**User Manual  
*“MicroRTS AI System”***

**Authors:**

Justin Homsi - 10373949

Matthew Burr - 10365565

Lucas Ryan

Nicholas Pleass

*Applied IT Project: MicroRTS AI System - Behaviour Trees and Applied Evolutionary Algorithms 2018*  
*Edith Cowan University*

***Date:*** 30-10-18

***Revision:*** 0.1

***Project Supervisors:*** Martin Masek, Peng Lam

[**About this AI**](#_gjdgxs) **3**

[**What is MicroRTS?**](#_30j0zll) **3**

[**System Requirements**](#_1fob9te) **3**

[**How to Use the System**](#_3znysh7) **3**

[Tutorial 1: Set Grid Size](#_2et92p0) 3

[Tutorial 2: Units & The Grid](#_tyjcwt) 4

[Tutorial 3: Saving States](#_3dy6vkm) 6

[Tutorial 4: Loading States](#_4i7ojhp) 7

[Tutorial 5: Evolution](#_1hmsyys) 8

[Tutorial 6: Advanced Evolution](#_1egqt2p) 10

[Tutorial 7: Retrieving the Output](#_279ka65) 12

[Tutorial 8: Traces](#_42ddq1a) 14

[Tutorial 9: Running a trace](#_3dhjn8m) 16

[**Conclusion**](#_tzj49ptssst8) **21**

# About this AI

The MicroRTS AI System provides simple easy to use Graphical User Interfaces (GUIs), that exist in the form of Java executable JAR files allowing the user to do the following:

* Evolve an AI using Genetic Programming algorithms.
* Create States (maps to use within a MicroRTS simulation)
* Run evolutions using parameters specified by the user.
* Create traces (replays of simulations) based on the best individual of a run.

# What is MicroRTS?

MicroRTS is a simple implementation of a Real Time Strategy (RTS) game, designed for AI research.

# System Requirements

In order to run this application, the user must first install Java Runtime Environment (JRE) 8 (or higher) on their computer. The JRE must be installed as the Java JAR files require it to run. The steps to install JRE are as follows:

* Download JRE 8 from the following link:
  + https://www.oracle.com/technetwork/java/javase/downloads/jre8-downloads-2133155.html
* Run the installer and follow the prompts

# How to Use the System

This section will include a series of tutorials on how to use the system.

## Tutorial 1: Set Grid Size

This tutorial will show the user how to set the size of the grid within a state.

* Step 1:
  + Open the MicroRTS\_States\_Traces JAR file by double clicking it
* Step 2:
  + At the top of the window that opens, locate the textboxes labeled “Width:” and “Length:”, at this point, both boxes will have a value of “8” as shown below:



*Figure 1: MicroRTS State Width and Height settings*

* Step 3:
  + Let’s try setting our grid to a size of 24x24. To do this, enter 24 in the “Width” box and press enter, then enter 24 in the “Height” box and press enter.   
    **Note: While doing this, you should notice the grid displayed on the right side of the window changing.**
  + You have now made a 24x24 grid!

## Tutorial 2: Units & The Grid

This tutorial will describe the differences between units in MicroRTS, as well as show the user how to place units on the grid.

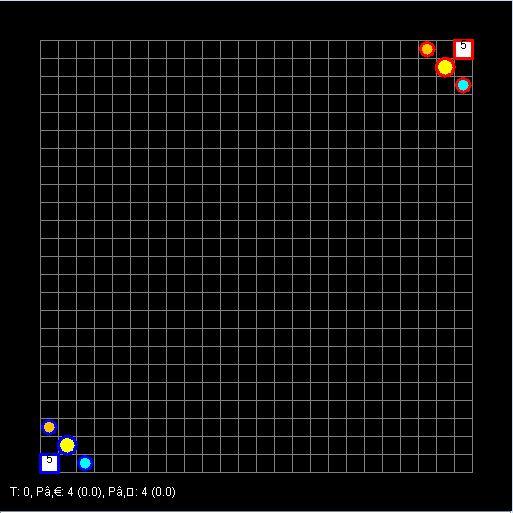
There are seven types of units within MicroRTS, they are: Resource, Base, Barracks, Worker, Light, Heavy, and Ranged.  
For the purpose of this tutorial, we will focus on the Base, Light, Heavy and Ranged unit types.

