Decision Making in Agent-Based Models

Guillem Francès[1], Xavier Rubio-Campillo[2], Carla Lancelotti[3], and Marco Madella[4]

[1] Artificial Intelligence Group, Universitat Pompeu Fabra
[2] Barcelona Supercomputing Centre
[3] CaSEs Research Group. Universitat Pompeu Fabra
[4] CaSEs Research Group. ICREA, Universitat Pompeu Fabra and IMF-CSIC

guillem.frances@upf.edu, xavier.rubio@bsc.es
carla.lancelotti@upf.edu, marco.madella@icrea.cat

Abstract. Agent-Based Models (ABM) are being increasingly applied to the study of a wide range of social phenomena, often putting the focus on the macroscopic patterns that emerge from the interaction of a number of agents programmed to behave in a plausible manner. This agent behavior, however, is all too often encoded as a small set of rules that produces a somewhat simplistic behavior. In this short paper, we propose to explore the impact of decision-making processes on the outcome of simulations, and introduce a type of agent that uses a more systematic and principled decision-making approach, based on casting the simulation environment as a Markov Decision Process. We compare the performance of this type of agent to that of more simplistic agents on a simple ABM simulation, and examine the interplay between the decision-making mechanism and other relevant simulation parameters such as the distribution and scarcity of resources. Our preliminary findings show that our novel agent outperforms the rest of agents, and, more generally, that the process of decision-making needs to be acknowledged as a first-class parameter of ABM simulations with a significant impact on the simulation outcome.

Keywords: Agent-Based Modeling, Social Simulation, Model-Based Behavior, Markov Decision Process

1 Introduction and Motivation

Recent years have witnessed a remarkable increase in the use of computer simulation methods and, more specifically, Agent-Based Model simulations, to enhance our understanding of an extremely wide array of social processes, from the emergence of social norms [1] to population dynamics [19], through all sorts of cultural [6], economic [20], or archaeological processes [10]. One of the reasons of this momentum is the fact that simulation stands as a compelling and affordable paradigm for the analysis of complex, highly non-linear environments involving the interaction of heterogeneous entities. Indeed, central to the development of ABM simulations and to the broader notion of complexity theory

is the ambition to explain the emergence of certain regularities at the macroscopic level from the microscopic-level interaction of agents. These agents are generally programmed to behave in a plausible manner, often in the form of a fixed set of simple condition-action rules [13, 2]. However, the plausibility of this type of behavioral strategy remains somewhat problematic, in particular, but not exclusively, when the simulation agents are meant to model human beings [21].

As a matter of fact, the problem of intelligent behavior, *i.e.* of choosing what action to perform next, has been one of the core concerns of Artificial Intelligence (AI) almost since the dawn of the discipline, with the General Problem Solver [12] being at the same time one of the first automated planners and one of the first AI programs. Geffner [8] classifies the different solutions historically used to address this problem into three categories or approaches. In the *programming-based* approach, a human programmer reflects on the characteristics of the problem, devises an *ad-hoc* way of solving it, and expresses this solution as a computer program. In the *learning-based* approach, the behavior is learnt from the experience of past actions and their associated rewards, as in reinforcement learning [18]. Finally, in the *model-based* approach, the behavior is derived from a model of the the world, *i.e.* a formal description of its possible states, the actions that can be performed and the goals to be achieved. As we suggested before, the approach usually employed to define the behavior of ABM simulation agents is the first one, as it offers the advantage of being simple and computationally inexpensive [21]. However, the only way in which this approach can be considered to model intelligent behavior is insofar as it embodies the intelligence of a human programmer. The model-based approach, in contrast, offers a more generic and principled method for the generation of behavior that can be considered intelligent and cognitively more plausible, thus fitting much better the objectives of ABM simulations.

The motivation of the present work is twofold. On the one hand, we aim at exposing the fundamental but seldom recognized affinity between agent-based modeling and AI, framing the problem of deriving the behavior of simulation agents in the context of well-studied model-based planning techniques. Incidentally, this will allow us to provide a generic mechanism where the modeler needs only specify the utility function that should govern the agent behavior, and let the actual behavior be automatically derived. On the other hand, we aim at exploring the impact of different decision-making strategies on the actual outcome of simulations, and check if the use of these more sophisticated (and computationally expensive) AI techniques pays off, not only conceptually and theoretically but also empirically, thus producing significantly different outcomes. We hypothesize that the use of different decision-making mechanisms can radically affect macro-level indicators (such as the carrying capacity of the simulated environment) that are frequently used for the analysis of emergent phenomena.

