Ipseity: An Open-Source Platform for Synthesizing and Validating Artificial Cognitive Systems in MAS

(Demonstration)

Fabrice Lauri
IRTES-SeT
Rue Thiery-Mieg, Belfort, FRANCE
fabrice.lauri@utbm.fr

Abderrafiaa Koukam
IRTES-SeT
Rue Thiery-Mieg, Belfort, FRANCE
abder.koukam@utbm.fr

ABSTRACT

This article presents an overview of IPSEITY, an open-source platform developed in C++ with the *Qt* framework. The current version of the platform includes a set of plugins implementing single-agent and multi-agent environments, hard-coded controllers based on Artificial Intelligence (AI) techniques, classical Reinforcement Learning (RL) techniques like Q-Learning, Sarsa, Epsilon-Greedy combined with some linear function approximators, as well as a Machine Learning (ML) technique for Apprenticeship Learning (AL). Its architecture allows users to execute standard AI approaches as well as model-based, model-free, offline, online, standard and approximated RL algorithms. IPSEITY is targeted at a broad range of users interested in AI in general, including industrial practitioners, as well as ML researchers, students and teachers. It is regularly used as a course support in Artificial Intelligence and it has been used successfully to manage power flows in simulated microgrids using multi-agent reinforcement learning.

Categories and Subject Descriptors

D.2.13 [**Software Engineering**]: Reusable Software

General Terms

Algorithms, Experimentation

Keywords

Multi-agent system, software platform, open-source, cognitive systems, artificial intelligence, reinforcement learning.

1. INTRODUCTION

Solving sequential decision problems involves writing codes, conducting experiments, and comparing results obtained from alternative approaches. Whereas solving Supervised Learning problems can be facilitated by using software tools like WEKA, there are currently no fully-featured experimental platforms dedicated to the study of AI algorithms for solving sequential decision problems. Designing and developping such a platform implies taking into account a wide range of environments, agent architectures, decision-taking algorithms and performance measures. For example, defining a multi-agent environment requires answering many questions about the structure of the state, action and time spaces, about the deterministism or stochasticity of the state transition function, whether decision-taking processes are static or dynamic, whether there will be independent learner agents, cooperative agents or competitive agents, etc. Likewise, defining the architecture of agents inhabiting a MAS, elaborating decision-taking algorithms and devising performance measures for evaluating the quality of the agents' decisions bring many software design issues.

To our knowledge, IPSEITY is currently the only multi-agent platform that allow users interested in solving sequential decision problems to easily study the influence of some design parameters on the performance obtained by dedicated AI algorithms using accepted benchmarks. Indeed, RL-Glue, CLSquare, PIQLE, RL Toolbox, JRLF, LibPGRL only support single-agent RL techniques. The *MATLAB MDP toolbox* by Chadès and the *MARL Toolbox* by Busoniu supports multi-agent RL under Matlab, but they are not as easily extensible as IPSEITY, that enjoys the concepts of encapsulation, inheritance and polymorphism of object-oriented programming. An overview about the key concepts, the main functionalities and the properties of this platform is presented thereafter.

Appears in: *Alessio Lomuscio, Paul Scerri, Ana Bazzan, and Michael Huhns (eds.), Proceedings of the 13th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2014), May 5-9, 2014, Paris, France.*

2. OVERVIEW

IPSEITY is an open-source platform especially dedicated to facilitating the implementation and the experimental validation of different kinds of behaviors for cooperative or competitive agents. These agents can evolve within a dynamic or static Multi-Agent System. The state space, the action space and the time space of the environment may be either discrete or continuous.

2.1 Kernel Concepts

In IPSEITY, a set of *agents* interact within a given *environment*. A set of agent groups, called *taxons* in IPSEITY, can be defined. Agents grouped together into the same taxon may behave similarly, because they share the same class of decision making mechanism. The actions of an agent are performed after deliberation of its *cognitive system*. A cognitive system implements the decision process that allows an agent to maps actions from perceptions and it must be plugged to a taxon. This means that all the agents associated to the same taxon use the same decision process. A