Listed below are the attributes of each of the unit types (Base, Light, Heavy, and Ranged) as well as a quick description.

* Bases
  + Have 10 hit points
  + Cannot move
  + Cannot attack
  + Can produce worker units
* Light units
  + Have 4 hit points
  + Can move
  + Can attack
  + Deal 2 damage per hit
  + Fastest unit in the game
* Heavy units
  + Have 4 hit points
  + Can move
  + Can attack
  + Deal 4 damage per hit
  + Slowest unit in the game, but hardest hitter
* Ranged units
  + Have 1 hit point
  + Can move
  + Can attack from range (at a distance of 3 squares or less)
  + Deal 1 damage per hit
  + Deals low damage, but is the only unit which can attack from range

Now that you know about the different types of units, let’s add a few to our grid.

* Step 1:
  + Open the MicroRTS\_States\_Traces JAR file by double clicking it if you haven’t already.
  + On the right side of the window, you will notice a grid with a couple of things already in it. The green squares are Resources, and the white squares are Bases. The colored outline around the Bases indicates which team it belongs to.
* Step 2:
  + Click on any empty square in the grid, you will notice a dropdown menu appear. Try placing any unit on the grid by selecting it from the dropdown menu.   
    **Note: Player 0 is Blue, and Player 1 is Red.**
* Step 3:
  + Now that you have placed a unit on the grid, lets remove a unit from the grid.
  + Click on a unit already in the grid (such as one of the Bases), and select “Remove Base” from the dropdown menu.
* Step 4:
  + Now that you know how to place and remove units from the grid, use what you have learned to make the following:



***Light unit – Small orange circle***

***Ranged unit – Small blue circle***

***Heavy unit – Large yellow circle***

**Base – White square**

*Figure 2: 24x24 3-vs-3 map*

* You have now made a state!

## Tutorial 3: Saving States

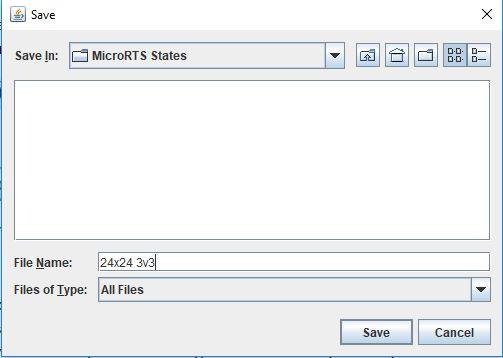
Now that we have created our state, we need to save it. Doing so will allow us to keep the state for later use in the evolution tutorials.

To save the state you have created, follow these steps:

* Step 1:
  + Above the Width and Height textboxes we have previously used, locate the “Save” button, as shown below:



* + Click “Save”
* Step 2:
  + Upon clicking “Save” you will be presented with a dialog box asking you where you would like to save and what you would like to name the file.
  + Navigate to the location you want to save your state in, this can be anywhere you like, however for ease, I will be saving to a folder on my Desktop named “MicroRTS States”.
  + Name your state, this can be whatever you like, however, giving it a name that describes what the state is can be helpful, such as “24x24 3v3”. This name represents that the state has a map of size 24x24 and each team has three units.
  + Once you have chosen the location to save in, and have given the state a name, click the “Save” button at the bottom of the dialog box, as shown below:



* You have now saved your state!

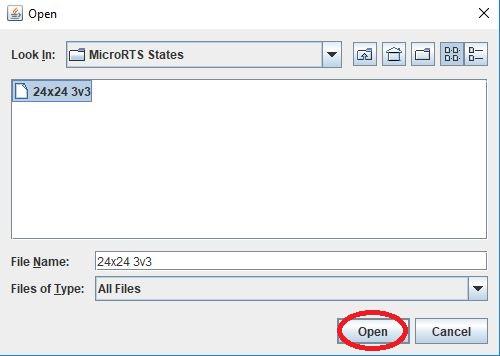
## Tutorial 4: Loading States

Now that we have created and saved a state, lets learn how to load our saved state. Loading states can be useful as it will allow you to modify the state without having to make a whole new one.

