**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**

**December 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.

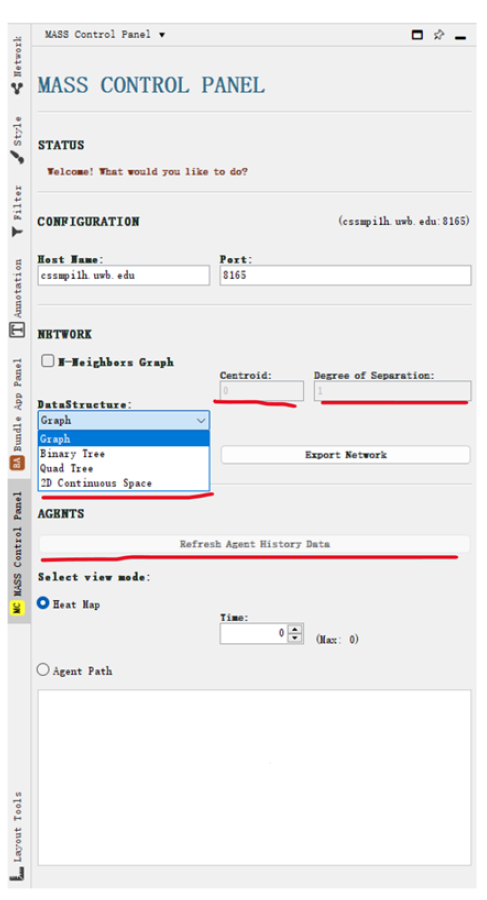


Figure 1: current MASS Control Panel

### **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**

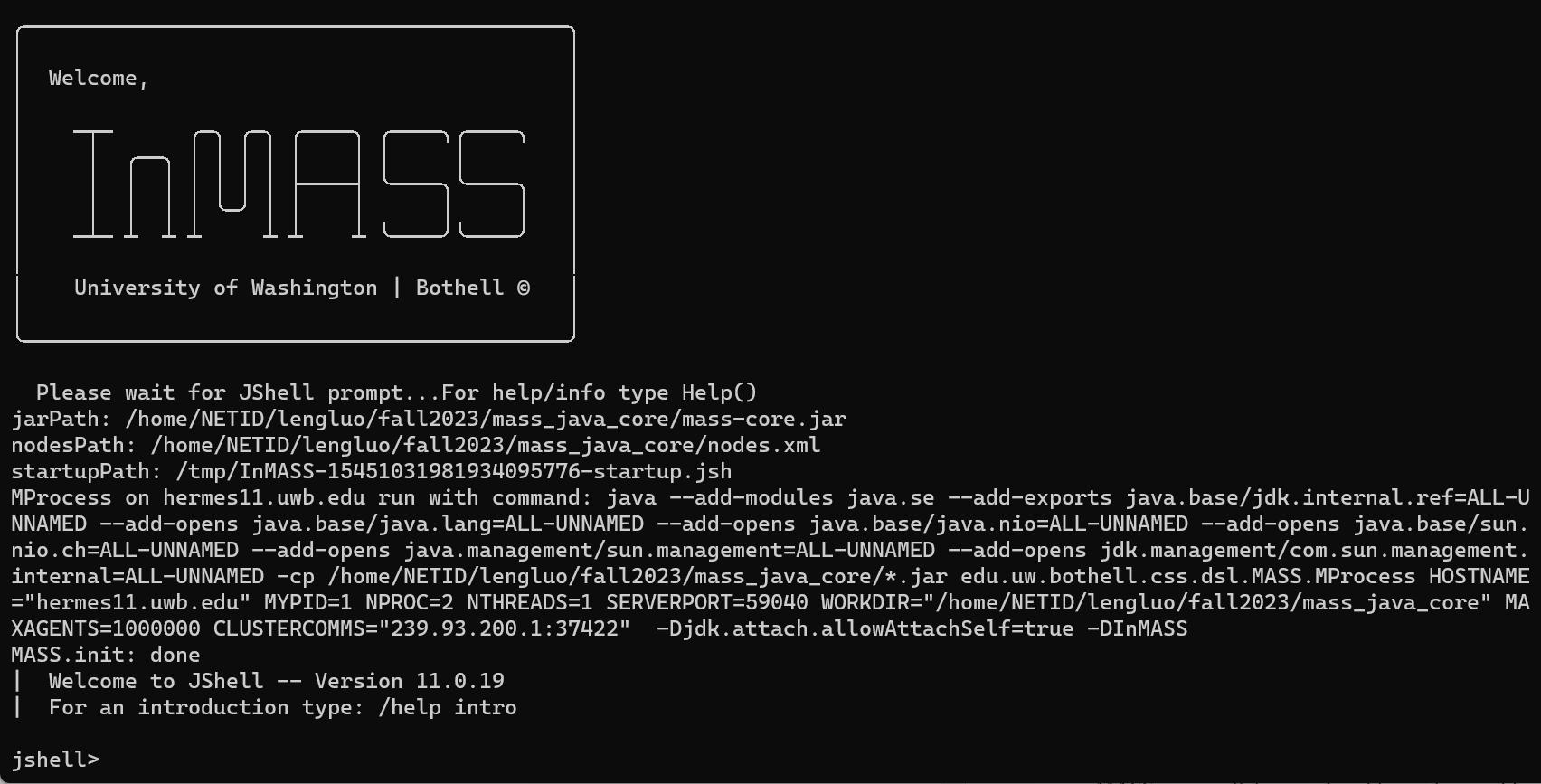
Interactive MASS (InMASS) allows line-by-line MASS simulations through a command-line interface. This is achieved by wrapping Java's JShell interface with additional support classes for functioning in MASS' distributed environment. It also allows simulations to be checkpointed and rolled back without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

