Using parallel computing to improve the scalability of models with BDI agents

Patrick Taillandier, Mathieu Bourgais, Alexis Drogoul, Laurent Vercouter

Using parallel computing to improve the scalability of models with BDI agents

Patrick Taillandier[1], Mathieu Bourgais[2,3], Alexis Drogoul[4,5], and Laurent Vercouter[3]

[1] MIAT, Universit de Toulouse, INRA, 31320 Castanet-Tolosan, France
[2] Normandie Univ, INSA Rouen, UNIHAVRE, UNIROUEN, LITIS, 76000 Rouen, France
[3] Normandie Univ, UNICAEN, UNIHAVRE, UNIROUEN, CNRS, IDEES, 76000 Rouen, France
[4] UMI 209 UMMISCO, IRD/UPMC, Bondy, France
[5] ICTLab, USTH, VAST, 18 Hoang Quoc Viet, Hanoi, Vietnam

Abstract. These last years have seen the development of several extensions of modeling platforms to include BDI agents. These extensions have allowed modelers with little knowledge in programming and artificial intelligence to develop their own cognitive agents. However, especially in large-scale simulations, the problem of the computational time required by such complex agents is still an open issue. In order to address this difficulty, we propose a parallel version of the BDI architecture integrated into the GAMA platform. We show through several case studies that this new parallel architecture is much more efficient in terms of execution time, while remaining easy to use even by non-computer scientists.

Keywords: Parallel Computation, Cognitive Agent, Modeling Platform, Simulation

1 Introduction

The interest of using cognitive and emotional agents in social simulation has been showed by many works [3, 23] and more and more models tend to integrate such complex agents. This spreading is partly due to the integration in the classic agent-based modeling platforms such as Netlogo [24] or GAMA [1, 10] of dedicated cognitive agent architectures that simplify the definition of cognitive agents through their modeling language. The architecture integrated in the GAMA platform is particularly rich: it does not only allow to define cognitive agents, but it links their cognition with an emotional module and a social-relation engine.

However, even with these architectures, an issue still open is the computing time they require. Indeed, as the studied systems are often composed of thousands of entities, the computation time of the proposed tools is a true concern. More-over, most of existing and generic agent-based modeling platforms execute the simulation in a single thread/core (except for the visualization part), whereas

recent computers integrate 4 cores or more, not to mention the possibility to use clusters and grids.

In order to address this problem and to benefit from multi-core architectures during the execution of simulations, we propose in this paper a new version of the BDI cognitive and emotional architecture integrated in the GAMA platform [8, 22, 5], which allows to parallelize in a transparent way most of the required computations.

This article is structured as follows: Section 2 proposes a state of the art on the existing cognitive architectures integrated in simulation platforms as well as the different works that concern the parallelization of agents in simulations. Section 3 presents the parallel version of the cognitive architecture. Section 4 presents experiments carried out with this architecture. Finally, Section 5 provides a conclusion.

2 Related Works

2.1 Cognitive agent architectures

These last years, many cognitive architectures dedicated to agent-based simulations have been proposed. A lot of them are based on the BDI paradigm [7]. This paradigm proposes a straightforward formalization of the human reasoning through intuitive concepts (beliefs, desires and intentions).

In order to ease the development of BDI agents, some works have proposed to integrate such an architecture into a specific framework. The most famous are the Procedural Reasoning System (PRS) [14], JACK [11] and JADEX [17]. Unfortunately, these frameworks still require a high level in computer science and Artificial Intelligence to be used.

To face this difficulty, other authors have proposed to directly integrate a BDI architecture into generic agent-based modeling platforms that are often used for social simulations. For example, an extension of NetLogo proposes a simplified BDI architecture for educational purposes [20]. Agents have beliefs, intentions and ways to answer to intentions. Another integration of the BDI architecture concerns the GAMA platform. The developed architectures have been created to be as complete as possible and available to a non expert public. It gives agents a beliefs base, a desires base, an intentions base and a reasoning engine as well as an emotional engine. Some works have already showed that the use of this architecture ease the modeling of human beings [22, 4]. However, if some previous works such as [23] showed that the architecture enables to simulate the behaviors of thousands of agents, its limitation of execution in a unique thread (and thus one core) raises a scalability question. To face this difficulty, we propose a new version of this architecture that allows to parallelize some of its computations on different computer cores.

