Open Game Tournaments in STARLITE

(Demonstration)

Jack Hopkins
Dept. of Computer Science
Royal Holloway
University of London, UK
jack.hopkins@me.com

Özgür Kafalı
Dept. of Computer Science
Royal Holloway
University of London, UK
ozgur.kafali@rhul.ac.uk

Kostas Stathis
Dept. of Computer Science
Royal Holloway
University of London, UK
kostas.stathis@rhul.ac.uk

ABSTRACT

STARLITE is a novel web-based agent platform that allows open, dynamic execution of agent simulations. Using STARLITE researchers can start simulations as web applications whose URI can then be used by other researchers to inspect progress with these simulations and deploy their agents on them dynamically to test their behaviour and performance. We describe the overall architecture of STARLITE and exemplify its main features through a tournament simulation for the Iterated Prisoner's Dilemma (IPD).

Categories and Subject Descriptors

I.2.11 [**Distributed Artificial Intelligence**]: Multiagent systems

General Terms

Experimentation

Keywords

Agent platforms, Game tournaments, Web agents

1. INTRODUCTION

Models and representation languages that enable agents to make decisions in social contexts are becoming increasingly important because they help us gain insights into how complex social interactions might evolve in the real world. However, how to make these models computationally tractable and use them to test hypotheses for practical applications is often a challenge. One common approach in this context is to experimentally validate theoretical models with simulation, as it is often the case with mechanisms based on game theory. In such applications accuracy can be traded for complexity, allowing for far more complex models that explain a wide range of phenomena, from altruism to eusociality. Insights into such systems can be made by analysing individual agent strategies and how they compare when competing with each other. Despite extensive research into such strategies, very few general simulation environments have been proposed for game simulations.

STARLITE (Simulation Testbed for Agent Research) [1] is a general and adaptive web-based framework that allows continuous testing of agent strategies contributed by researchers from all over the world, via custom simulations (e.g. tournaments). It is built upon GOLEMLITE [1], a framework that allows both agents and environments in which they are situated to be specified imperatively (in Java) or declaratively (in Prolog). The work introduces STARLITE and outlines how it can be used to run an IPD tournament.

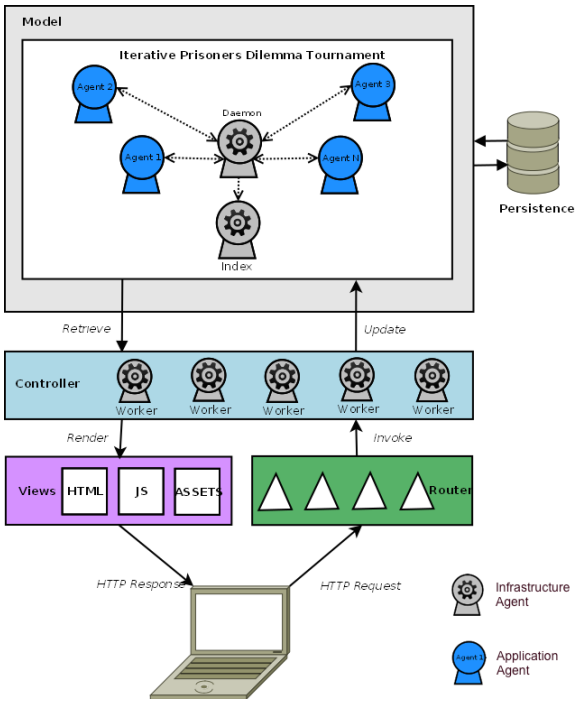

Figure 1: The MVC-based STARLITE architecture.

2. RELATED WORK

Popular agent platforms (e.g. JADE) do not allow developers to add agents dynamically from any web-location to an already running application. Modelling4All[2] allows non-programmers to build agents on the web. However, developers can only utilise high level functions such as *do-at-setup* and *do-every [interval]*, which can be restrictive in some applications. Because Modelling4All programs execute in the browser within a 'NetLogo' Virtual Machine [4], their agents cannot communicate over HTTP and connect to external APIs/web-services in the way that STARLITE agents can. Reliance on NetLogo limits incorporating work from the wider agent developers' community into applications. Axelrod's simulation ex-

[1]http://dice.cs.rhul.ac.uk/software_project/starlite/

[2]http://m.modelling4all.org/

Appears in: *Proceedings of the 14th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2015), Bordini, Elkind, Weiss, Yolum (eds.), May 4–8, 2015, Istanbul, Turkey.*

periment for IPD tournaments incorporated agents submitted by leading game theorists. Since this experiment several frameworks have been developed for research into game theory, with Halpern *et al.* [2] being a notable example. Despite recent implementations of IPD tournaments [3], there have been relatively few attempts to create generalised testbeds for game-theoretic simulations.

