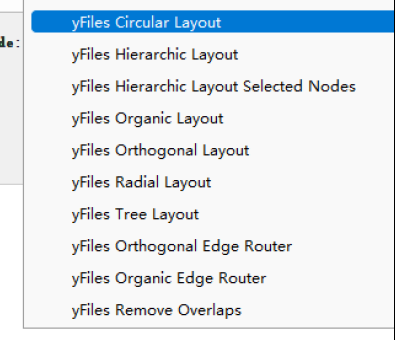
Select the data structure that you are importing

1. install an external plugin (yFiles algorithm) in the Apps -> App Manager

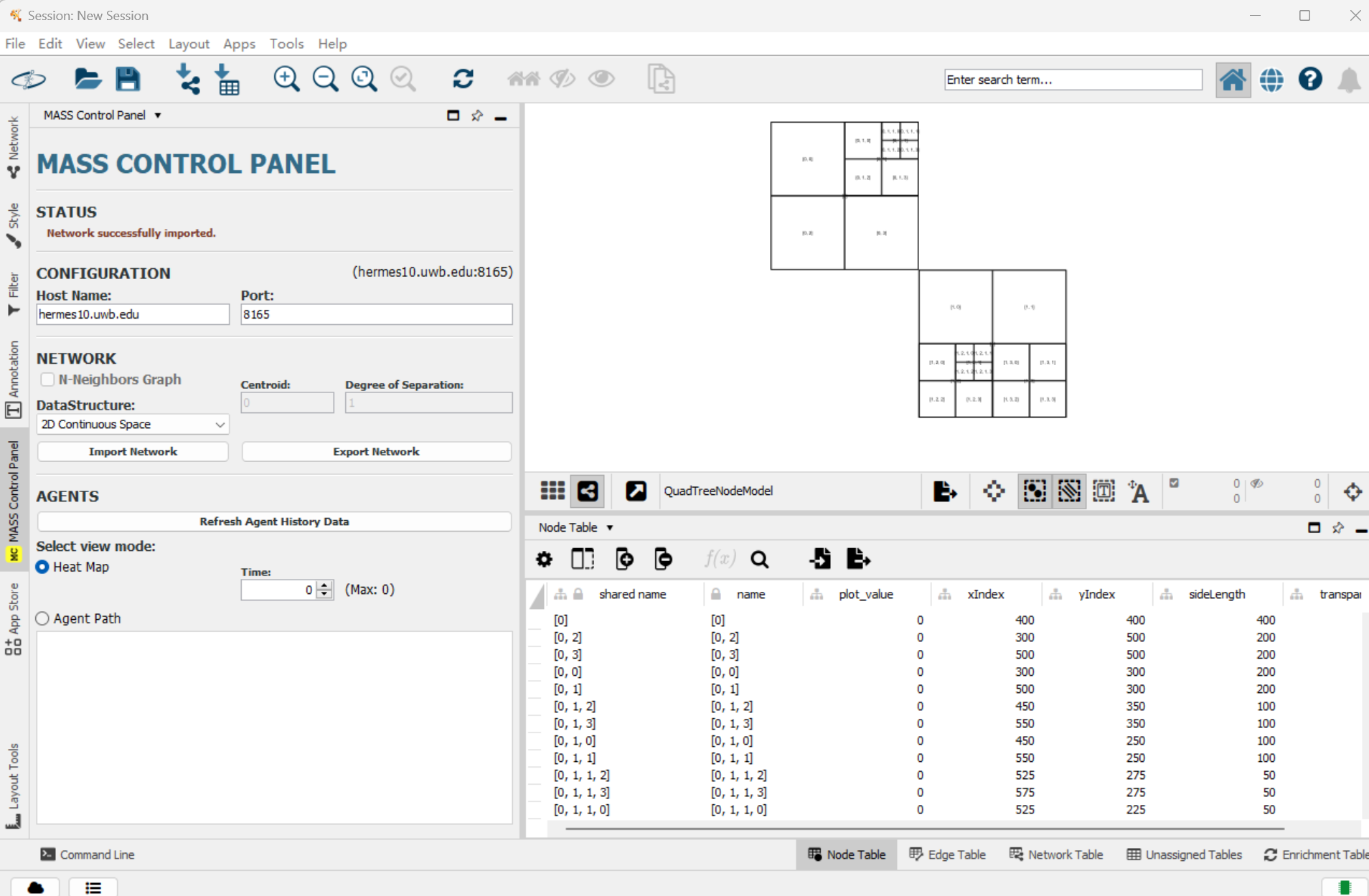




1. After installation select the correct layout to visualize network
   1. default layout for 2D space
   2. Tree Layout for quad tree and binary tree
   3. Circular layout for graph



1. Click Import Network to see visualization



A4. Web GUI Compilation

1. Download the MASS library. Switch branch to yifeiyang/gui
2. Compile the package: mvn -DskipTests package
3. Move the jar file to root: cp ./target/mass-core.jar .
4. Run inMASS application:java -cp mass-core.jar InMASS
5. Run MASS benchmark application in InMASS:
6. Open your browser and visit local 8080 port