Doctoral Thesis Proposal

Autonomous Interactive Agents for Population Ecology

Aureliano Rama
University of Pisa, Italy

April 14th, 2010

1 Introduction

In this work I'm proposing the creation of a formalism (a language) and a graphical notation to easily express autonomous agents with unique identities, abilities and actions, all of which can change through time and through interactions with other agents. While many example of similar formalisms and notations already exists (see "State of the Art", [4], [10], [9], [7], [6], [25], [14], [16], [34], [30], [35]) they are mostly devised and used by computer scientists, with little to no contact with the biologists and chemists that are supposed to be the final users. Hence many of these preexisting tools are not used in real case study, despite their great potentials.

In the first chapter I'm going to describe the state of the art, a few related works that explore the same field I'm proposing to study, to see what it has already been done and what there's still to do. In the second, I'll be exploring what naturalists have to say on the subject of population ecology and what their approach to simulation is. Finally I'll detail the open problems my work is going to address and describe my proposal.

2 State of the Art

While a comprehensive model is something that every scientist dream about [33], in the past computational and methodological limitations were forcing researcher to concentrate on limited scope problems [13]. Here I list some of the principal methods along with their results and problems.

2.1 Dynamic Input/Output Automata [6]

This work presents a model for an automaton that has the capability of creating others. While this is certainly not a new concept in the programming field (where objects create and destroy objects all the time [3], [1], [2]), its application to a state-machine model is interesting and different: a mathematical automaton is normally intended as static, which means that once created to do a task, that's the only task it'll ever do. While a Turing machine is in principle capable of rewriting its own code, this feature is rarely used in real life application, mainly for its complicated semantics.

A Dynamic I/O Automaton (DIOA) model is a mathematical state-machine model for defining and analyzing dynamic systems of interacting components. The systems considered are dynamic in two senses: (1) components can be created and destroyed as computation proceeds, and (2) the events in which the components may participate may change. This model admits a notion of external system behavior, based on sets of traces. It also features a parallel composition operator for dynamic systems, which respects external behavior, and a notion of simulation from one dynamic system to another, which can be used to prove that one system implements the other.

The DIOA model is defined to support the analysis of mobile agent systems and can also be used for other forms of dynamic systems, such as systems described by means of object-oriented programs, and systems containing services with changing access permissions.

The basic idea is to extend the I/O automaton model with special create actions, and combine such extended automata into global configurations. The DIOA model admits a notion of external system behavior, based on sets of traces. It also features a parallel composition operator for dynamic systems, which respects external behavior and satisfies standard execution projection and pasting results, and a notion of simulation relation from one dynamic system X to another dynamic system Y , which can be used to prove that X implements Y .

The model here can create and destroy automata but the list of the possible actions, while it can be infinite, it cannot change over time: once the automaton is created, its set of actions is immutable. This feature, included mostly for simplicity, set a clear limitation of this model in its usefulness at representing biological and chemical objects whose property are often unknown or even impossible to determine beforehand.

2.2 Artificial Biochemistry [10]

In this work, the authors investigate stochastic automata collectives where by a collective they mean a large set of interacting, finite state automata. This is not quite the situation we have in classical automata theory, because here they are interested in automata interactions. It is also not quite the situation with cellular automata [33], because these automata are interacting, but not necessarily on a regular grid. And it is not quite the situation in process algebra, because they are interested in the behavior of collectives, not of individuals.[32] And in contrast to Petri nets, they model separate parts of a system separately.

Stochastic here means that automata interactions have rates. These rates induce a quantitative semantics for the behavior of collectives, and allow them to mimic chemical kinetics. Chemical systems are, physically, formed by the stochastic interactions of discrete particles. For large number of particles it is usually possible to consider them as formed by continuous quantities that evolve according to deterministic laws, and to analyze them by ordinary differential equations (ODEs). However, one should keep in mind that continuity is an abstraction, and that sometimes it is not even a correct limit approximation. In biochemistry, the stochastic discrete approach is particularly appropriate because cells often contain very low numbers of molecules of critical species: that is a situation where continuous models may be misleading. Stochastic automata collectives are hence directly inspired by biochemical systems, which are sets of interacting macromolecules, whose stochastic behavior ultimately derives from molecular dynamics.

This approach is sound with regards to external effects, as the great accent on behaviors and the large use of Control theory language prove. At the same time the possibility of building a chemical machine from simpler and well known parts is a great improvement respect to the past.[9] Yet, most of this behavior refinement is obtained through sophisticated and artificial tweaking of connections and rates, thus losing the direct connection with the entity the model is trying to simulate: it's quite difficult (if not often simply impossible) to link one or more of the automata to a single cell with the same behavior.