To load the state you have created, follow these steps:

* Step 1:
  + Open the “MicroRTS\_States\_Traces” JAR file by double clicking it if you haven’t already.
* Step 2:
  + Next to the “Save” button we used in the previous tutorial, locate the “Load” button, as shown below:

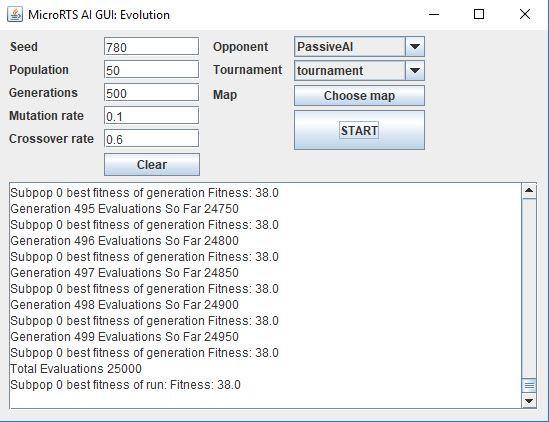


* + Click “Load”
* Step 3:
  + Upon clicking “Load”, you will be presented with a dialog box asking you which file you would like to load.
  + Locate the file you saved in the previous tutorial (for me this is located in the “MicroRTS States” folder I created on my Desktop).
  + Once you have located the file, select it and then click the “Open” button at the bottom of the dialog box, as shown below:
  + You have now loaded your previously saved state!

## Tutorial 5: Evolution

Now that we have learned how to create, save, and load states, we can begin evolving using the state we have previously created.

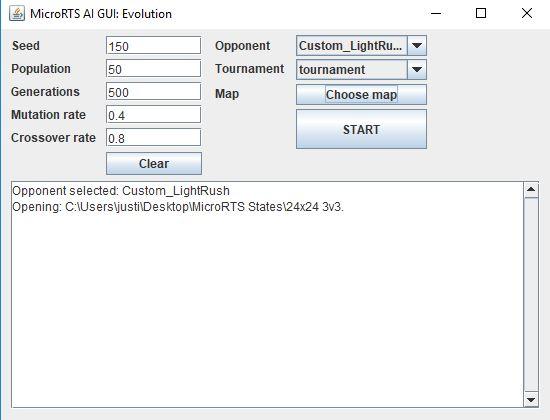
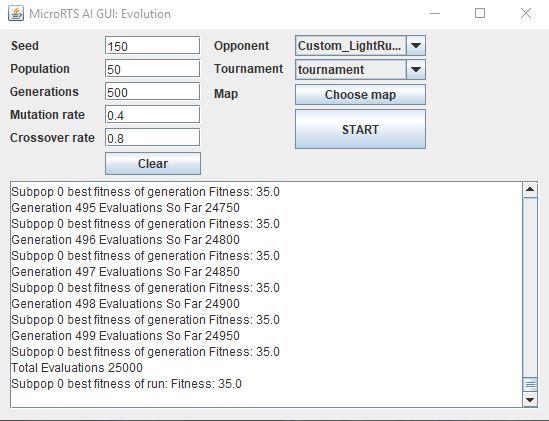
To run an evolution, follow these steps:

* Step 1:
  + Open the “BT\_App” JAR file by double clicking it
  + Upon opening the file, you will be presented with a window in which you can enter and select specific parameters for the evolution, however, for the purpose of this tutorial, we will be using the default parameters. Default parameters are used if no parameters are specified, these default parameters are:
    - Seed: 780
    - Population: 50
    - Generations: 500
    - Mutation Rate: 0.1
    - Crossover Rate: 0.6
    - Opponent: PassiveAI
    - Tournament: tournament
  + What these parameters mean will be explained in the next tutorial.
* Step 2:
  + Click on the “Choose map” button and just as we have done in the previous tutorial, locate your saved state, select it and click “Open”.
    - This specifies the state to use for the evolution
* Step 3:
  + Click on the “START” button
  + Upon clicking “START”, you should notice text being printed to the large textbox at the bottom of the window.
  + This text is showing us information regarding our evolution, where:
    - “Subpop 0” is our population
    - “Fitness #” is our best fitness score of the generation
    - “Generation #” is the current generation the evolution has reached
    - “Evaluations So Far” is a count of how many evaluations have taken place
  + Upon completion of the evolution you will see:
    - Subpop 0 best fitness of run: Fitness: #
    - This is telling us the fitness of the best individual tested during the evolution
  + This is what your window should look like at the end of the evolution if you followed the tutorials exactly:
  + Don’t worry if you see something different, you may have just used a different map!
  + You have now successfully run an evolution!