Outline of the paper. The remainder of this paper is organized as follows. The next section offers a brief account of Markov Decision Processes and of

the UCT algorithm, which form the basis of the novel type of ABM agent we present. Section 3 describes a simple ABM model that we put to work in order to evaluate the impact of different decision-making strategies and compare them to this novel type of agent, and Section 4 discusses some preliminary empirical results. Finally, Section 5 concludes the paper and outlines some ideas for future research.

2 Model-Based Behavior

2.1 Model-based planning and Markov Decision Processes

The alternative to traditional rule-base behavior that we propose is based on *finite-horizon* Markov Decision Processes (MDPs). In a nutshell, these are fully-observable, stochastic state models where the objective is to find a suitable policy of action that maximizes the expected reward that can be accumulated in a fixed number of timesteps, the so-called *horizon* of the problem. MDPs have been widely used and studied in several fields, from artificial intelligence to operational research and economics [4, 18], but to the best of our knowledge this work constitutes the first attempt to use them in the context of agent-based models and social simulations. The basic idea is to cast the simulation environment as an MDP and automatically derive the behavior of each simulation agent by selecting at each time step the action that best suits her interests, suitably defined through a utility function. Formally, a finite-horizon MDP is defined by *(i)* a set S of possible states of the world, *(ii)* an initial state $s_0 \in S$, *(iii)* a set $A(s)$ of actions that can be applied in each state $s \in S$, *(iv)* transition probabilities $P_a(s'|s)$ that encode the probability of transitioning from state s to state s' when the action $a \in A(s)$ is applied, and, finally *(v)* a utility or reward function $r : S \times A \to \mathbb{R}$ that models the agent interest by specifying the reward $r(s, a)$ obtained by applying action $a \in A(s)$ when s is the actual state of the world.

2.2 The UCT Algorithm

In order to choose the adequate action in an MDP, we employ the UCT algorithm [9], an anytime optimal algorithm [3] for finite-horizon MDPs that is guaranteed to converge to the optimal sequence of actions when given enough time. Being one of the most popular Monte-Carlo Tree Search methods [5], UCT successfully tackles extremely large state spaces by running a number of stochastic simulations from the initial state of the problem that help building incrementally a partial search tree containing the most promising nodes. The algorithm has been empirically proven to excel at finding an adequate balance between the *exploitation* of actions that are believed to offer the highest reward and the *exploration* of actions that appear to be sub-optimal but might emerge as better options when sufficiently explored. For a more thorough discussion on UCT and Monte-Carlo Tree Search methods, we refer the interested reader to [5]; for the purpose of this work, it suffices to note that the two parameters of the algorithm

that are relevant to our simulations are the planning *horizon* h and the number of stochastic simulations run from the initial state, which we call the *width* w of the algorithm.

3 Model Description

We next describe a simple Sugarscape-like model [7] that we have designed and implemented on top of the Pandora simulation framework [17] in order to test the different decision-making mechanisms that we consider.[5]

3.1 Resource distribution and dynamics

Agents interact in a 50×50 grid-like resource map where each map cell contains an amount of resources between 0 and a maximum that depends on the particular cell. These per-cell maximum values are spatially autocorrelated, meaning that the value of each cell relates to that of neighboring cells, following a standard ecological model of resource distribution in which spatial autocorrelation is a key feature that adapted foraging strategies need to take into account [11]. The higher the autocorrelation factor we use, the more clustered the map resources are. At the beginning of the simulation all map cells start at their maximum amount of resources. Whenever this amount is diminished by the action of agents, each cell increases one amount of resources per timestep, up to its maximum.

3.2 Agent dynamics

Agents are basic resource-accumulating entities, and start the simulation at random map locations. At each time step, they can either remain in their current cell or move to one of the 8 neighboring cells (for a total of 9 possible actions, diagonal moves are allowed). After the move, each agent collects from her current cell an amount of resources which is distributed uniformly between 1 and the resources available on the cell. After the resource collection, agents consume a fixed amount of resources λ, a simulation parameter intended to model resource scarcity. If the total amount of resources accumulated by an agent is less than λ, the agent dies; if, on the contrary, this amount surpasses a certain threshold value (currently 20λ), the agent gives birth to a new agent, which will be located in the same cell, and both agents see their amount of resources set to a fixed value 5λ.