2.2 Parallelization of a simulation

If defining an experiment that runs a set of simulations on several computer cores (one simulation per core) is a common practice, in particular with dedicated

frameworks such as OpenMOLE [19], the parallelization of a unique simulation is much more uncommon. A reason is that in agent-based simulations, the agents often interact with many others, which makes difficult the decomposition of a simulation. However, some works propose solutions to tackle this difficulty.

Among these works, some proposes ad'hoc workarounds dedicated to a specific model or to a specific application domain such as [16, 13] and thus cannot be applied for all types of applications.

Other works propose sets of generic algorithms or toolkits such as MCMAS [12] that integrates classic algorithms (diffusion, path-finding...) for agent-based simulation. If this type of toolkits cannot be directly used by modelers that are not computer scientists, they can be used as basic components to improve modeling platforms.

At last, some works propose dedicated environments in which modelers have to respect some specific constraints to define their agents [13, 18, 9]. If these frameworks allow to obtain high scalability results, their use is far beyond the reach of most of modelers, in particular when modelers have to define cognitive agents.

To conclude, if solutions already exit to parallelize a simulation, their lack of genericity and their difficulty of use explain why only few modelers use them.

The work proposed here propose that propose a way to address this issue by offering the possibility to modelers to parallelize in a transparent some of the computations required by the BDI architecture of GAMA.

3 Parallel cognitive and emotional architecture

Our architecture is based on the cognitive architecture integrated in GAMA (*simple_bdi*) [8, 22, 5], which aims at letting modelers use the BDI paradigm to define their cognitive agents. The main strengths of this architecture is to be very versatile (a minimal core with a lot of optional features) and usable even by a non expert public [22, 4] through the use of the GAML language (the modeling language of the GAMA platform). An important objective of this work was to propose a new version of the architecture allowing to parallelize the computation as easy to use as the current one with the minimum of new keywords. Thus, we defined the new architecture such as using it rather than the classic one just requires for the modeler to change the name of the control architecture from *simple_bdi* to *parallel_bdi*.

Like the classic simple BDI architecture, our architecture provides agents with 6 bases:

- **Belief** (what it thinks): the internal knowledge the agent has about the world.
- **Desire** (what it wants): the objectives that the agent would like to accomplish.
- **Intentions** (what it is doing): the desire that the agent has currently chosen to fulfill.

- **Uncertainty** (what it expects to be true): the uncertain events that the agent expects.
- **Emotions** (what it feels): the emotions of the agent. In this work, emotions are based on the OCC theory [15] which means they are seen as a valued answer to the cognitive appraisal of a situation.
- **Social relations** (its relationship with others): the social links the agent has with other agents. Here, social links are defined thanks to the dimensional model of interpersonal relation of Svennevig [21]. A social link contains informations about liking, dominance, solidarity and familiarity towards another agent.

The reasoning engine of the agents with the architecture is also almost similar to the one of simple BDI, except that it integrates the notion of parallelization. Indeed, in GAMA, the agents are scheduled as follow: at each simulation step, every agents are activated one per one according to a given order, which is by default, their order of creation, but that can be simply modified. Once a BDI agent is activated, it executes its complete cycle of reasoning (perceives the world, chooses and executes plans...). One of the major modifications of our new architecture is to split this cycle in sub-steps that can be either parallelized (i.e. distribution of the computation of this sub-step on the different cores of the computer) or not. Thus, each sub-step can be executed sequentially (while keeping the activation order defined by the scheduler) or in parallel (all BDI agents execute the sub-step at the same time).

This reasoning engine is composed of 8 sub-steps (Figure 1).