3. STARLITE OVERVIEW

STARLITE generalises and distributes a meta-platform to act as a repository for multi-agent experiments. To support scalability and fault recovery, STARLITE uses several state-of-the-art Java and Scala technologies allowing for high performance, easy extensibility as well as supporting a distributed web platform, to mitigate the single point of failure concern which plagues most centralised systems. STARLITE utilises the actor-based environment *Akka* for load balancing and task execution, and the Play framework[3] for Scala templating and HTTP request handling. Figure 1 summarises this architecture, based on the Model-View-Controller (MVC) pattern. STARLITE supports the following types of agents:

- Platform Infrastructure Agents: These agents exists outside the tournament domain model, and handle incoming connections over HTTP. They are stateless agents and therefore suitable for recycling and easy substitution / recovery (as the agent state need not be transferred when the agent is replaced). The number of these *worker* agents increase proportionally with the server load, to allow the platform to scale well with the number of users and running simulations.

- Application Infrastructure Agents: These are application specific agents, which in the case of game tournaments are responsible for maintaining tournament state and serving custom application views over HTTP. Moreover, they can provide tournament data asynchronously for other agent views, meaning that we can display agent utility information in real-time without having to reload view pages.

- Application Agents: These are the application agents, which (in the case of IPD) participate in tournaments and implement different game theoretic strategies.

All agent classes can be added dynamically into running applications through the posting of source code RESTfully to a STARLITE server. Compilation takes place via *hot patching*, and the resulting agent can be instantiated at will and added to any tournament that the user has permission to access. Tournament scenarios are considered first class entities alongside agents, and can similarly be uploaded, instantiated and configured in real-time.

4. APPLICATION: IPD TOURNAMENT

We have reimplemented Axelrod's IPD tournament in STARLITE, with the use of two infrastructure agents and four initial application agents, each of which employs one of the following distinct strategies: *defect*, *cooperate*, *random* and *tit for tat*. We consider the former three to be control strategies, allowing for other strategies to be better examined through competition. These application agents serve dynamic web content representing the historical success of the agent and its utility over time against all competitors. In addition to the application agents, a *utility inspector* agent keeps an up to date reference of all other entities in the tournaments, and is responsible for directing user-agent interaction via an HTML view

[3]https://www.playframework.com

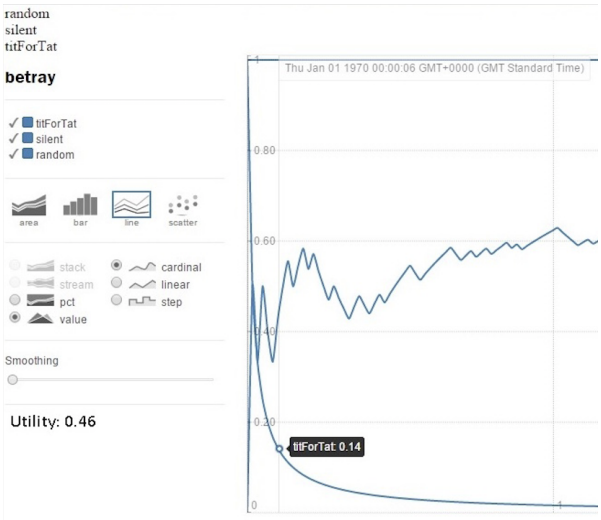

Figure 2: Web view from a running Defect agent.

served over the web. The other infrastructure agent is the *tournament manager*, which is responsible for brokering agent-to-agent interaction within the tournament. This agent maintains the tournament state, and is informed by the utility inspector agent whenever an agent enters or leaves a running tournament. For each simulation iteration, the *tournament manager* broadcasts messages to all application agents, each containing the ID of another agent and the decision history of the pair. The recipient can then reason on this information and reply with a decision to the manager.

Figure 2 demonstrates a view from the Web interface. All application agents display their global utility and also display a chart showing the running mean utility of the subject agent against all other competitor agents. In this application, by default each agent will play every other 500 times, at which point it will go dormant to save server resources. The addition of any new agents to the application will wake up all application agents to resume competition.

5. CONCLUSIONS & FUTURE WORK

We have described the overall architecture of STARLITE and exemplified its main features through a tournament simulation for the Iterated Prisoner's Dilemma (IPD). We have also used the platform to build agent applications for negotiation, e-health and systems biology. Because of its web-based interface, STARLITE is suitable for social experiments, where human users can compete/cooperate with software agents playing different games. This is an exciting direction that we would like to pursue in the future, as we have all the necessary components to support such experimentation.

REFERENCES

[1] S. Bromuri and K. Stathis. Situating Cognitive Agents in GOLEM. In *EEMAS'07*, Germany, Oct. 2007. Springer.

[2] J. Y. Halpern and R. Pass. Algorithmic rationality: Adding cost of computation to game theory. *SIGecom Exch.*, 10(2):9–15, 2011.

[3] C. H. Pence and L. Buchak. Oyun: A new, free program for iterated prisoner's dilemma tournaments in the classroom, 2012.

[4] E. Sklar. NetLogo: a multi-agent simulation environment. *Artificial Life*, 13(3):303–311, 2007.