Figure 2: 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].

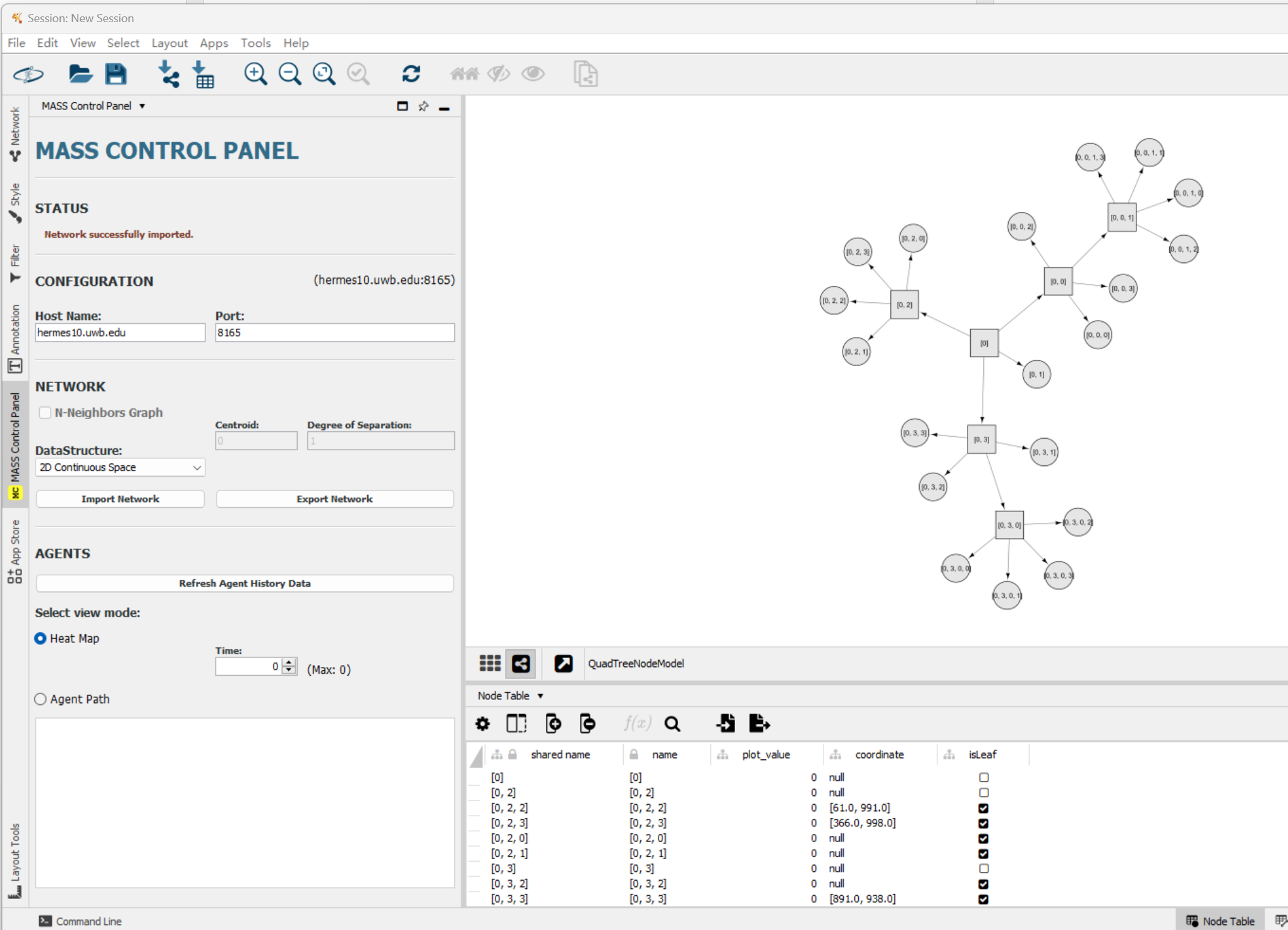


Figure 3: MASS-Cytoscape

### **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].

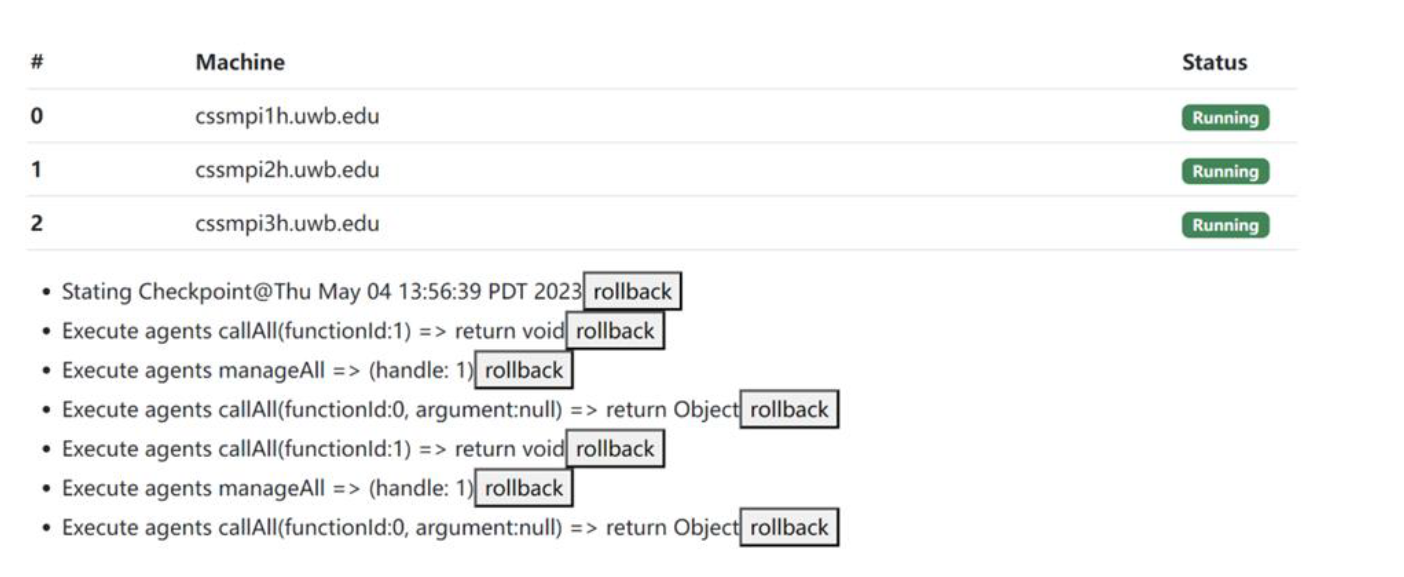


Figure 4: screenshot of current Web GUI

## 

## **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 could 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 would simplify the workflow and enhance usability.

## **3Quarter’s Achievement**

1. Enhanced the usability of mass-cytoscape, ensuring it operates on both Windows and Linux.
2. Became familiar with the functionalities of MASS, InMASS, the MASS-Cytoscape plugin, and the web GUI, gaining important insights for the integration task. The most challenging part was familiarizing myself with the existing codebase and technology stack. While implementing quadtree visualization, I faced several challenges, including MASS compilation issues, problems with InMASS running on multiple nodes, benchmark application execution difficulties, and Cytoscape operational problems. Before I could successfully run the first data visualization created by a previous student, I invested a considerable amount of time in understanding the overall architecture, the application's workflow, and the interaction between different components. Although it was time-consuming, this process turned out to be very valuable. It gave me a good starting point for figuring out how to integrate the applications in the next quarter.
3. Worked with a team member to debug MASS installation issues. The problem was particularly tricky because the code version was the same as the previous year, yet no issues were encountered then. The MASS library is the backbone of all our applications, so the installation failure was quite bothersome. It led me to question if this was the source of all the problems I had encountered.
4. Identified an issue that was preventing inMASS from running on multiple nodes and created documentation for future users. This problem, which had persisted for over two weeks, made it challenging for me to implement quadtree visualization for the program running on multiple nodes. The debugging process involved modifying the project configuration file, updating SSH configurations, adjusting various permissions, and altering the method of output streaming. I summarized the debugging process and compiled an instructional document. This document will help other lab members setting up the InMASS application more smoothly.