## Tutorial 6: Advanced Evolution

Now that we have run an evolution with the default parameters, let’s try specifying our own parameters.

To run an evolution with specific parameters, follow these steps:

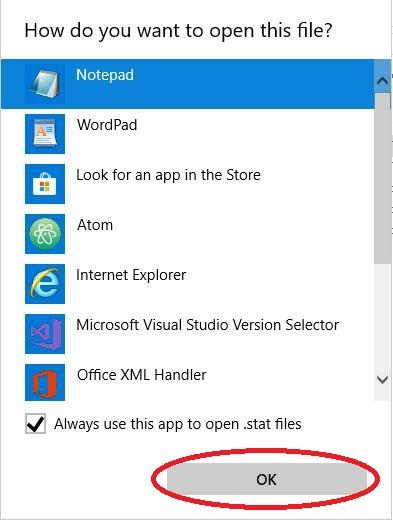
* Step 1:
  + Open the “BT\_App” JAR file by double clicking it
  + Upon opening the file, you will be presented with a window in which you can enter and select specific parameters for the evolution
* Step 2:
  + We will now enter the parameters to use for our evolution.
  + In the textbox labeled “Seed”, enter 150
    - Our randomly generated Behaviour Trees will be based on this seed
    - Some seeds may be better than others at a given problem
  + In the textbox labeled “Population”, enter 50
    - This is the size of our initial random population of Behaviour Trees
  + In the textbox labeled “Generations”, enter 500
    - This is how many generations we would like to reach with our evolution
  + In the textbox labeled “Mutation Rate”, enter 0.4
    - This is the probability that a mutation will occur between generations
    - 0.4 means a 40% chance
  + In the textbox labeled “Crossover Rate”, enter 0.8
    - This is the probability that a crossover will occur between generations
    - 0.8 means an 80% chance
  + For the dropdown box labeled “Opponent”, click the arrow and select “Custom\_LightRush”
    - This is the opponent our AI will be versing in the game
    - Custom\_LightRush is a modified WorkerRush AI that cannot produce units
  + For the dropdown box labeled “Tournament”, click the arrow and select “tournament”
    - This is the kind of tournament (in terms of settings) to be used
  + For the “Map” selection, click on the “Choose map” button and just as we have done in the previous tutorial, locate your saved state, select it and click “Open”
    - This is the state to be used for the evolution
  + Your window should now look something like this:
* Step 3:
  + Click “START”
  + Upon clicking “START”, you should notice text being printed to the large textbox at the bottom of the window.
  + This text is showing us information regarding our evolution, where:
    - “Subpop 0” is our population
    - “Fitness #” is our best fitness score of the generation
    - “Generation #” is the current generation the evolution has reached
    - “Evaluations So Far” is a count of how many evaluations have taken place
  + Upon completion of the evolution you will see:
    - Subpop 0 best fitness of run: Fitness: #
    - This is telling us the fitness of the best individual tested during the evolution
  + This is what your window should look like at the end of the evolution if you followed the tutorials exactly:
  + Don’t worry if you see something different, you may have just used different parameters or a different map!
  + You have now successfully run an evolution (again)!

## Tutorial 7: Retrieving the Output

Now that we have run an evolution, there are a couple of files that you may have noticed have been created. These files can be found within the same folder as the “BT\_App” JAR file.

The files created are named in the following way:

* file number + “-Seed\_” + seed + “\_Crossover\_” + crossover rate + “\_Mutation\_” + mutation rate + “\_out”.stat
* file number + “-Seed\_” + seed + “\_Crossover\_” + crossover rate + “\_Mutation\_” + mutation rate + “\_out\_2”.stat
* in our case (if you followed the tutorial exactly) the files will be named:
  + 0- Seed\_150\_Crossover\_0.8\_Mutation\_0.4\_out.stat
  + 0- Seed\_150\_Crossover\_0.8\_Mutation\_0.4\_out\_2.stat
* The out.stat file contains information regarding:
  + The best individual of each generation
    - The Fitness of the individual
    - The Behaviour Tree of the individual
  + This file also contains (at the bottom of the file) the best individual of the run with the same information about that individual as stated above about the best individual of each generation
* The out\_2.stat file contains four columns of numbers, these columns in order from left to right are:
  + Generation number
  + Mean fitness score of the generation
  + Best fitness score of the generation
  + Best fitness score of the run
* These files can be opened in any text editor. For the purpose of this tutorial, we will be using notepad. Follow the steps below to open the files:
  + Locate the “0- Seed\_150\_Crossover\_0.8\_Mutation\_0.4\_out.stat” file
  + Double click the file, a dialog box will open asking ‘how would you like to open this file?’
  + In the list, locate and click on “Notepad” and then click “OK”, as shown below:



