

MALSIM: Multi-Agent Learning Simulator and Benchmarking Tool

Drew Wicke

January 14, 2012

The multi-agent learning simulator (MALSIM) will be a simulator and benchmarking tool for multi-agent learning algorithms written in Java. The goal of MALSIM will be to provide multi-agent learning researchers a common framework in which to compare and test various MAL algorithms. MALSIM will solve various problems in the multi-agent learning community. However, in order to create MALSIM technical challenges will be encountered, various technologies used and a user friendly GUI created.

In the multi-agent learning community, when testing a new algorithm empirically the results are usually based on few trials and compared to few opposing algorithms. MALSIM will provide the multi-agent learning community the needed generic testbed for benchmarking and simulating new learning algorithms. MALSIM will provide generic mechanisms to simulate, test and visualize experiments. Currently there are a couple of very good simulators made for multi-agent systems, however they do not provide a benchmarking mechanism [5, 3, 4]. Multiagent Learning Testbed (MALT) provides services similar to that of MALSIM [6]. MALT is extensible and offers a built in mechanism to graph and visualize the results of experiments. However, MALT is coded in MATLAB and is made for two player repeated matrix games. Based on the usefulness of previously created tools the need for a common, generic and extensible testbed for multi-agent learning algorithms is apparent.

In order to provide this tool technical challenges such as a generic implementation and the inherent difficulties of parallel programming will be encountered. The tool must be generic enough to provide the standard features of any simulated game and yet stay functional for all types of multi-agent learning algorithms. Also, while programming the tool for a multithreaded environment challenges such as deadlock, race conditions and communication will be addressed.

MALSIM will be written in Java in order to be platform independent. Technologies such as XStream, a tool that serializes Java objects to and from XML, will be used to provide a mechanism to save and load the tool's model [2]. Also, JFreeChart will be used to provide a way to visualize in a graph based form the results of the experiment [1].

A GUI as shown in figure 1 will be provided. The GUI will allow the user to

set up the experiment by setting things like the game to be played, the agents to be in the tournament, the mechanism to choose what agents play against each other, as well as other properties of the experiment. The GUI will also provide a view of what current games are being played a way for users to pause them. When the experiment is finished the user will be able to view and save the results of the experiment. The GUI will also give the user the option to view and create graphs to compare and contrast the data from the experiment.

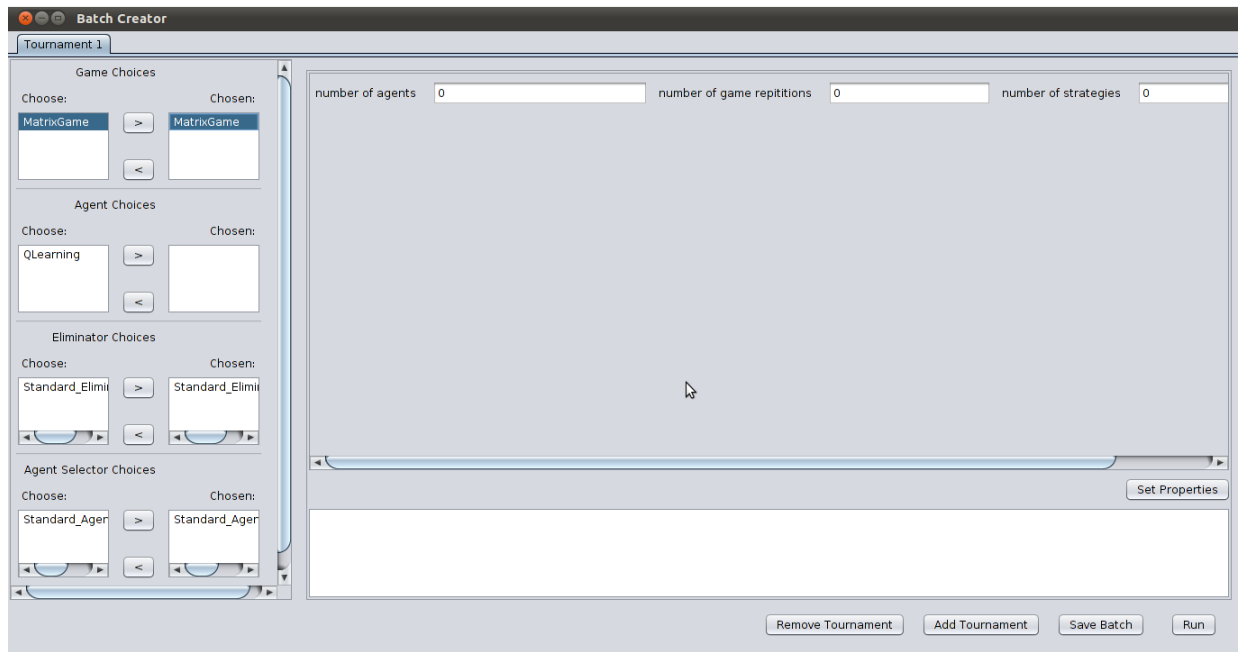


Figure 1: The MALSIM GUI for setting the parameters for the experiment.

References

- [1] Jfreechart. <http://www.jfree.org/jfreechart/>.
- [2] Xstream. <http://xstream.codehaus.org/>.
- [3] T. Balch. Teambots 2.0, 2000.
- [4] C. JIANG, W. HAN, and Y. HU. Repast-a multi-agent simulation platform [j]. *Journal of System Simulation*, 8, 2006.
- [5] S. Luke, C. Cioffi-Revilla, L. Panait, and K. Sullivan. Mason: A new multi-agent simulation toolkit. In *Proceedings of the 2004 SwarmFest Workshop*, volume 8. Citeseer, 2004.
- [6] E. Zawadzki, A. Lipson, and K. Leyton-Brown. Empirically evaluating multiagent learning algorithms. Technical report, Working Paper, November, 2008.