2.3 Mobile Ambients [11]

This work may seems only slightly related to this proposal since it focuses on mobility of computational agents in a network of computers. Yet, many of the actual considerations and observations done here are seamlessly applicable to biological cells since mobility and coherence are two of their main features.[34]

The inspiration for this work comes from the potential for mobile computation over the World-Wide Web. The geographic distribution of the Web naturally calls for mobility of computation, as a way of flexibly managing latency and bandwidth. Because of recent advances in networking and language technology, the basic tenets of mobile computation are now technologically realizable. The high-level software architecture potential, however, is still largely unexplored, although it is being actively investigated in the coordination and agents communities.

The main difficulty with mobile computation on the Web is not in mobility per se, but in the handling of administrative domains. In the early days of the Internet one could rely on a flat name space given by IP addresses; knowing the IP address of a computer would very likely allow one to talk to that computer in some way. This is no longer the case: firewalls partition the Internet into administrative domains that are isolated from each other except for rigidly controlled pathways. System administrators enforce policies about what can move through firewalls and how.

Mobility requires more than the traditional notion of authorization to run or to access information in certain domains: it involves the authorization to enter or exit certain domains. In particular, as far as mobile computation is concerned, it is not realistic to imagine that an agent can migrate from any point A to any point B on the Internet. Rather, an agent must first exit its administrative domain (obtaining permission to do so), enter someone elses administrative domain (again, obtaining permission to do so) and then enter a protected area of some machine where it is allowed to run (after obtaining permission to do so). Access to information is controlled at many levels, thus multiple levels of authorization may be involved. Among these levels we have: local computer, local area network, regional area network, wide-area intranet and internet. Mobile programs must be equipped to navigate this hierarchy of administrative domains, at every step obtaining authorization to move further. Similarly, laptops must be equipped to access resources depending on their location in the administrative hierarchy. Therefore, at the most fundamental level we need to capture notions of locations, of mobility and of authorization to move.

2.3.1 Ambients

An ambient is a bounded place where computation happens. The interesting property here is the existence of a boundary around an ambient. If we want to move computations easily we must be able to determine what should move; a boundary determines what is inside and what is outside an ambient.

An ambient is something that can be nested within other ambients. As we discussed, administrative domains are (often) organized hierarchically. If we want to move a running application from work to home, the application must be removed from an enclosing (work) ambient and inserted in a different enclosing (home) ambient. A laptop may need a removal pass to leave a workplace, and a government pass to leave or enter a country.

An ambient is something that can be moved as a whole. If we reconnect a laptop to a different network, all the address spaces and file systems within it move accordingly and automatically. If we move an agent from one computer to another, its local data should move accordingly and automatically.

More precisely, we investigate ambients that have the following structure:

- Each ambient has a name. The name of an ambient is used to control access (entry, exit, communication, etc.). In a realistic situation the true name of an ambient would be guarded very closely, and only specific capabilities would be handed out about how to use the name. In our examples we are usually more liberal in the handling of names, for the sake of simplicity.

- Each ambient has a collection of local agents (also known as threads, processes, etc.). These are the computations that run directly within the ambient and, in a sense, control the ambient. For example, they can instruct the ambient to move.

- Each ambient has a collection of subambients. Each subambient has its own name, agents, subambients, etc.

In all of this, names are extremely important. A name is:

- something that can be created, passed around and used to name new ambients.

- something from which capabilities can be extracted.

2.3.2 Relationship to this proposal

As soon as we translate the internet jargon to a more biological related one, the relationship of this work with my proposal is immediate. As said, an agent can be though of as a cell or a protein, something that has some "computational" ability along with space-time constraints (that meant that it has boundaries on its allowed positions and on the rate it can produce results). The notion of an ambient is quite naturally translated, since most of the actual "computation" in biological systems is obtained by means of separating cells and objects from one another and barriers are present almost anywhere: chemical barriers (i.e. acidity differential), biological barriers (membranes) and physical barriers (electromagnetic repulsion) are quite common. Administrative authorization in this context is a little more tricky to understand, since there's no notion of an external programmer to rely to, but we can interpret that as the chemical and biological compatibility that any such agent must have to interact with its environment.

Without going into too many details, it's quite obvious that the results obtained in this field can be used in the biological domain. Such a claim is reinforced by the authors themselves in another work, explained in next section.

2.4 BioAmbients [34]

Biomolecular systems, composed of networks of proteins, underlie the major functions of living cells. Compartments play an essential role in the functioning of such systems, by organizing them in a hierarchical and modular way. In order to perform its function, a molecule must be present in the right location. Localization of molecules to specific compartments is a key regulatory mechanism in diverse biological systems.[17], [24] Compartmentalization and movement across boundaries play a critical role in computational systems as well as seen with the ambient calculus above.

The authors have previously developed an abstraction for biomolecular systems using the π-calculus process algebra [11], which successfully handled their molecular and biochemical aspects of biological systems, but provided only a limited solution for representing compartments.

In this work, they extend this abstraction to handle compartments and, as said before, they are motivated by the ambient calculus, a process algebra for the specification of process location and movement through computational domains. The authors then present the Bioambients calculus, which is suitable for representing various aspects of molecular localization and compartmentalization, including the movement of molecules between compartments, the dynamic rearrangement of cellular compartments, and the interaction between molecules in a compartmentalized setting. Since BioAmbients contains the stochastic π-calculus, the authors discuss only the additional entities and operations. The full syntax, congruence laws and semantics are given.

