FMAP: A Platform for the Development of Distributed Multi-Agent Planning Systems

Alejandro Torreño, Óscar Sapena, Eva Onaindia

Universitat Politècnica de València
Camino de Vera, s/n, 46010, Valencia

Abstract

The development of cooperative Multi-Agent Planning (MAP) solvers in a distributed context encompasses the design and implementation of decentralized algorithms that make use of multi-agent communication protocols. In this paper, we present FMAP, a platform aimed at developing distributed MAP solvers such as MAP-POP, FMAP and MH-FMAP, among others.

Keywords: multi-agent planning, distributed algorithms
2010 MSC: 68-20, 68-42

1. Introduction

Cooperative Multi-Agent Planning (MAP) generalizes automated planning to a context in which several autonomous entities, or agents, plan and act concurrently in a common environment towards a common goal. The recent 2015 Competition of Distributed and Multiagent Planners (CoDMAP) [1] was the first attempt to showcase the current MAP technology. The competition was arranged in two tracks, a centralized and a distributed track, in order to classify and compare the participating MAP solvers. Most of the planners are based on a single-host running process and took part in the centralized track only. Centralized MAP rules out the need of a communication infrastructure and distributed computation, and enables to reuse widely developed single-agent planning technology. In contrast, the development of fully-distributed MAP platforms is far less exploited as evidenced by the few solvers that participated in the CoDMAP distributed track.

Distributed computation poses additional requirements and challenges such as an efficient agent communication and the design of algorithms suited

to a distributed control. In cooperative MAP, where all agents are aimed at solving a common goal, the distribution of the information and distributed problem-solving compel agents to constantly exchange knowledge and partial solutions. Hence, implementing distributed MAP algorithms is a challenging task that involves a set of independent *software agents*.

This paper introduces FMAP, a software platform for the development of MAP solvers based on multi-agent heuristic search [2]. FMAP supports language, runtime and associated components to develop distributed solutions to a cooperative MAP task. We provide a flexible platform that allows for a fast development of distributed search algorithms and heuristics and includes a thoroughly-tested communication infrastructure. In order to minimize communication overhead, FMAP uses a democratic leadership scheme by which a coordinator role is scheduled among the agents, so that one of the agents centralizes messaging and leads the procedure at each iteration [2].

The remainder of this article is as follows: section 2 analyzes the architecture and the main features of our software platform; section 3 compares FMAP against other existing MAP tools; and section 4 concludes.

2. System Architecture and Functionalities

FMAP attains cooperative MAP tasks in which a set of independent planning agents jointly develop a course of action or plan to reach a common goal from a given initial situation or *state*. A state of the world is defined as a finite set of state variables, each associated to a finite set of mutually exclusive values. Agents have a *local view* of the world defined by their state variables and values. The variables/values of an agent which are not shared with the others are *private* to the agent.

The architecture of FMAP is depicted in Figure 1. The platform works with a factored planning representation such that the user introduces, for each planning agent ag_i, a domain file that describes its domain knowledge, and a problem file that contains ag_i's local view of the initial state and the goal, as well as the information of ag_i that is shareable with the others.

FMAP can be run through either a *command-line mode* or an *interactive graphical interface* that serves as a powerful execution monitoring and debugging tool. Since agents communicate via TCP-IP sockets, the command-line mode allows tasks to be executed in multiple hosts, running one or various agents in each host. For this purpose, the user must provide an *agent list* file (see Figure 1) that specifies the names and IP addresses of the agents.

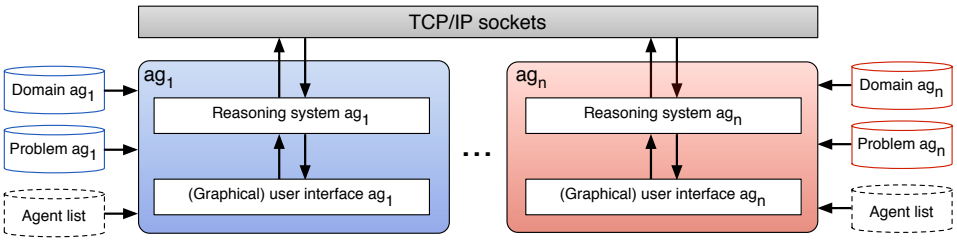

Figure 1: Architecture of the FMAP platform

A planning agent integrates a (graphical or textual) user interface and a reasoning system. The interactions among the components of the agent's reasoning module are described in Figure 2. Via the *pre-processor*, the agents lexically and syntactically analyze their domain and problem files (*parsing*), and instantiate the arguments of the actions and state variables of their tasks (*grounding*). In grounding time, agents jointly analyze the reachability of the task actions, excluding those that will not be reachable when solving the planning task. After pre-processing the MAP task, agents generate a set of *auxiliary structures* to facilitate the heuristic evaluation of plans; namely, a *landmark graph* and a set of domain transition graphs (*DTGs*) [3].