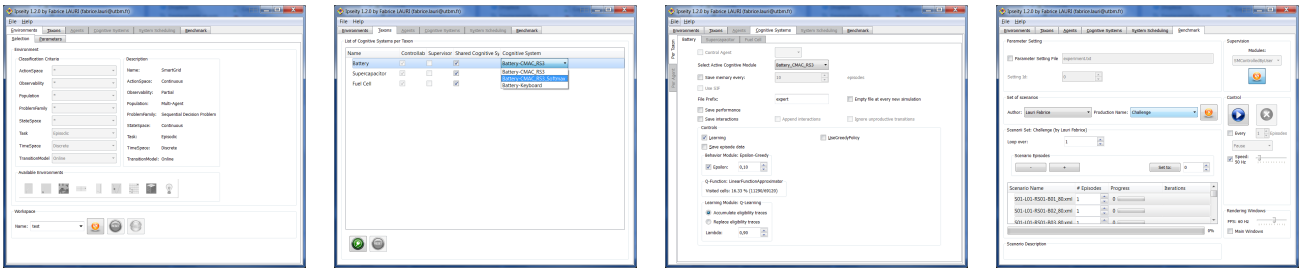

Figure 1: Some screenshots of Ipseity

cognitive system embeds a set of AI algorithms that might be executed alternatively, concurrently or sequentially at will. The current version of Ipseity provides only the alternative execution of algorithms, but the action returned by the executed algorithm may be used to revise the future decisions of the idle algorithms. For example, the user can interact with the system by using its keyboard to indicate the good actions the agent has to perform in some situations. These expert actions may be used by a ML algorithm during the simulation involving the user or afterward. This functionality is particularly useful in Apprenticeship Learning. The instants at which the environment updates and at which the agents perceive and act can be defined within a *system scheduling*. For studying the quality of the decisions taken individually or collectively by the agents under some initial conditions, the user can define a *benchmark*. A benchmark consists of a *parameter setting* and a set of *scenarios*. The parameters of the environment or those of the tested decision-taking algorithms can be automatically set from predefined values specified in the parameter setting. A scenario specifies the initial environmental condition of the MAS. The order and the numbers of episodes of the predefined scenarios may be changed by the user himself or an agent. Statistics about the performance of the executed algorithms and about the agents' interactions are generated within a *workspace* repository during the simulation of a benchmark. All these concepts have been defined within classes in Ipseity.

2.2 Platform Functionalities

The current version of Ipseity allows the user to load a predefined environment, to create and destroy workspaces, to load predefined scenarios, to plug and unplug cognitive systems to taxons, to set the parameters of a predefined system scheduling, of the loaded environment and of the loaded cognitive systems, to load a predefined scenario supervisor, to select the scenarios involved in the simulation, to start/pause/stop the simulation, to modify the simulation speed and to select the statistics that are generated during the simulation. The predefined single-agent environments are Acrobot, Cartpole, DoubleIntegrator, InvertedPendulum, MountainCar, RasendeRoboter, Rubik's Cube. The multi-agent environments are SmartGrid [2] and a grid-world video-game named Delirium2.

2.3 Properties

Ipseity satisfies many of the requirements proposed by Kovacs *et al.* [1] about the design of software for RL. In particular, Ipseity uses kernel concepts and components (i.e. data representations and algorithms) that are as flexible as possible. Indeed agent perceptions and actions are represented by 64-bit float vectors, allowing agents to be immerged in discrete or continuous environments. Ipseity is broadly extensible as it is based on object-oriented programming and it uses DLL as plugins to implement environments, cognitive systems, system schedulings and scenario supervisors. These plugins can be easily integrated in other systems and applications. For example, Ipseity can be used to learn the behaviors of some agents. Once the learning phase is finished, agents can perceive information from a remote environment and act according to the learnt behavior. Such integration has been realized between Ipseity and a Java-based microgrid simulator [2]. Ipseity can easily be extended by specialized plugins for the target application area. Customized extensions include new environments, algorithms that take part in the decision processes of some cognitive systems or rendering modules for some predefined environments. Ipseity supports the user in keeping track of all the data generated during simulations, such as the performed actions of agents and statistics about the performance results of the AI algorithms that are empirically studied. Ipseity provides a user-friendly interface (see Fig.1) with informative icons and widgets for setting up all the parameters involved in the simulation of a MAS, including those of the environments, of the system scheduling and of the cognitive systems.

3. CONCLUSION

An overview of Ipseity has been presented in this article, with a focus on the Reinforcement Learning based functionalities. Ipseity is highly modular and broadly extensible. It can be freely downloaded from `http://www.ipseity-project.com` under a GNU *GPLv3* open-source licence. A demonstration of the platform is available on `http://www.ipseity-project.com/demo/ipseity.avi`. Latest developments in the platform cover an approach (Reward-regularized Classification for AL) dedicated to Apprenticeship Learning. RCAL will be presented at AAMAS 2014. Forthcoming functionalities include new multi-agent environments, new state-of-the-art RL algorithms and a Web front-end for facilitating the collaboration among the community.

4. REFERENCES

[1] T. Kovacs and R. Egginton. On the analysis and design of software for reinforcement learning, with a survey of existing systems. *Machine Learning*, 84:7–49, 2011.

[2] F. Lauri, G. Basso, J. Zhu, R. Roche, V. Hilaire, and A. Koukam. Managing Power Flows in Microgrids using Multi-Agent Reinforcement Learning. In *Agent Technologies in Energy Systems (ATES)*, 2013.