5. Implement basic Quadtree agent visualization in MASS-Cytoscape. Due to time constraints, I could only achieve the minimal requirements in this quarter. The existing graph visualization example was a great help in understanding the components of MASS-Cytoscape, boosting my confidence to finish the remaining parts and enhance user experience during the quarter break. Below is a checklist of items I need to address:
   1. The current graph application is limited to accepting only CSV files as input. In contrast, the existing triangle counting program uses TXT files and requires a different format. To address users' needs for a broader range of input file options, I plan to extend the features and provide more examples of input files.
   2. Range search agent visualization is not properly displayed in Cytoscape. I need to debug the binary tree agent history
   3. When too many places and agents are created, all visualizations encounter issues with label overlap. Once I decide whether to continue developing Cytoscape using its JavaScript version or desktop version, I will utilize their existing features to enhance the visualization.
   4. Sometimes, Cytoscape faces freezing issues during the import of agent information. For this problem, I plan to extend the classes for agent history tracking and synchronization among different nodes
   5. Expand the heatmap functionality, which illustrates the number of agents in a place by using various colors to fill that space, to include both binary tree and quadtree structures
   6. Fully display the agent information of range search program within InMASS
   7. Add an output display for quadtree agents and quadtree places within InMASS
6. Made several updates to the current MASS-Cytoscape plugins to enhance user experience
7. I proposed two potential solutions for octree visualization: transitioning Cytoscape to its web version or continuing with Cytoscape Desktop as the primary interface. Cytoscape.js and Cytoscape Desktop are different versions of Cytoscape, each with specific use cases and distinct advantages. Cytoscape.js provides better flexibility for web integration and user interactivity. However, it requires converting the current codebase to JavaScript and re-implementing some plugins. On the other hand, Cytoscape Desktop offers better plugin compatibility, which means our current application wouldn't need major changes. The drawback is its inability to integrate directly into our web GUI, resulting in separate user interfaces for students and developers. To better estimate the cost and difficulty of both development directions, I plan to experiment with Cytoscape.js' framework by implementing a basic 3D space visualization, and then develop a complete design to present at the beginning of next quarter.