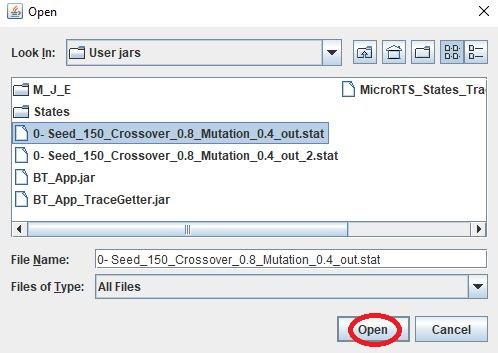
* + The file is now open in Notepad
  + Repeat these steps to open the “0- Seed\_150\_Crossover\_0.8\_Mutation\_0.4\_out\_2.stat” file

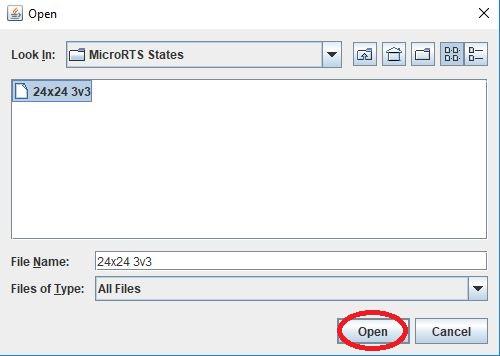
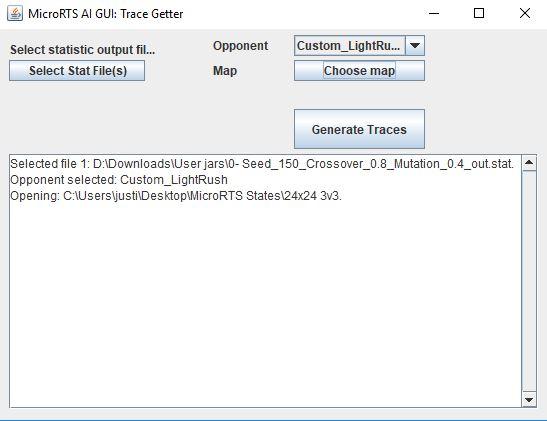
## Tutorial 8: Traces

Now that we’ve run some evolutions and have our data, you may be interested in seeing visually, how the best individual of each run performed. This can be done using traces.

First, we must generate a trace from our out.stat file, to do so, follow these steps:

* Step 1:
  + Open the “BT\_App\_TraceGetter” JAR file by double clicking it
  + Upon opening the JAR file, you will be presented with a window much like the previous one, but with less options.
* Step 2:
  + On the left-hand side of the window, click on the “Select Stat File(s)” button
    - This will open a dialog box for you to locate and open your out.stat file generated during the last tutorial
  + The out.stat file will be located in the same folder as the “BT\_App” JAR file and will be named “0- Seed\_150\_Crossover\_0.8\_Mutation\_0.4\_out.stat”
  + Select the file and click “open”, as shown below:
    - **NOTE: Make sure you select the out.stat file and not the out\_2.stat file.**