Step 1. Degradation of predicates: The first step of the reasoning cycle is the degradation of the knowledge of the agent. The predicates stored in the cognitive bases are reduced in lifetime. This step is automatically parallelized.

Step 2. Degradation of emotions: The second step of the reasoning cycle is the degradation of the emotions of the agent. The intensity of agent's emotions are reduced by their decay value. Like Step 1, this step is automatically parallelized.

Step 3. Perception: The third step of the reasoning cycle of the agent is the perception of the environment. The perceptions are described by the modeler through *perceive* blocks. Each perceive block is applied and can participate to update the beliefs of the agent and create social links with other agents. Example of definition of perceive blocks through the GAML language can be found in [22]. As the modeler has the possibility to define interactions with other agents in a perceive block that can introduce a parallel access to a variable (with for instance a simultaneous writing and reading of the variable), we let the modeler choose if he/she wants to parallelize each block or not. By default, all the perceive blocks are parallelized, but a modeler can specify that he/she wants a perceive to be not parallelized by just setting to $false$ the *parallel* facet of the perceive statement.

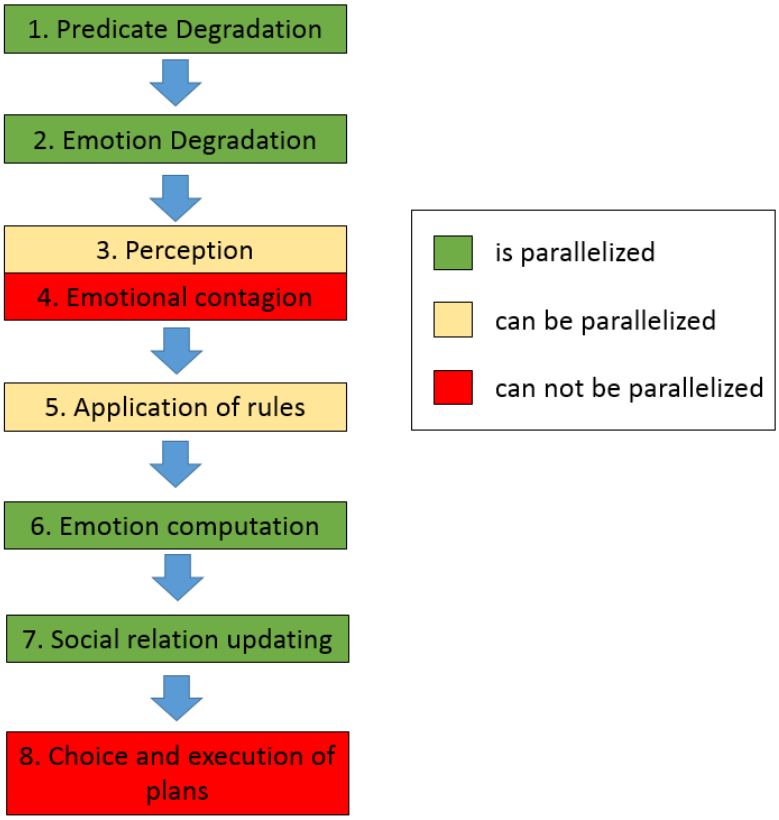

Fig. 1. Schema of our reasoning engine

Step 4. Emotional contagion: The emotional contagion processes are defined inside a perceive block and allows an agent to update its emotions according to the emotions of the nearby agents. As this step requires each agent to simultaneously modify its emotion bases and check the emotion bases of the nearby agents, this step cannot be parallelized. As a consequence, the *parallel* facet of a perceive block with emotion contagion statements inside should always be set to $false$.

Step 5. Inference rules: In this step, the agent applies its inference rules to manage the belief, desire and emotion bases according to its previous beliefs and desires (and eventually to other events). This step gives a dynamic to the overall behavior as the agent can act according to a change in the environment. These inference rules can also be influenced by emotions or social relations. Example of definition of rules through the GAML language can be found in [22] Like for the perceive block, the modeler has the possibility in a rule to define interactions with other agents. So, we let the modeler choose if he/she wants to parallelize

each rule or not. By default, all the rules are parallelized, but a modeler can specify that he/she wants a rule to be not parallelized by just setting the $false$ value to the *parallel* facet of the rule.