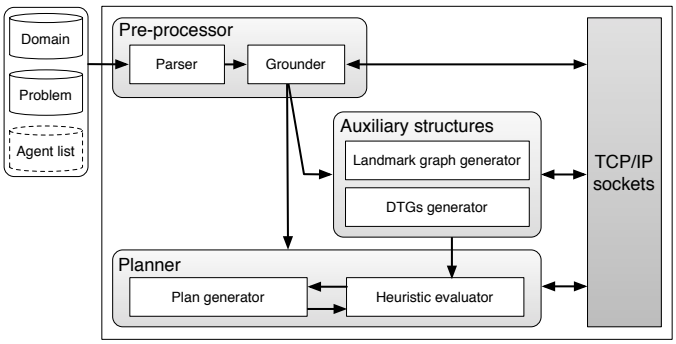

Figure 2: Structure of an agent's reasoning system

The *planner* component of an agent includes a *plan generator* and a *heuristic evaluator*. Currently, the platform applies MH-FMAP, a multi-agent A* search scheme governed by two different heuristic functions that are applied alternatively to select plans [3]. Nonetheless, the code of FMAP is easily extendible and the programmer can modify and add new search schemes and heuristic functions.

As depicted in Figure 2, agents communicate via TCP/IP sockets. The communication infrastructure of FMAP includes a collection of robust methods to synchronously and asynchronously exchange messages. A FMAP agent incorporates a queue to store and manage the received messages. The socket-based infrastructure of the platform allows agents to be executed in different machines and automatically coordinate through the network.

From a development perspective, FMAP is a software platform implemented in Java that is executable under multiple operating systems. In order to maximize modularity, FMAP heavily draws upon Java *interfaces*, which encapsulate the components of the code. This facilitates ease of replacement of the existing algorithms by new implementations, as long as they adhere to the corresponding interfaces.

3. Comparison with other MAP Tools

The state of the art in MAP includes several tools that share some of the design principles of FMAP. LAPKT[1] is a command-line modular platform for single-agent planning that allows users to assemble and combine different search strategies and heuristics. The centralized MAP solver MAP-LAPKT compiles the MAP task into a single-agent task and uses LAPKT to solve it.

Regarding distributed MAP tools, we can cite the two solvers that participated in the distributed track of the 2015 CoDMAP along with MH-FMAP. MAPlan [4] applies a distributed heuristic search scheme and combines several local and global heuristic functions. Similarly to our tool, MAPlan communicates agents through network message passing. PSM [5] solves a MAP task by merging the various finite automata that represent the agent plans. PSM was the top solver in the distributed CoDMAP thanks to its efficient handling of communication among agents, based on the use of a specialized broker agent.

Unlike the aforementioned solvers, FMAP is designed as an extendible tool whose modular design enables an easy replacement of the code. Moreover, FMAP is also the first MAP tool to feature a powerful graphical interface for execution monitoring and debugging purposes. Finally, the Java-based implementation turns FMAP into a portable tool and the message infrastructure allows agents of a MAP task to be effortlessly run in multiple hosts under different operating systems.

[1] http://lapkt.org

Regarding performance, MH-FMAP, which is the most advanced MAP solver developed with the FMAP platform [3], has been extensively tested and compared with other approaches in the literature. In [3], MH-FMAP was tested via a comprehensive benchmark that includes 10 domains from the IPC (International Planning Competition) adapted to a MAP context. Moreover, in the 2015 CoDMAP, MH-FMAP exhibited a consistent performance and it ranked third in the distributed track[2].

4. Conclusions

FMAP is a software platform that represents one step ahead in the simplification of the design, development and debugging of a distributed MAP solver. The extendible components of the platform such as the communication infrastructure, the GUI and the search engine alleviate debugging tasks and allow users to focus on the design and development of distributed algorithms, thus saving time and effort.

References

[1] A. Komenda, M. Stolba, D. L. Kovacs, The international competition of distributed and multiagent planners (CoDMAP), AI Magazine 37 (3) (2016) 109–115.

[2] A. Torreño, E. Onaindia, O. Sapena, FMAP: Distributed cooperative multi-agent planning, Applied Intelligence 41 (2) (2014) 606–626.

[3] A. Torreño, E. Onaindia, O. Sapena, Global heuristics for distributed cooperative multi-agent planning, in: Proceedings of the 25th International Conference on Automated Planning and Scheduling (ICAPS), 2015, pp. 225–233.

[4] D. Fišer, M. Štolba, A. Komenda, MAPlan, in: Proceedings of the Competition of Distributed and Multi-Agent Planners (CoDMAP-15), 2015, pp. 8–10.

[5] J. Tožička, J. Jakubuv, A. Komenda, M. Pěchouček, Privacy-concerned multiagent planning, Knowledge and Information Systems 48 (3) (2016) 581–618.

[2] http://agents.fel.cvut.cz/codmap/results/

Required Metadata

Current executable software version

S1	Current software version	v1.0
S2	Permanent link to executables of this version	*https* : *//bitbucket.org/altorler/fmap*
S3	Legal Software License	GNU General Public License v3
S4	Computing platform/Operating System	Java-compatible platform
S5	Installation requirements & dependencies	Java 1.7 or greater
S6	If available, link to user manual - if formally published include a reference to the publication in the reference list	*https* : *//bitbucket.org/altorler/fmap /overview*
S7	Support email for questions	altorler@upvnet.upv.es, osapena@dsic.upv.es

Table 1: Software metadata

Current code version

C1	Current code version	v1.0
C2	Permanent link to code/repository used of this code version	*https* : *//bitbucket.org/altorler/fmap*
C3	Legal Code License	GNU General Public License v3
C4	Code versioning system used	Git
C5	Software code languages, tools, and services used	Java 1.7
C6	Compilation requirements, operating environments & dependencies	Java JRE 1.7 or greater
C7	If available Link to developer documentation/manual	*https* : *//bitbucket.org/altorler/fmap /overview*
C8	Support email for questions	altorler@upvnet.upv.es, osapena@dsic.upv.es

Table 2: Code metadata