Dedale: Demonstrating a Realistic Testbed for Decentralized Multi-agents Problems

Cédric Herpson(✉)

Sorbonne Université, CNRS, LIP6, 75005 Paris, France
cedric.herpson@lip6.fr

Abstract. The *Dedale* platform is a peer-to-peer multi-agent testbed dedicated to the study of MAS coordination, learning and decision-making problems under realistic hypotheses: Asynchrony, partial observability, uncertainty, heterogeneity, open environments, limited communication and computation. *Dedale* facilitates the implementation of reproducible and realistic experiments in discrete and (3D) continuous environments. Agents can face cooperative or competitive exploration, patrolling, pickup and delivery, treasure(s) or agent(s) hunt problems with teams of dozens of heterogeneous agents. This paper presents the demonstration elaborated in order to exibit the platform's capabilities.

Keywords: Agent testbed · Coordination · Learning · Decision-making

1 Introduction

Existing multi-agents testbeds make unrealistic hypotheses. They either focus on large-scale complex adaptive systems restricted to synchronous environments with no or few communication capabilities [1], or they assume closed-world environments, homogeneous agents and a perfect vision of the system [4,5]. In both cases, these platforms hypotheses make rendering solutions based on them either ineffective or inoperable in real situations. As a result, researchers working on multi-agent coordination, learning and decision-making problems often use their own (unpublished) toy examples environments which make the results difficult to reproduce. Moreover, they usually do not scale to real-life use-cases and can unfortunately turn out to be over-fitting the proposed algorithms.

Dedale[1] aims to facilitate and improve the experimental evaluation conditions of the developed algorithms and to contribute to the progress of the field towards decentralised solutions able to deal with real-world situations.

This demonstration article first presents the strengths of *Dedale* towards this goal (Sect. 2). We then illustrate the platform key capabilities through the configuration, instantiation and analysis of two use-cases standing on hand-made and real-geographically based environments (Sect. 3) before concluding.

[1] http://dedale.gitlab.io/.

Y. Demazeau et al. (Eds.): PAAMS 2020, LNAI 12092, pp. 426–429, 2020.
https://doi.org/10.1007/978-3-030-49778-1_40

2 Main Purpose

The purpose of *Dedale* is to provide a platform allowing both the research and teaching communities to tackle decentralized problems under parametrizable but realistic hypotheses and environments. In a companion paper, we defined what we called the 8 fallacies of MAS that such a testbed should avoid:

1. Agents take turns executing each other
2. Agents are homogeneous and run at the same speed
3. Agents have access to unlimited resources
4. Agents are reliable
5. Agents are sure
6. Agents have a global and perfect vision of the system
7. Agents number does not change over time
8. Agents communication respects the 8 fallacies[2] of distributed systems.

Dedale is the first platform to avoid all these unrealistic hypotheses. To create it, we combined the well-known Jade framework [2] with the GraphStream[3] and jMonkeyEngine[4] (Jme3) libraries. While Jade is in charge of the MAS management, GraphStream and Jme are respectively handling the two types of environments provided by Dedale: discrete dynamic graphs and continuous 3D-environments. Through the use of *Dedale*'s API, users' agents will have to evolve and cooperate within the chosen environments to accomplish their goals. We currently allow users to choose the type of open-research problems they wants to study among 3 multi-agent classical use-cases of increasing difficulties:

- Distributed exploration [7],
- Cooperative patrolling and pursuit-evasion games [3],
- Treasure(s) hunt & pickup and delivery problems (PDP)[6].

For each of them, the user can define the topology he wants to work on as well as the number and characteristics of the agents. To properly compare the performances of different proposals to a given problem instance, several evaluation metrics are available: 1) The number of messages exchanged between the agents 2) The number of actions executed and 3) The overall time needed to complete a task. In the treasure-hunt case, the quantity of collected resources is also stored.

To further enhance users' interest in studying realistic multi-agent problems with Dedale, the platform is able to function as a node in a network of peer-to-peer Dedale environments. This gives user' agents the ability to test their robustness against environments - and potentially agents - unknown to their designer. The size, complexity and richness of the distributed environment accessible to agents thus becomes virtually unlimited.

[2] The network is secure, reliable, instantaneous, with infinite bandwidth, the topology is fixed and homogeneous, communications costs are non-existent.
[3] http://graphstream-project.org/.
[4] https://jmonkeyengine.org/.