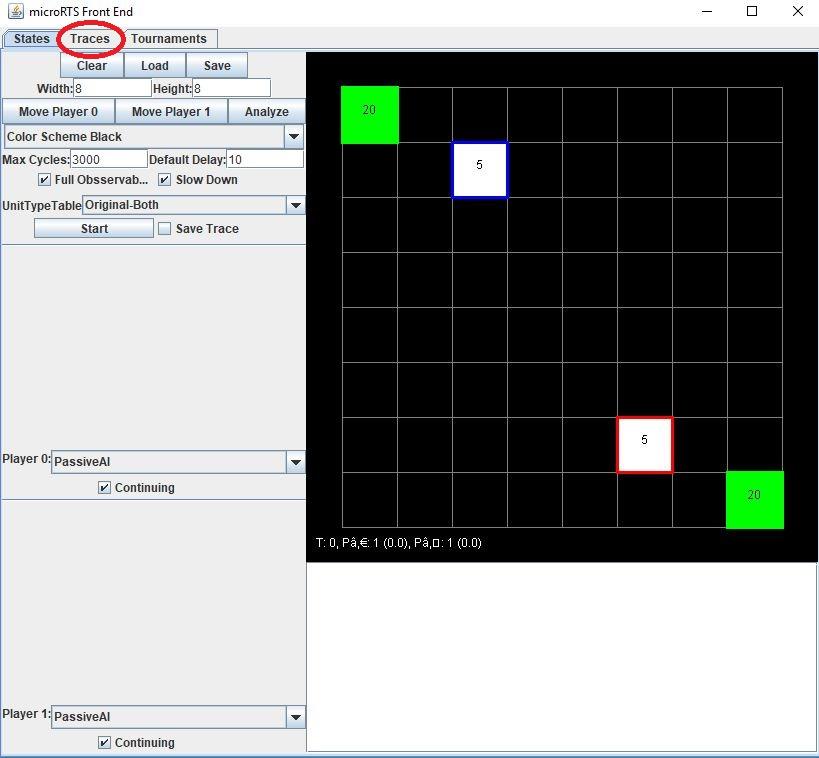
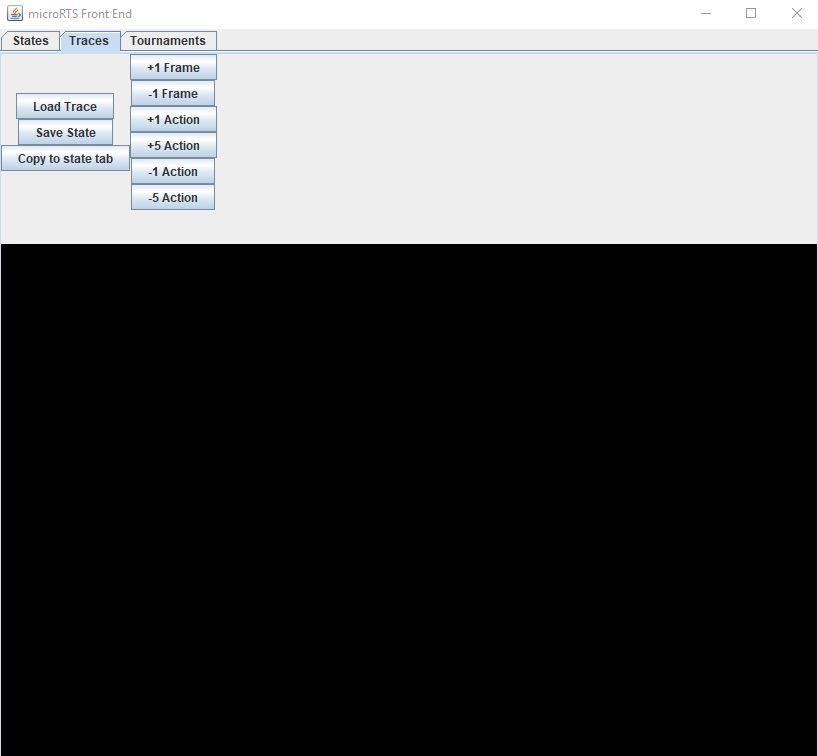
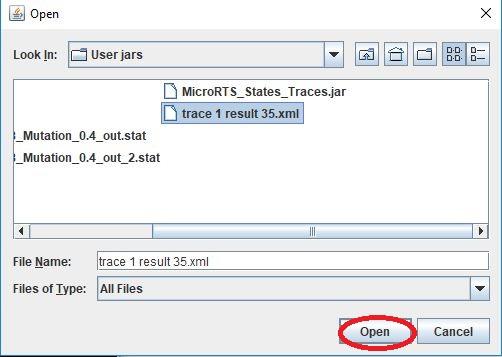
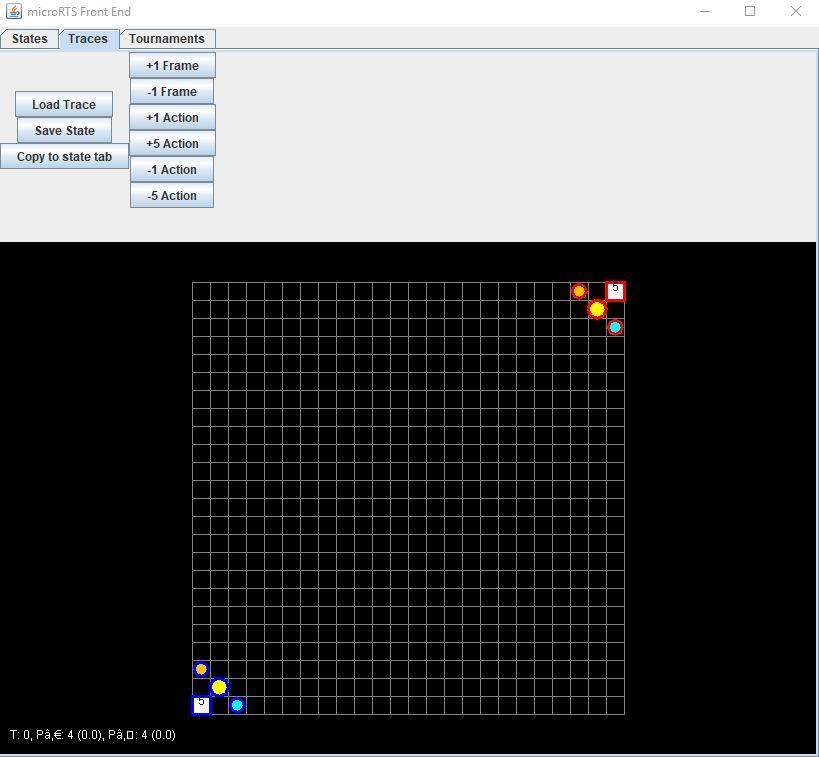