## **4Plan for Winter 2024**

|  |  |  |
| --- | --- | --- |
| **Quarter** | **Weeks** | **Tasks** |
| Winter 2024 | 1-2 | Design and propose a significant change for MASS-Cytoscape development,impacting overall data visualization, user interaction, and the integration of InMASS operations with MASS-Cytoscape  Create a snapshot for a stable version of MASS-Cytoscape and InMASS incorporating the changes I made during the break |
| 3-5 | Development of Graph/Tree construction and agent population with Cytoscape |
| 6-9 | integration of InMASS, Cytoscape, MASS, and web menu |
| 10 | Test, debug, and create documentation for updated GUI and workflow.  Identify the user group; create questionnaires and user guide |

## 

## **5.Summary**

I managed to stick to my fall quarter plan and completed the tasks as scheduled. Initially, I was unaware of the 10th week of fall quarter. I was able to start the tasks for the first two weeks of Winter 2024 ahead of time. Consequently, I'll use the first two weeks of Winter to create a snapshot of the stable version of my current development and . I've allocated an extra week for integrating InMASS, Cytoscape, MASS, and the web GUI, pushing the development of Octree visualization to a later date. Given the complexity of implementing 3D visualization, I might not finish the integration task on time. In that case, I'll conduct a design and possibly leave this task for a future student. I will at least implement the octree class in the MASS library.

This quarter, I worked on enhancing the MASS-Cytoscape functionalities, focusing on Quadtree agent visualization. A key challenge was familiarizing myself with the existing codebase, particularly while addressing issues in implementing quadtree visualization and running InMASS on multiple nodes. Through this process, I gained valuable insights into the system's architecture.

Moving forward, my focus for the next quarter will be on further developing these visualizations and achieving a seamless integration of MASS, InMASS, and Cytoscape components. This effort aims to improve usability and workflow efficiency, guided by comprehensive testing and user-focused documentation.

## **Appendix**

Repo URL:

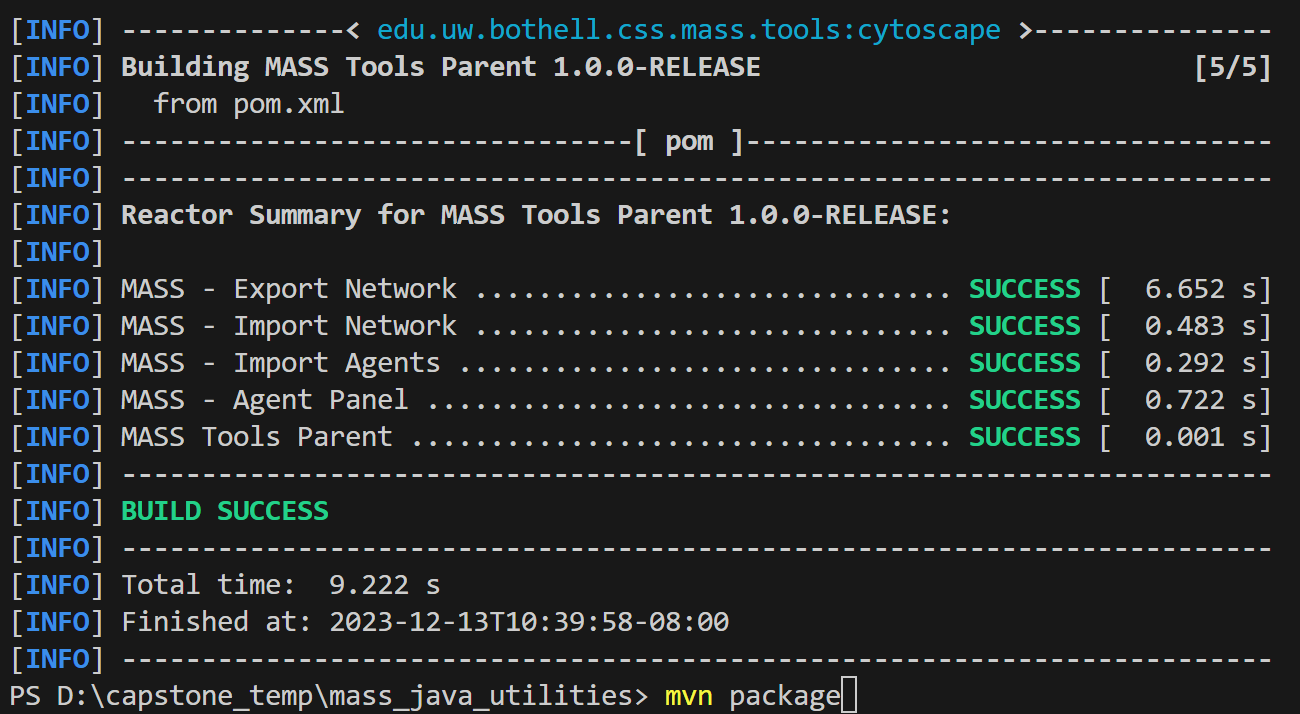
**Mass-cytoscape**: <https://bitbucket.org/luoleng/mass-cytoscape/src/master/>

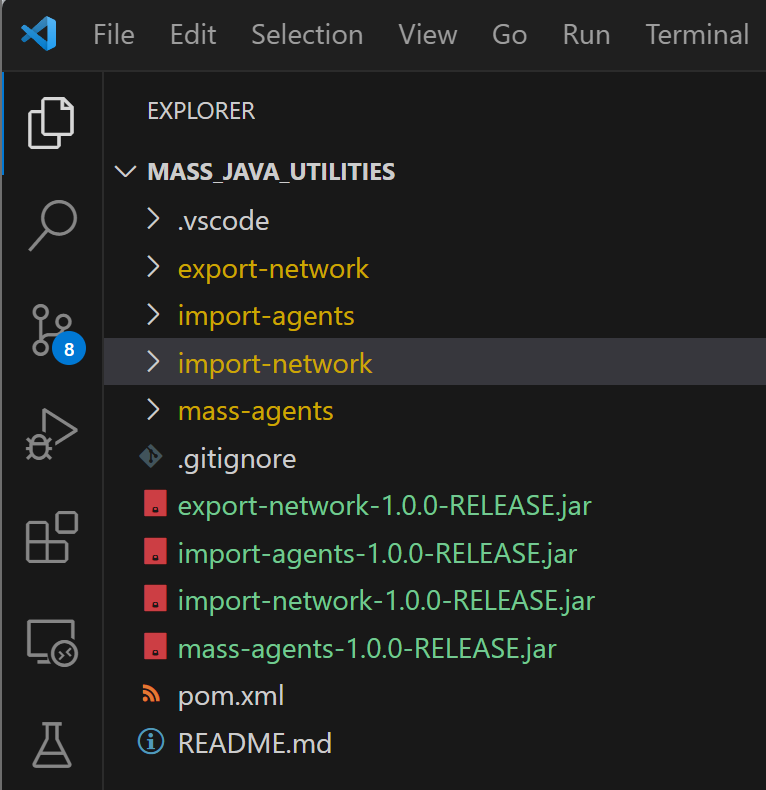
**Mass-java-core**: <https://bitbucket.org/mass_library_developers/mass_java_core/src/c6706f414e256111223baebe11253f4a11b05a73/?at=Luo%2FinMASS>