3.3 Agent behavior

We examine a number of possible decision-making strategies to choose among the 9 possible actions. We first consider a baseline `random` agent that chooses

[5] The model, implemented in `C++`, can be downloaded from `https://github.com/gfrances/model-based-social-simulations/releases/tag/eumas2014`.

uniformly at random between the available actions. Second, we consider a `greedy` agent that chooses among the 9 possible destination cells the one with the highest amount of available resources, breaking ties at random. We also consider a `lazy` agent that only moves whenever the current cell does not satisfy her needs, *i.e.* when the amount of resources in the cell multiplied by a certain slack parameter α ($0 < \alpha \leq 1$) is less than the agent's daily consumption requirements. In that case, the agent moves to the first satisfactory cell, according to a random ordering; in case none of the 9 possible destinations is satisfactory, a random action is chosen. Finally, we consider a novel `MDP`-based agent, which we describe more in detail next.

3.4 Modeling the world as a Markov Decision Process

As previously mentioned, the decision-making process of an `MDP` agent is based on choosing the optimal action according to a specific utility function and to the evaluation performed by the UCT algorithm on an MDP model of the world that is constructed by each agent at each timestep. The states of the MDP contain information regarding *(i)* the position of the agent, *(ii)* the amount of resources held by the agent, and *(iii)* the availability of resources in each cell of the map. The initial state of the MDP is derived from the actual state of the world in the current time step, and the transition probabilities between states are given by the simulation dynamics described above, the only stochasticity arising from the resource recollection process. Most relevantly, the utility function of the agent is designed to strongly penalize those states in which the agent is dead, and otherwise is proportional to the amount of resources held by the agent. It is important to note that at this stage, the presented MDP model does not take into account the indirect competition of other agents that might be consuming resources from neighboring cells.

4 Experiment Design and Empirical Results

4.1 Assessing the Impact of UCT Parameters

Before discussing the fully multi-agent simulations, and in order to calibrate the width and horizon parameters of the UCT algorithm discussed in Section 2, we first run some single-agent (only one agent, no agent reproduction) simulations, measuring the amount of resources that the agent is able to accumulate over time. To simplify things, we only explore moderate resource consumption factors $\lambda \in \{2, 3\}$, and fix the map autocorrelation factor to 25. We examine the performance of `MDP` agents using varying horizon ($h \in \{2, 4, 6, 8, 10, 12\}$) and width ($w \in \{50, 100, 500, 1000, 5000\}$), running simulations with the agent starting in a number of different random locations that are consistently the same for the different combinations of values of w and h. We expect the amount of accumulated resources to grow with both the allowed width and horizon. The results of these simulations, not shown here for the sake of brevity, are not entirely consistent

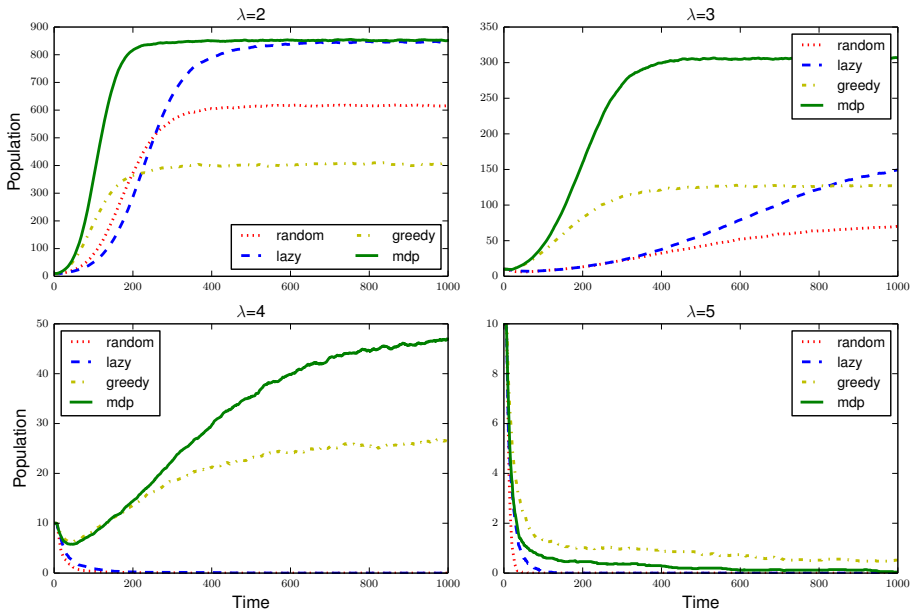

Fig. 1. Population dynamics for the four types of agents under different resource scarcity conditions λ.

for the lower width values, which do not permit a sufficient exploration of the search tree. For higher values $w \in \{1000, 5000\}$, however, the amount of accumulated resources slightly increases as the horizon grows, although the differences are not significant. Because of this, and since the computation time required by UCT increases with both the width and horizon, we stick with an intermediate combination $\langle h = 8, w = 1000 \rangle$ for the remainder of our experiments.

