MALSIM: Multi-Agent Learning Simulator and Benchmarking Tool

Drew Wicke

January 18, 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 help to eliminate empirical testing issues that are prevalent 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 their are a couple of very good simulators made for multi-agent systems, however they do not provide a benchmarking mechanism[7, 4, 6]. Multiagent Learning Testbed (MALT) provides services similar to that of MALSIM [10]. 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. GAMUT a benchmarking and game generating tool for game theoretic algorithms is also similar to MALSIM, since many multi-agent learning algorithms are made to play game theory games [9]. 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. 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]. GAMUT may also be integrated into the platform in order to easily generate games. MPJ or JMPI, both pure Java ports of MPI or mpiJava, a Java MPI wrapper may be used to make MALSIM capable of running on a cluster computer [3, 8, 5].

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.

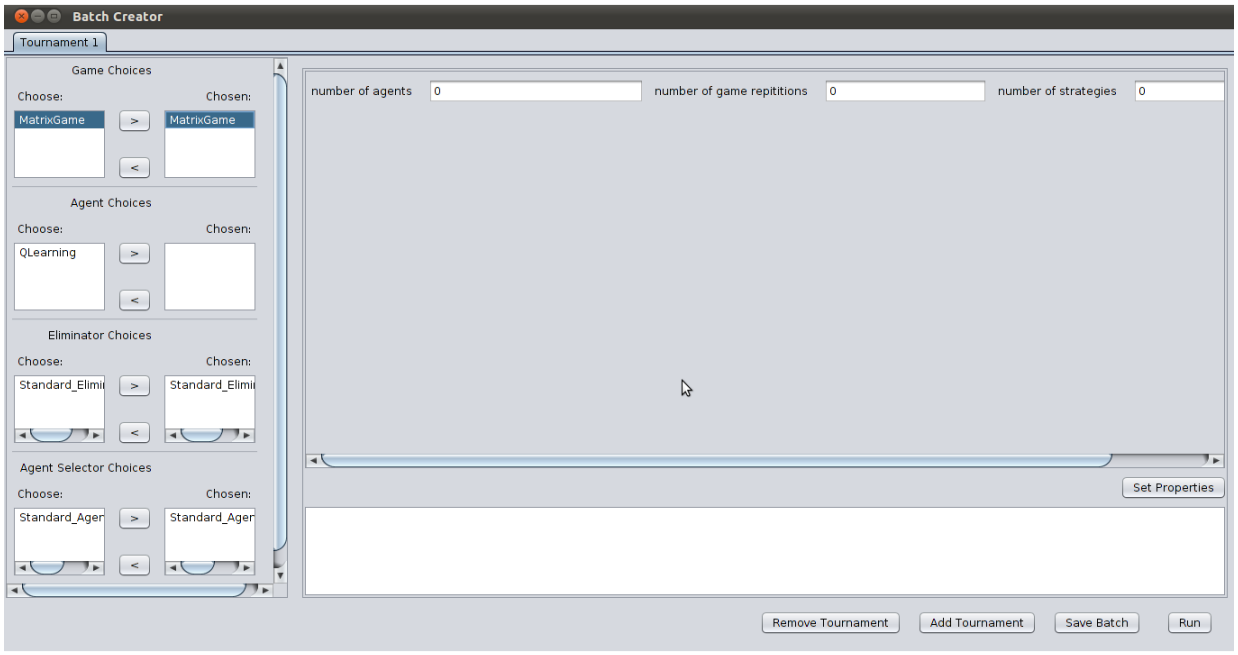

Figure 1: The MALSIM GUI for setting the parameters for the experiment.

References

[1] Jfreechart. *http://www.jfree.org/jfreechart/*.

[2] Xstream. *http://xstream.codehaus.org/*.

[3] M. Baker, B. Carpenter, G. Fox, S.H. Ko, and X. Li. mpijava: a java mpi interface. In *Proceeding of the International Workshop on Java for Parallel and Distributed Computing*, 1998.

[4] T. Balch. Teambots 2.0, 2000.

[5] M. Bornemann, R. van Nieuwpoort, and T. Kielmann. Mpj/ibis: a flexible and efficient message passing platform for java. *Recent Advances in Parallel Virtual Machine and Message Passing Interface*, pages 217–224, 2005.

[6] C. JIANG, W. HAN, and Y. HU. Repast-a multi-agent simulation platform [j]. *Journal of System Simulation*, 8, 2006.

[7] 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.

[8] S. Morin, I. Koren, and C.M. Krishna. Jmpi: Implementing the message passing standard in java. In *Proc. Int. Parallel and Distributed Processing Symposium (IPDPS 2002), Ft. Lauderdale, FL*, 2002.

[9] E. Nudelman, J. Wortman, Y. Shoham, and K. Leyton-Brown. Run the gamut: A comprehensive approach to evaluating game-theoretic algorithms. In *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems-Volume 2*, pages 880–887. IEEE Computer Society, 2004.

[10] E. Zawadzki, A. Lipson, and K. Leyton-Brown. Empirically evaluating multiagent learning algorithms. Technical report, Working Paper, November, 2008.