Moving Robot Accountability to the Cloud: Effects in Knowledge Representation Processes

Miguel Á González-Santamarta[1], Francisco J. Rodríguez-Lera[1], Francisco Martín[2], Camino Fernández-Llamas[1], and Vicente Matellán[1]
[1]Robotics, Group, Universidad de León, León, Spain, {*mgons, fjrodl, vicente.matellan, camino.fernandez*}*@unileon.es*
[2]Intelligent Robotics Lab, Universidad Rey Juan Carlos, Madrid, Spain, *fmrico@urjc.es*

***Abstract*—Future legal requirements will push for creating cognitive architectures and decision-making frameworks capable of offering human-interpretable information about their actions. Fulfilling these requirements will require a high computational cost. Thus, these systems have been moved from local scenarios to cloud environments where it is possible to exploit High Performance Computing features, but uploading these processes to the cloud would also impose unacceptable latencies for some robotics but suitable for auditing.**

This paper presents an empirical evaluation of knowledge manipulation and processing based on PDDL (Planning Domain Description Language) in four scenarios: local native, local container, mixed and cloud. The knowledge about the robot itself and its environment is a cornerstone in any cognitive architecture no matter where is the scenario. The knowledge representation enables in turn reasoning and the implementation of regular cognitive functions such as memory, attention, information processing, and reasoning on cognitive architecture. In this initial iteration, we can see the high-performance impact of partially moving knowledge manipulation to the cloud, particularly in mixed local/cloud scenarios.

I. Introduction

Knowledge representation for planning in robotics is usually made using PDDL [1]. We propose interacting with the PDDL directly from the code, letting knowledge be updated in run-time, instead of using files for representing the domain and the problem. We have developed two different alternatives, an in-memory based storage, and a MongoDB [1] based storage one. This proposal will help the planning system to be run on the cloud.

There have been different proposals for PDDL based planning in cloud scenarios, such as [4]. The pros and cons of using AI in the cloud have been also analyzed in the literature, for instance in [5].

We have tested our approach in different scenarios and different storage alternatives. All these scenarios are based on the three-layers cognitive architecture that we are developing. This cognitive architecture is the new version of our previous architecture [3]. This new version, whose name is MERLIN2, is composed of four layers. Each of these layers can use our PDDL management system. The layers are:

- Mission Layer: this layer generates the PDDL goals.
- Planning Layer: this layer creates the plans.
- Executive Layer: this layer is made up of the actions that the robot can use.
- Reactive Layer: this layer is composed of reactive systems.

[1]https://www.mongodb.com/

A. Paper Objectives

Although legal requirements impose that information has to be stored for a year, a decision-making system has to work with real-time information. For this reason, the implementation of the cognitive architectures must take into account the robot performance, in particular, if they have to face the development of human-interpretable systems that increase the resources required. This paper deals with measuring the collateral effects of using cloud-based architectures for processing, storing, and manipulating information about the robot environment or robot status. In particular, this study evaluates the performance in four different scenarios: native local processes, a locally running container, a mixed approach and a cloud approach.

II. Experimental Setup

A. Hardware Description

The on-board evaluation was carried out in a MSI (Micro-Star International Co., Ltd.,) laptop product name MS-16JF, with 32GB RAM, and using 6 cores of an Intel(R) Core(TM) i7-8750H CPU @ 2.20GHz able to run 2 threads per core. A server with 16GB RAM and running on an Intel(R) Xeon(R) CPU E5-2640 v4 @ 2.40GHz was used for the cloud option. It assembles 8 sockets with 1 core per socket. Besides, each core can run 1 thread.

B. Knowledge Management

This research proposes two types of information manipulation associated with where and how the robot knowledge is being processed. These methods are based on two different ROS2[2] implementations. The first one uses a non-relational approach based on MongoDB. The other one is based on an in-memory approach. The development is set up on software design patterns [2]. This way, the Data Transfer Object (DTO) and Data Access Object (DAO) patterns are used to manage the PDDL associated with architecture knowledge.

C. Software Approach

In order to simplify the experimental setting, the cognitive architecture has been deployed in three Docker [3] containers: one container for MongoDB, another for the knowledge base of the architecture and the third one for an architecture simulation. The knowledge base is a ROS2 node that has three ROS2 services for each PDDL element. The first service is

[2]https://index.ros.org/doc/ros2/Releases/Release-Foxy-Fitzroy/
[3]https://www.docker.com/

used to query the knowledge base, the second service is used to save, edit or delete the knowledge; and the third service is used to delete all the knowledge of a PDDL element.

D. Scenarios

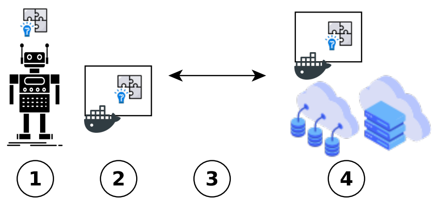

Fig. 1. Proposed framework.