Step 6. emotion computation: The emotion computation module, that is optional, allows to automatically create, with no intervention from the modeler, emotions based on the agent's mental state. This creation process is based on the OCC theory [15] and its logical formalism [2] which proposes to integrate this theory in a BDI cognitive architecture. The creation process is detailed in [6]. This step is automatically parallelized.

Step 7. Social relation updating: The social engine is used to dynamically update the social relations of the agent with other agents according to the emotions and the cognition. The computation formulations for the update of social relations is described in details in [6]. This step is automatically parallelized.

Step 8. Choice and application of plans: This step consists in choosing one or several plans and to execute them. The choice of a plan passes through the choice of an intention and then of a plan to achieve this intention. The whole process is influenced by the cognitive bases but also by the emotions and the social relations of the agent and, through the execution of plans, can influence these bases. This cognitive engine is described in details in [22]. As the execution of plans will impact the other agents, this step cannot be parallelized.

This architecture (with its new extension) is already available within the last version (1.7) of the GAMA platform [10]. As stated in the previous paragraphs, modelers can easily use it - with just few lines of codes (by choosing the $parallel_bdi$ control architecture and optionally setting the parallel of some perceive and rule statements to false) - through the GAMA dedicated modeling language.

4 Case Study

In order to validate the parallel BDI architecture, we test its results on the three models provided with the BDI architecture:

- **Gold Miners**: this model concerns gold miners that try to find and extract gold nuggets (see Figure 2). A Miner agent wanders around to find gold nuggets. When it perceives some gold nuggets, it stores this information and begins to extract the closest one then, it brings back the gold nugget to the base and go to search for another gold nugget.
- **Firefighters**: this model concerns the actions of firefighters against fires (see Figure 3). In this model, a firefighter agent patrols, looking for fires. When it finds one, it tries to extinguish it by dropping water, and when it has no more water, it goes to the nearest lake to refill its water tank.

– **City Escape**: this model concerns the evacuation of drivers from a city after an hazard (see Figure 4). In the studied city, a factory of chemical products is on fire and some products are spread into the air. If a driver smells the toxic gas, he/she will think that it is possible that a catastrophe happened so he/she will go to a shelter. Some drivers will not be afraid by the gas but when they will see the fire, they will panic and go to the shelters twice faster than the normal speed. In addition, if a driver sees other drivers fleeing, he/she will have a probability to flee to a shelter.

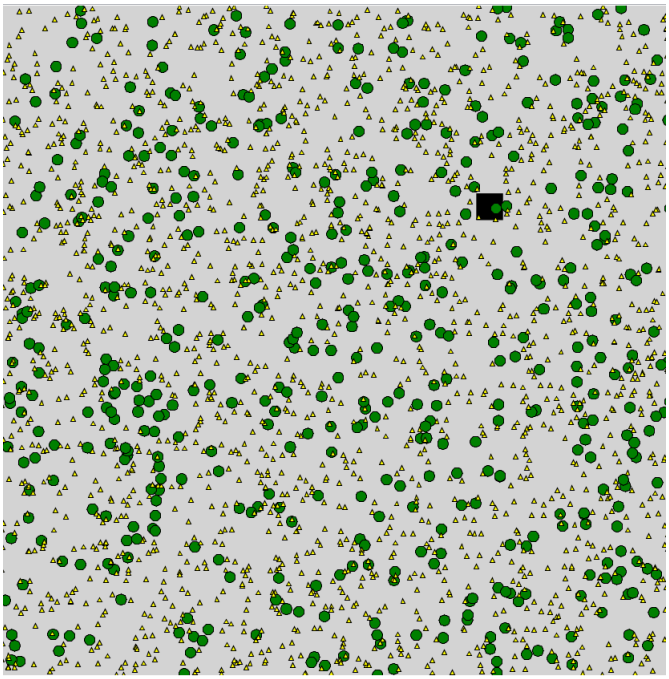

Fig. 2. Snapshot of the Gold Miner model: the green circles are the miners, the yellow triangles the gold nuggets, and the black square the base.