3 Demonstration

As shown in Fig. 1, the platform configuration is done in 2 stages: Definition of the environment in which the agents will evolve then configuration of the agents themselves. Setting up the environment allows to define the type of problem the agents will face. The demonstration aims to present these different aspects.

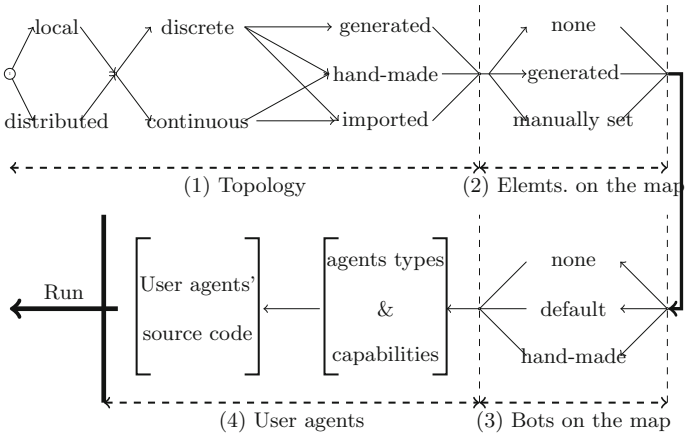

Fig. 1. Dedale platform configuration process. In the upper part the configuration of the environment, in the lower part the configuration of the agents. The 4 steps can be modified independently of each others.

Setting Up an Environment. The user chose to connect or not its environment to the network then select the type of topology he wants (discrete or continuous). In the case of a distributed setup, the choice of the neighbours can be manually set or be left up to the platform. The user can then design, generate, or import from OpenStreetMap a topology. From there he can select the location and type of resources that the environment will posses. We currently offers traps (the agent dies when it touch them) and two types of treasure chest (diamond and gold) with parametrizables detection radius and openning conditions. These may require the cooperation of more than one agent. The setting up of an environment generates two editables files that can easily be modified, shared to, and used by, any Dedale user.

Setting Up Agents. Depending on the use case chosen, the user will choose whether or not to activate the presence of opponents on the map. As detailed in the companion paper, they can be used as intruders for the patrolling case, or as disruptive elements for collecting resources. We will present two of the default behaviors – collecting and moving resources – that make the environment dynamic and decisions uncertain. The user then defines the respective characteristics and capabilities of his agents within the environment (communication radius, transport or resource collection, ...) before linking them to the classes

Fig. 2. Real-time view of agents looking for intruders in Aquila

developed to control their behaviours. Once the simulation started, Dedale's interface (Fig. 2) allows users to follow their agents progress in real-time whether regarding their locations or the generated statistics.

4 Conclusion

This demonstration highlighted the realistic hypotheses and the configuration flexibility of *Dedale*. It presented two of the different classes of open research problems that can currently be studied within 2D-discrete and 3D-continuous environments. Coupled with the execution statistics, this demonstration illustrates how easy it is to set up reproducible experiments and obtain comparative measurements of different solutions to a multi-agent problem with *Dedale*.

References

1. Adil, K., Jiang, F., Liu, S., Jifara, W., Tian, Z., Fu, Y.: State-of-the-art and open challenges in rts game-ai and starcraft. Int. J. Adv. Comput. Sci. Appl. **8**(12), 16–24 (2017)
2. Bellifemine, F.L., Caire, G., Greenwood, D.: Developing Multi-agent Systems with JADE, vol. 7. Wiley, Hoboken (2007)
3. Chevaleyre, Y.: Theoretical analysis of the multi-agent patrolling problem. In: Proceedings of the IEEE/WIC/ACM - IAT, pp. 302–308. IEEE (2004)
4. Hausknecht, M., Mupparaju, P., Subramanian, S., Kalyanakrishnan, S., Stone, P.: Half field offense: an environment for multiagent learning and ad hoc teamwork. In: AAMAS Adaptive Learning Agents (ALA) Workshop (2016)
5. Resnick, C., et al.: Pommerman: a multi-agent playground. arXiv:1809.07124 (2018)
6. Savelsbergh, M.W., Sol, M.: The general pickup and delivery problem. Transp. Sci. **29**(1), 17–29 (1995)
7. Shantanu, D.: Graph explorations with mobile agents. In: Flocchini, P., Prencipe, G., Santoro, N. (eds.) Distributed Computing By Mobile Entities. Lecture Notes in Computer Science, vol. 11340, pp. 403–422. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-11072-7_16