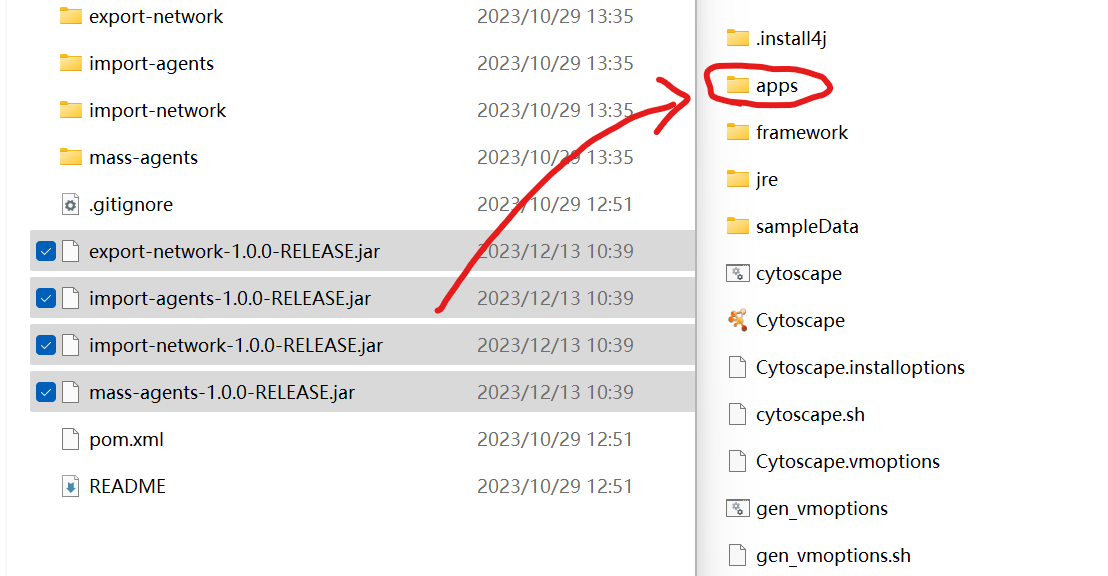
A1. Cytoscape Installation

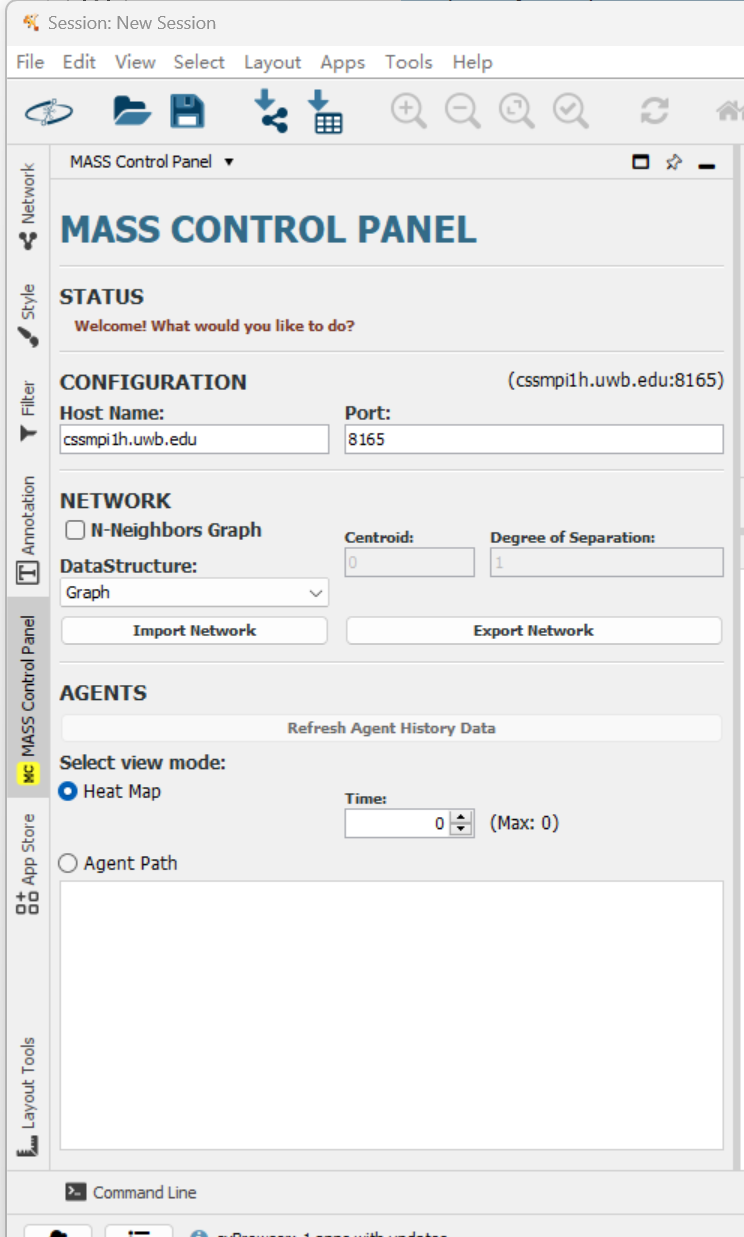
1. Download and install Cytoscape:<https://cytoscape.org/download.html>
2. Download JDK 11 and set JAVA\_HOME environment
3. we use Cytoscape version 3.10.1 which supports both Windows and Mac with M chip