Table 1 gives quantitative information about the 3 models.

The experiment was carried out on a simple i7 windows laptop with 4 cores and 32go of RAM. For each model, we stopped the simulation after 200 simulation steps and we measured the computation time (without the display part - the models are executed in batch mode) and an indicator to illustrate the output difference between the simple BDI architecture and the parallel one. Indeed, as the parallel architecture introduces modifications about the general scheduler of GAMA, it is important to have an idea of the impact of this modification on the simulation output. The indicator was defined as follow for the 3 models:

– **Gold Miners**: number of remaining gold nuggets after 200 simulation steps

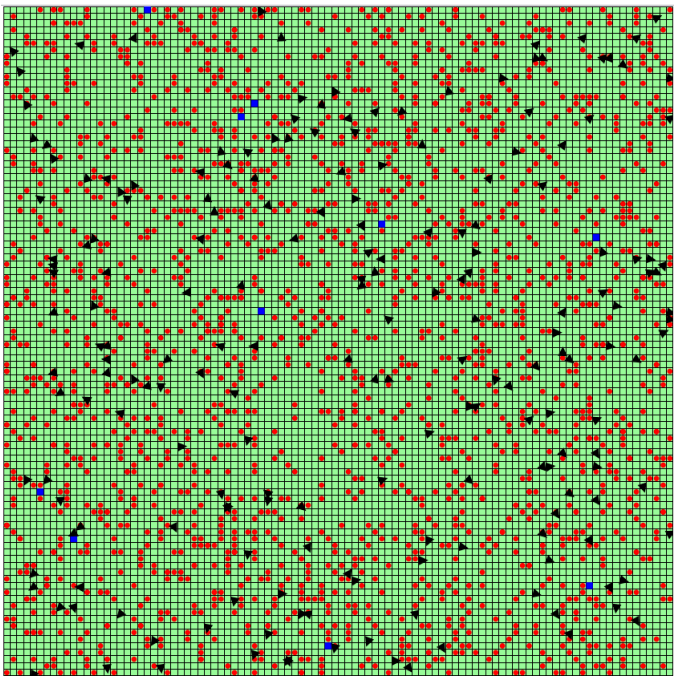

Fig. 3. Snapshot of the Firefighter model: the black triangles are the firefighters, the red circle the fires, and the blue squares the lakes

- **Firefighters**: number of remaining fires after 200 simulation steps
- **City Escape**: number of drivers that have not escaped after 200 simulation steps

As the models are stochastic, we run 10 times each model (using the same series of seeds for the *simple_bdi* and *parallel_bdi* architectures) and we computed the mean values for the computation time and the output indicator.

Table 2 and Table 3 present respectively the results obtained in terms of output indicators and computation time for the 3 models with the two architectures.

Table 1. Quantitative information concerning the 3 models

Data	Gold Miner	Fire fighter	Escape city
number of cognitive agents	500	200	1000
number of plans per agents	4	2	3
number of perceive per agents	1	2	3
number of rules per agents	2	2	2
use of emotions	false	false	true