* + Now, we must select the same opponent we selected when running the evolution, in this case it was the “Custom\_LightRush” opponent.
    - Click the arrow on the dropdown box next to the “Opponent” label, and select “Custom\_LightRush”
  + And finally, we must select the same map used in the evolution. This was the state we created earlier:
    - As in previous tutorials, click on the “Choose map” button, navigate to the location you have saved your state in, select it, and click “Open”, as shown below:
    - Your window should now look something like this:
* Step 3:
  + Click on the “Generate Traces” button
    - You will see text printed to the textbox indicating that it is getting traces, followed by text informing you that it has “retrieved trace: 1”
      * You can retrieve multiple traces at a time by selecting more than one out.stat file, however in our case, we only have one out.stat file at the moment
    - You have now created your trace, which can be found in the same folder as the “BT\_App\_TraceGetter” JAR file that we are currently using.
      * In the next tutorial, we will cover running a trace

## Tutorial 9: Running a trace

Now that we have created a trace file, we can watch our evolved AI play the game.

To run a trace, follow these steps:

* Step 1:
  + Open the “MicroRTS\_States\_Traces” JAR file that we used to create our state
* Step 2:
  + At the top of the window, there are three tabs named “States”, “Traces”, and “Tournaments”.
  + Locate the “Traces” tab and click on it, as shown below:
  + You should now see a window which looks like this:
* Step 2:
  + Now that we are on the Traces tab, we can load our trace. This is done as follows:
  + Click on the “Load Trace” button located on the left side of the window
  + As with the previous tutorials, a dialog box will open for you to locate and select your trace
  + Navigate to the folder where the “BT\_App\_TraceGetter” JAR file is located
  + Here you will find the trace created in the previous tutorial
    - If you followed all the tutorials exactly, the trace file will be called “trace 1 result 35”
  + Select the trace file and click “Open”, as shown below:
  + You should now see the trace we created in the bottom area of the window, like so:
* Step 3:
  + Now that we have loaded our trace, we can begin stepping through it to view the AI in action.
  + There are three ways to step forward through a trace, and three ways to step backwards through a trace, these are:
    - Forward:
      * +1 Frame
      * +1 Action
      * +5 Action
    - Backwards:
      * -1 Frame
      * -1 Action
      * -5 Action
    - 1 Frame is equal to one second of game time
    - 1 Action is one action taken by either controller
    - 5 Action is five actions taken by either controller
  + Stepping through a trace is a good way to see the strategy used by our evolved AI
  + You have now successfully run a trace!

# Conclusion

This concludes the tutorials and the user manual. You now know everything you need to effectively use the system and evolve the AI.