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>

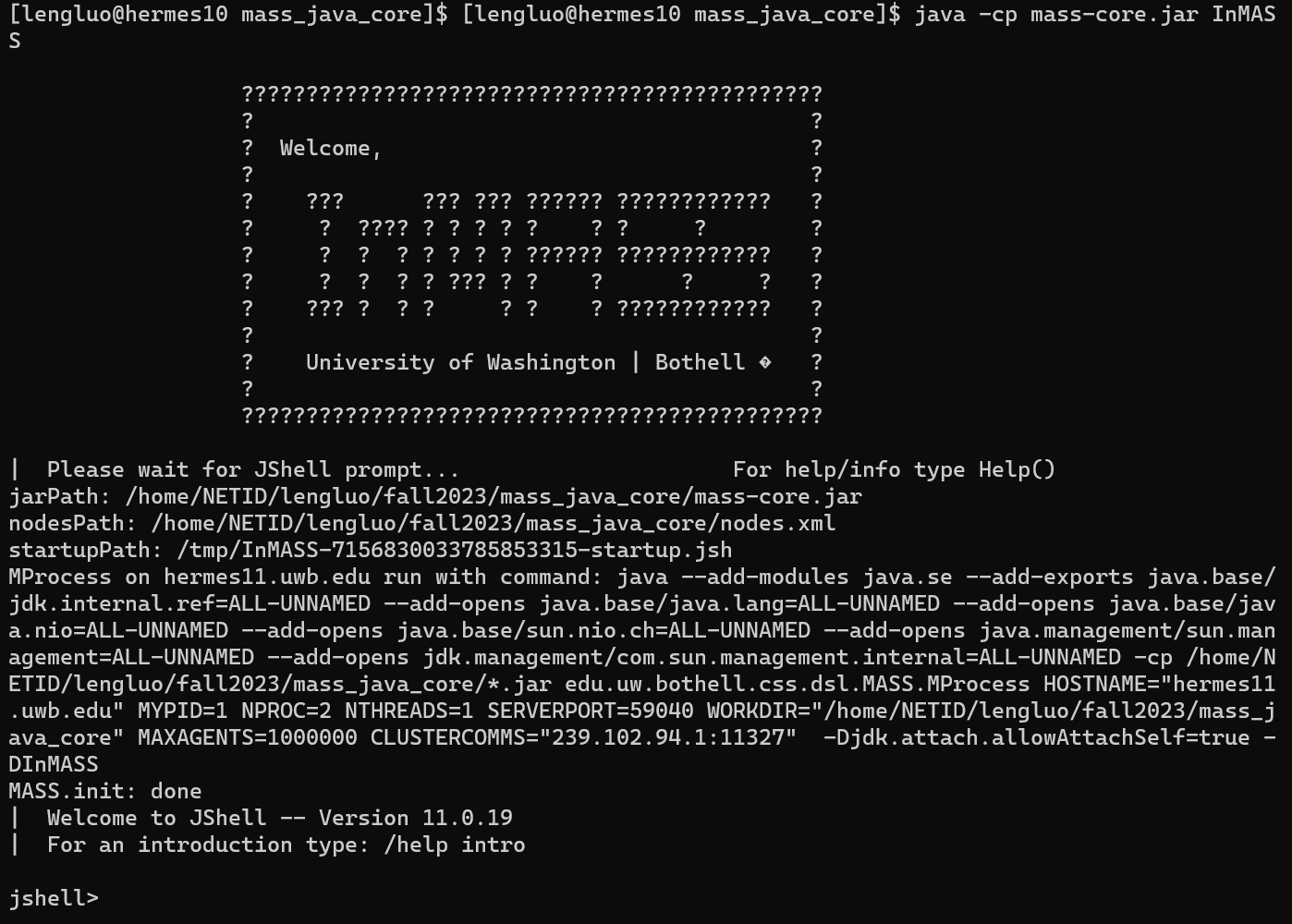
<username>{your NET ID}</username>

<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

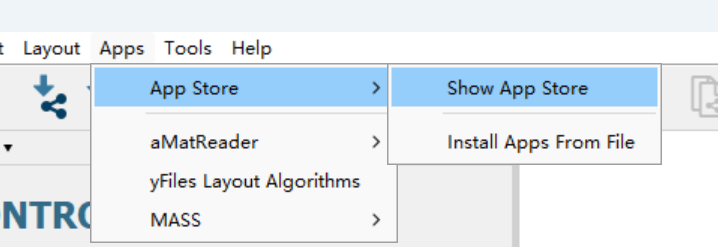
Use /exit to quit

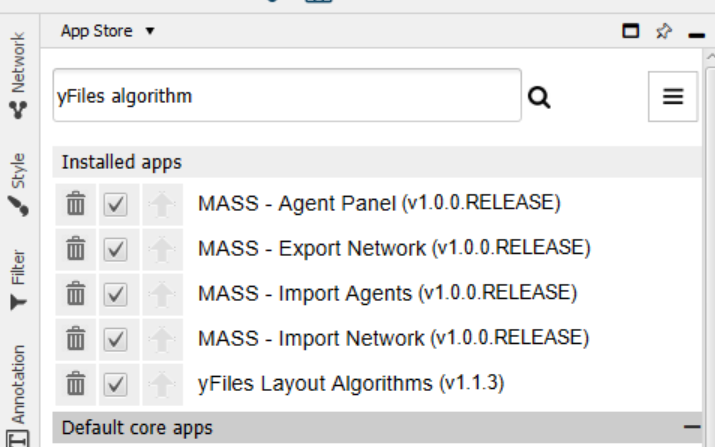
1. Now open Cytoscape and import network