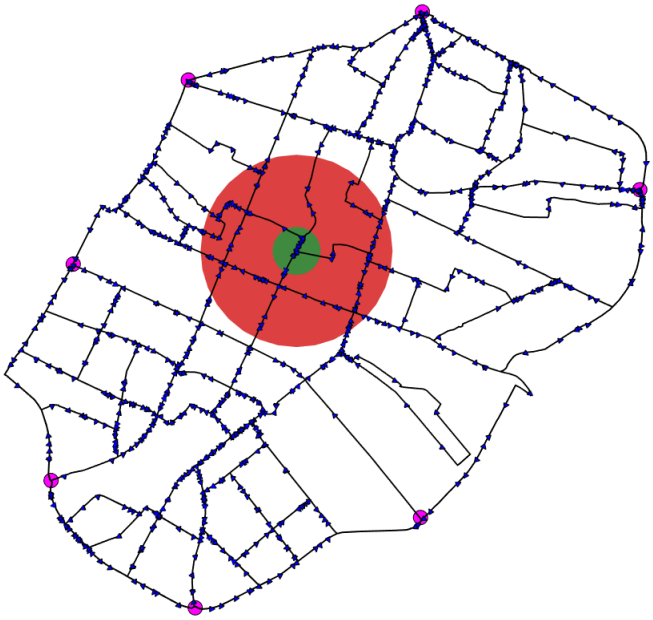

Fig. 4. Snapshot of the Escape city model: the blue triangle are the drivers, the magenta circles the shelters, the green circle the fire perception radius, and the red circle the gas perception radius

The first result that we can observe is that the difference for the 3 models in terms of output indicators is very low, which means that the use of our architecture had a insignificant impact on the simulation results.

In terms of computation time, using the parallel BDI architecture allowed for the two gold miner model to decrease the computation time by 38%, the fire fighter model by 43% and the escape city model by 17%.

The fact that the improvement of the computation times of the escape city model was less important whereas it integrates some features linked to the emotional modules that are parallelized can be explained by the fact that the plans carried out by the agents are more time-consuming. Indeed, the agent have first to compute the shortest path between their current location and their target, and then to move along the polyline roads. It could have been possible to modify a bit the model to integrate the shortest path computation inside the perception rather than in a plan to get a better result, but as the objective of the experiment was to show that the use the parallel architecture allows to improve the results by just changing the control architecture (without modifications of the model), we did not.

Nevertheless, the results obtained are promising considering the fact that the experiment was carried on a simple laptop computer. The improvement of the

computation time will have be far more important on a more powerful computer (with more cores).

Table 2. Output indicator with the 3 models (mean results for 10 simulations)

Model	using *simple_bdi*	*parallel_bdi*
Gold Miner	1299.6	1303.3
Fire fighter	782.7	783.2
Escape city	964	978.2

Table 3. Computation times (in ms) with the 3 models (mean results for 10 simulations)

Model	using *simple_bdi*	*parallel_bdi*
Gold Miner	8746 ms	5394 ms
Fire fighter	10455 ms	5862 ms
Escape city	6084 ms	5033 ms

5 Conclusion

In this paper, we have presented a parallelized version of the cognitive architecture integrated in the GAMA platform. As shown by the experiments carried out on three models, the use of the parallel BDI architecture that just requires to change one or two words in the code allows to significantly decrease the computation time required even on a simple laptop computer without altering the simulation results.

We plan in the coming months to carried out more experiments with more complex models and with a more powerful computer (with more than just 4 cores) in order to better evaluate the impact of the parallel BDI architecture.

In terms of improvements, we plan to work on the parallelization of the emotion contagion step and of the choice and execution of plan step.

Acknowledgment

This work is part of the ACTEUR ("Spatial Cognitive Agents for Urban Dynamics and Risk Studies") research project funded by the French National Research Agency.