4.2 Comparative Performance

We now turn to compare `MDP` agents with the other decision-making strategies that we consider in multi-agent simulations. We have run a number of simulations comparing the four decision-making strategies under varying values of the resource scarcity parameter $\lambda \in \{2, 3, 4, 5\}$ and resource autocorrelation factor $autocorrelation \in \{1, 10, 25\}$. Each subplot in Figure 1 shows the population growth for the 4 agent types, averaged over 50 runs on 5 different randomly generated maps with a fixed resource autocorrelation value of 25. As expected, resource scarcity has a big impact on population growth for all types of agents: as the value of λ increases, resources are more scarce and the total population achieved by any agent type sharply decreases, with $\lambda = 5$ simulations being hardly able to sustain any agent. For the remaining values of λ, we note that the carrying capacities for different agents vary broadly, and that the `MDP` agent

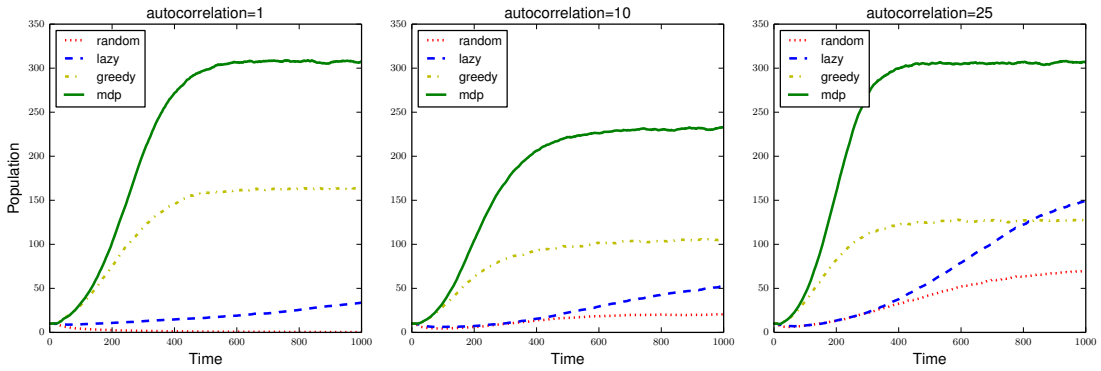

Fig. 2. Population dynamics for the four types of agents under different resource distribution conditions.

outperforms the rest of agents by a large margin. In general, the population of `greedy` agents increases more rapidly than that of `lazy` and `random` agents, although the carrying capacity of the system for this type of agent is lower in some contexts. This is due to the fact that many `greedy` agents located in nearby cells will tend to overpopulate the same cell if its amount of resources is higher than that of the neighbors, whereas `random` and `lazy` agents will tend to disseminate more over the available space.

Figure 2 focuses on the impact of resource distribution on the performance of agents, with each subplot showing the results of 50 runs on maps generated with resource autocorrelation factors $1, 10, 25$, for a fixed value of $\lambda = 3$. Interestingly, `random` and `lazy` agents perform better as the resources of the map tend to be more clustered, but the same does not hold for `greedy` and `MDP` agents. In the case of `greedy` agents, a more uniform distribution of resources, might help overcome the negative effects of their myopic nature; in the case of `MDP` agents, the way in which the clusterization of resources affect population dynamics is not entirely clear from the results of this experiment, and deserves further examination.

5 Conclusions

We have presented a preliminary examination of the use of sound and principled model-based AI techniques to handle the problem of decision making in ABM simulations, in an attempt to bridge the large and (to our opinion) inexplicable gap between the two disciplines. Our empirical findings show that agents employing these techniques adapt to the simulation environment significantly better, and that this holds irrespective of resource distribution and scarcity issues. Due to the exploratory nature of this work, however, we have put aside a large number of issues that deserve further analysis. Coupling the UCT algorithm with a base policy that exploits the particular characteristics of the simulation model, for instance, should be a straight-forward manner to improve the efficiency of

MDP agents — exploring discretization strategies in order to reduce the size of the state space should be another. More relevantly, the possibility of linking the use of model-based techniques to the systematic analysis of the role of bounded rationality (both as bounded information and bounded complexity [14]) in ABM simulations, on the one hand, and the stimulus posed by ABM simulations to develop truly multi-agent planning techniques [8], on the other, constitute, in our opinion, promising areas for future research.