Change Host Name

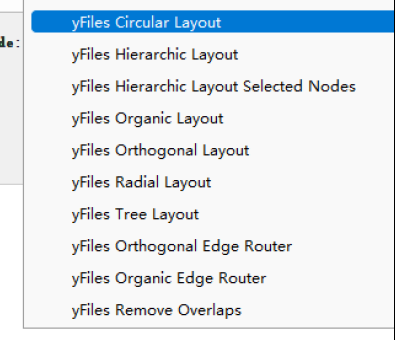
Select the data structure that you are importing

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

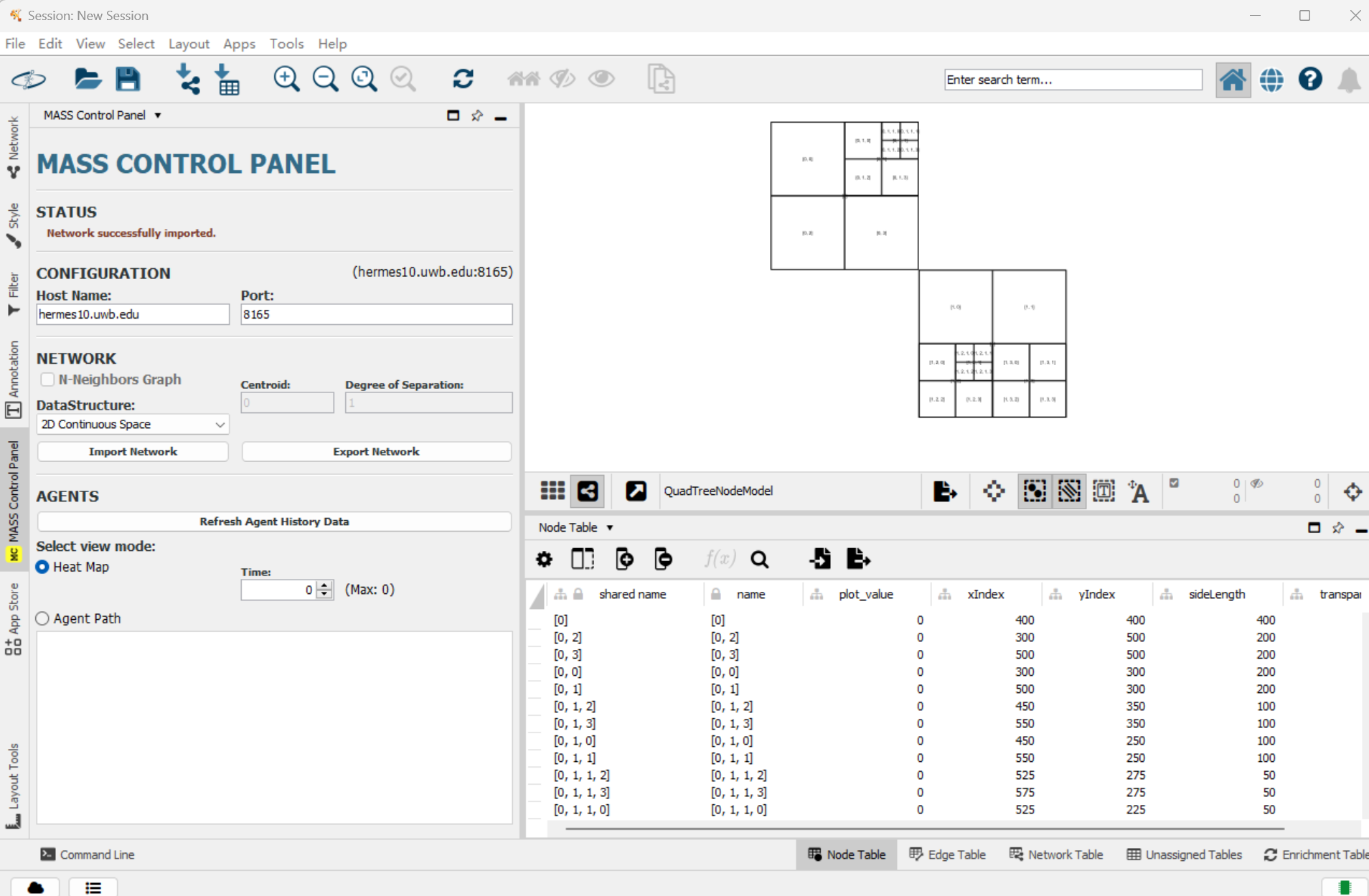




1. After installation select the correct layout to visualize network
   1. default layout for 2D space
   2. Tree Layout for quad tree and binary tree
   3. Circular layout for graph



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