This research proposes four scenarios of knowledge management associated with where information is being analyzed (Figure1). Some scenarios have been build using docker containers. There three containers: the main container that manipulates the knowledge, the MongoDB container and the in-memory storage container. The scenarios are the following:

1) Local Native: no containers are used. The knowledge manipulation is performed in the local computer on-board the robot.
2) Local Container: the three containers are executed in the local computer on-board the robot.
3) Mixed (local+cloud): the main container is executed in the computer and the storage containers are in the cloud.
4) Cloud: the three containers are in the cloud.

III. Results

A. System Impact

In order to evaluate the performance of these four approaches, we propose to run a task iteratively. The task consists of five knowledge manipulation jobs associated with a real task proposed in SciRoc rulebook[4]. These manipulations, which defines an iteration in our experiment, are the following:

1) Resetting the knowledge base.
2) Loading the initial knowledge.
3) Checking the state of the tables of a restaurant.
4) Serving and order to a table.
5) Guiding a person to a table.

The experiment ran 3000 times the task for each scenario, except for the mixed scenario that was run only 60 times. This is due to the great amount of time needed to run the mixed scenario. The command `time` was used to determine the time spent in user and kernel mode for each iteration.

TABLE I
Seconds needed to perform knowledge manipulation.

	Local (Native)		Local Container		Mixed (60)		Cloud	
	M2	DB	M2	DB	M2	DB	M2	DB
Real	10809.344	8683.147	12273.79	10586.088	-	7272.529	11318.827	12161.162
User	10427.892	6361.181	11759.6	8181.645	-	445.152	9105.024	8315.343
Sys	2928.975	518.99	2669.207	669.852	-	33.938	2912.633	1277.224

[4]https://sciroc.org/e03-deliver-coffee-shop-orders/

B. Experiment Performance

Table I summarizes the total experimental times of our experiment. It presents the four scenarios and the two options in-memory (M2 aka MERLIN2) and database (DB aka MongoDB). The Mixed mode using MERLIN2 will be done in future iterations because the network used must be prepared to use ROS2 communication mechanisms.

For a fine grain evaluation, Figure 2 shows the performance behavior along the time of three scenarios and the two options: Local, Local Container and Cloud.

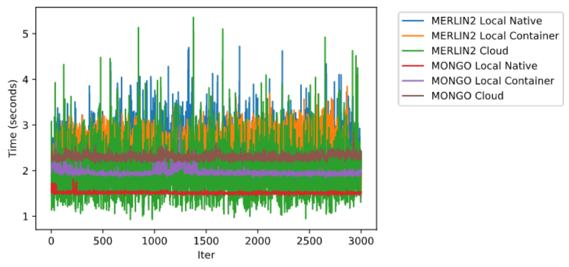

Fig. 2. Comparison of total times of Local Native, Local Container and Cloud scenarios.

IV. Discussion and Conclusions

When thinking about deploying accountability systems for cognitive architectures we have to evaluate the effects of performing the decision-making and knowledge manipulation locally or in the cloud. It is necessary to have in mind the tradeoff between real-time decision-making systems and asynchronous accountability and auditing systems. For these reason, we have evaluate the performance of an architecture that manipulates knowledge in four different scenarios.

Comparing the Local Native scenario with the Local Container scenario, we see that the container scenario performance is lower and requires more time in both the in-memory and MongoDB approaches. MongoDB approach has better performance. Besides, not also the Cloud scenario has times similar to the Local Container option, but it also allows the robot to reduces its workload. Attending the performance stability, we observed that working with MERLIN2 produces a lot of fluctuations whereas using MongoDB is more stable.

On the other hand, the mixed scenario in the MongoDB approach presents very poor results for 60 iterations. On average, each iteration takes more than 120 seconds. At this stage, this implementation is not usable in a real environment. Thus, alternative deployments along with the mixed scenario using an in-memory approach would be explored in the next iterations of this research.

Future research should further develop and confirm these initial findings, not only by replicating the experience a greater number of times but also by measuring other impacts such as CPU performance, memory consumption and network load. Moreover, we would also investigate the performance ratio when running locally in the cloud without a container.

Acknowledgements

This work has been partially funded by the Kingdom of Spain under grant RTI2018-100683-B-I00, and by INCIBE SA under grant Adenda 4.

References

[1] M. Fox and D. Long, “PDDL 2.1: An extension to pddl for expressing temporal planning domains,” *J. Artif. Intell. Res. (JAIR)*, vol. 20, pp. 61–124, 12 2003.

[2] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, *Design Patterns: Abstraction and Reuse of Object-Oriented Design*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, pp. 361–388. [Online]. Available: https://doi.org/10.1007/978-3-642-48354-7_15

[3] M. González-Santamarta, F. J. Rodríguez-Lera, C. Álvarez Aparicio, M. Guerrero-Higueras, and C. Fernández-Llamas, “Merlin a cognitive architecture for service robots,” *Applied Sciences*, vol. 10, no. 17, 2020. [Online]. Available: https://www.mdpi.com/2076-3417/10/17/5989

[4] Y. Lenir, F. Devin, and P. Loubiere, “Cloud resource management using constraints acquisition and planning.” 01 2011.

[5] H. Sun, M. Vukovic, J. Rofrano, and C. Lin, “Advantages and challenges of using ai planning in cloud migration,” 2019.