References

1. Gama website, 2017.

2. Carole Adam. Emotions: from psychological theories to logical formalization and implementation in a bdi agent, 2007.
3. Carole Adam and Benoit Gaudou. Bdi agents in social simulations: a survey. *The Knowledge Engineering Review*, 31(03):207–238, 2016.
4. Carole Adam, Patrick Taillandier, and Julie Dugdale. Comparing agent architectures in social simulation: Bdi agents versus finite-state machines. In *Hawaii International Conference on System Sciences (HICSS)*, 2017.
5. Mathieu Bourgais, Patrick Taillandier, and Laurent Vercouter. An agent architecture coupling cognition and emotions for simulation of complex systems. In *Social Simulation Conference*, 2016.
6. Mathieu Bourgais, Patrick Taillandier, and Laurent Vercouter. Enhancing the behavior of agents in social simulations with emotions and social relations. In *MABS (to be published)*, 2017.
7. M Bratman. Intentions, plans, and practical reason. 1987.
8. Philippe Caillou, Benoit Gaudou, Arnaud Grignard, Chi Quang Truong, and Patrick Taillandier. A simple-to-use bdi architecture for agent-based modeling and simulation. In *ESSA 2015*, 2015.
9. Nicholson Collier and Michael North. Repast hpc: A platform for large-scale agent-based modeling. *Large-Scale Computing Techniques for Complex System Simulations*, pages 81–110, 2011.
10. Arnaud Grignard, Patrick Taillandier, Benoit Gaudou, Duc An Vo, Nghi Quang Huynh, and Alexis Drogoul. Gama 1.6: Advancing the art of complex agent-based modeling and simulation. In *International Conference on Principles and Practice of Multi-Agent Systems*. Springer, 2013.
11. Nick Howden, Ralph Rönnquist, Andrew Hodgson, and Andrew Lucas. Jack intelligent agents-summary of an agent infrastructure. In *5th International conference on autonomous agents*, 2001.
12. Guillaume Laville, Kamel Mazouzi, Christophe Lang, Nicolas Marilleau, Bénédicte Herrmann, and Laurent Philippe. Mcmas: a toolkit to benefit from many-core architecure in agent-based simulation. In *European Conference on Parallel Processing*, pages 544–554. Springer Berlin Heidelberg, 2013.
13. Nicolas Marilleau, Christophe Lang, Pascal Chatonnay, and Laurent Philippe. Rafale-sp: a methodology to design and simulate geographical mobility. *Stud. Inform. Univ.*, 10(1):38–76, 2012.
14. Karen L Myers. User guide for the procedural reasoning system. *SRI International AI Center Technical Report.*, 1997.
15. Andrew Ortony, Gerald L Clore, and Allan Collins. *The cognitive structure of emotions*. Cambridge university press, 1990.
16. Jon Parker. A flexible, large-scale, distributed agent based epidemic model. In *Simulation Conference, 2007 Winter*, pages 1543–1547. IEEE, 2007.
17. Alexander Pokahr, Lars Braubach, and Winfried Lamersdorf. Jadex: A bdi reasoning engine. In *Multi-agent programming*. Springer, 2005.
18. Paul Richmond, Dawn Walker, Simon Coakley, and Daniela Romano. High performance cellular level agent-based simulation with flame for the gpu. *Briefings in bioinformatics*, 11(3):334–347, 2010.
19. Sebastien Rey-Coyrehourcq Romain Reuillon, Mathieu Leclaire. Openmole, a workflow engine specifically tailored for the distributed exploration of simulation models. *Future Generation Computer Systems*, 29(8):1981 – 1990, 2013.
20. Ilias Sakellariou, Petros Kefalas, and Ioanna Stamatopoulou. Enhancing netlogo to simulate bdi communicating agents. In *Hellenic Conference on Artificial Intelligence*. Springer, 2008.

21. Jan Svennevig. *Getting acquainted in conversation: a study of initial interactions*. John Benjamins Publishing, 2000.
22. Patrick Taillandier, Mathieu Bourgais, Philippe Caillou, Carole Adam, and Benoit Gaudou. A bdi agent architecture for the gama modeling and simulation platform. In *MABS 2016 Multi-Agent-Based Simulation*, 2016.
23. Quang Chi Truong, Patrick Taillandier, Benoit Gaudou, Minh Quang Vo, Trung Hieu Nguyen, and Alexis Drogoul. Exploring agent architectures for farmer behavior in land-use change. a case study in coastal area of the vietnamese mekong delta. In *International Workshop on Multi-Agent Systems and Agent-Based Simulation*, pages 146–158. Springer, 2015.
24. Uri Wilensky and I Evanston. Netlogo: Center for connected learning and computer-based modeling. *Northwestern Univ., Evanston, IL*, 1999.