Acknowledgments This research is part of the SimulPast Project (CSD2010-00034) funded by the CONSOLIDER-INGENIO2010 program of the Spanish Ministry of Science and Innovation. The implementation of MDP agents relies on Blai Bonet's `mdp-engine` library[6]. Resource maps were generated using the R statistical environment [16] and *gstat* package for geostatistical analysis [15].

References

1. Axelrod, R.: An evolutionary approach to norms. American Political Science Review 80(4), 1095–1111 (1986)
2. Bandini, S., Manzoni, S., Vizzari, G.: Agent based modeling and simulation: An informatics perspective. Journal of Artificial Societies and Social Simulation 12(4), 4 (2009), http://jasss.soc.surrey.ac.uk/12/4/4.html
3. Bonet, B., Geffner, H.: Action selection for MDPs: Anytime AO* versus UCT. In: Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence (2012)
4. Boutilier, C., Dean, T., Hanks, S.: Decision-theoretic planning: Structural assumptions and computational leverage. Journal of Artificial Intelligence Research 1, 1–93 (1999)
5. Browne, C.B., Powley, E., Whitehouse, D., Lucas, S.M., Cowling, P.I., Rohlfshagen, P., Tavener, S., Perez, D., Samothrakis, S., Colton, S.: A survey of Monte Carlo Tree Search methods. IEEE Transactions on Computational Intelligence and AI in Games 4(1), 1–43 (2012)
6. Epstein, J.M.: Generative social science: Studies in agent-based computational modeling. Princeton University Press (2006)
7. Epstein, J.M., Axtell, R.: Growing artificial societies: Social science from the bottom up. Brookings Institution Press (1996)
8. Geffner, H., Bonet, B.: A concise introduction to models and methods for automated planning. Synthesis Lectures on Artificial Intelligence and Machine Learning 8(1), 1–141 (2013)
9. Kocsis, L., Szepesvári, C.: Bandit based Monte-Carlo planning. In: 17th European Conference on Machine Learning. pp. 282–293. Springer (2006)
10. Lake, M.: Trends in archaeological simulation. Journal of Archaeological Method and Theory 21(2), 258–287 (2014)
11. Legendre, P.: Spatial autocorrelation: Trouble or new paradigm? Ecology 74(6), pp. 1659–1673 (1993)
12. Newell, A., Shaw, J.C., Simon, H.A.: Report on a general problem-solving program. In: Proc. of the Int. Conf. on Information Processing. pp. 256–264 (1959)

[6] Available at https://code.google.com/p/mdp-engine/, the implementation of the UCT algorithm that we use is described in [3].

13. O'Sullivan, D.: Complexity science and human geography. Transactions of the Institute of British Geographers 29(3), 282–295 (2004)
14. Papadimitriou, C.H., Yannakakis, M.: On complexity as bounded rationality. In: Proc. of the 26th ACM symposium on theory of computing. pp. 726–733 (1994)
15. Pebesma, E.J.: Multivariable geostatistics in S: the gstat package. Computers & Geosciences 30, 683–691 (2004)
16. R Development Core Team: R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna, Austria (2008), http://www.R-project.org, ISBN 3-900051-07-0
17. Rubio-Campillo, X.: Pandora: A versatile ABM platform for social simulation. In: Sixth International Conference on Advances in System Simulation. IARIA (2014)
18. Russell, S., Norvig, P.: Artificial intelligence: A modern approach. Prentice Hall (2010)
19. Schelling, T.C.: Dynamic models of segregation. Journal of mathematical sociology 1(2), 143–186 (1971)
20. Tesfatsion, L.: Agent-based computational economics: A constructive approach to economic theory. Handbook of computational economics 2, 831–880 (2006)
21. Wellman, M.P.: Putting the agent in Agent-Based Modeling (2014), http://web.eecs.umich.edu/srg/wp-content/uploads/2014/08/transcript.pdf, talk from the 13th Int. Conf. on Autonomous Agents and Multi-Agent Systems (AAMAS-14)