The calculus detailed provides basical abstract operations on compartments like merging, splitting, fusion and complex formation. It provides also a mean to model movement with respect to the compartments (entering and exiting).

2.5 Calculus of Looping Sequences [7]

The Stochastic Calculus of Looping Sequences (SCLS+) are suitable to describe microbiological systems, such as cellular pathways, and their evolution. Systems are represented by terms and the terms of the calculus are constructed by basic constituent elements and operators of sequencing, looping, containment and parallel composition. The looping operator allows tying up the ends of a sequence, thus creating a circular sequence which can represent a membrane. The evolution of a term is modeled by a set of rewrite rules enriched with stochastic rates representing the speed of the activities described by the rules, and can be simulated automatically.

2.6 Hierarchical State Machine [5]

Finite state machines (FSMs) are widely used in the modeling of systems in various areas. Descriptions using FSMs are useful to represent the flow of control (as opposed to data manipulation) and are amenable to formal analysis such as model checking. In the simplest setting, an FSM consists of a labeled graph whose vertices correspond to system states and edges correspond to system transitions. In practice, to describe complex systems using FSMs, several extensions are useful such as communicating FSMs in which the system is described by a collection of FSMs that operate concurrently and synchronize with one another periodically, There is a rich body of theoretical results concerning complexity and expressiveness of state machines. By labeling the edges of an FSM with alphabet symbols and by introducing initial and final states, FSMs can be used to define regular languages. Analysis problems of interest include emptiness of the language, model checking with respect to temporal requirements, and inclusion and equivalence of the languages of two machines. For a single FSM, the complexity of some problems depends on whether the machine is deterministic or not.

While the impact of features such as nondeterminism and concurrency on complexity and expressiveness of finite-state machines is well understood, there is almost no literature on understanding the impact of introducing hierarchy in state machines. In hierarchical (nested) FSMs, the states of an FSM can be ordinary states or superstates which are FSMs themselves. The notion of hierarchical FSMs was popularized by the introduction of Statecharts [20], and exists in various object-oriented software development methodologies such as Room and the Unified Modeling Language (Uml). Hierarchical state machines have two descriptive advantages over ordinary FSMs. First, superstates offer a convenient structuring mechanism that allows us to specify systems by stepwise refinement, and to view it at different levels of granularity. Second, by allowing sharing of component FSMs, we need to specify components only once and then can reuse them in different contexts, leading to modularity and succinct system representations.

2.6.1 Dynamic Hierarchical Machine [25]

In this paper the authors introduce Dynamic Hierarchical Machines (DHMs), which extend HSMs by allowing communication and dynamic activation of machines. Any DHM M_1 can send to a concurrent DHM M_2 a third DHM M_3, which starts running either in parallel with M_1 and M_2, or inside M_2, depending on a symbol put on the transition performed by M_2. At any stage of the computation, DHMs are identical to HSMs. The two models differ because the structure of DHMs can change dynamically. The authors present the formal semantics of DHMs, given in Structural Operational Semantics (SOS) style and from the format of the SOS transition rules they derive a congruence result for trace equivalence.

3 Population Ecology

What have naturalists, biologists and chemists been working on, when it comes to simulations of large population of interactive agents, like animals, whose behavior is not only affected by the environment but also by other animals and whose actions often are able to modify the parameters of the simulation, i.e. by modifying the environment itself?

Since a long time ago, most of them have been approaching this from two different prospectives ([18], [13], [17], [23], [22], [24], [36]): a classical one, using differential equations and a newer one using individual-based modeling. The distinction is between "pragmatic" motivation, which uses the individual-based approach as a tool without any reference to the theoretical issues which have emerged from the classical state variable approach and "paradigmatic" motivation, which explicitly refers to theoretical ecology. Models which use individuals as a basic unit have occasionally been used in ecology since the 1970s, but only since the late 80s has individual-based modeling been an explicitly delineated approach of ecological modeling.[17] "Individual-based modeling" (IBM) refers to simulation models that treat individuals as unique and discrete entities which have at least one property in addition to age that changes during the life cycle, e.g. weight, rank in a social hierarchy, etc.

3.1 Individual Based Approaches [13]

How much and in what way does individual behavior affect population dynamics? The two different approaches to modeling population dynamics, classical and individual based, give no clear answer to this question. In classical population models such as the logistic equation, no reference is made to behavior at all. Instead, everything is reduced to the population growth rate f(N). Classical population models have been successful in devising and demonstrating important concepts such as density dependence or intra- and interspecific competition. Classical models had, however, only limited success in making ecology more predictive. They have produced understanding in the sense of possible explanations of observed phenomena, but the model predictions are not testable as a rule, and therefore, classical population models do not directly contribute to the scientific cycle of hypotheses, predictions, and testing. One reason for this could be that the level of aggregation of the classical models is chosen a priori instead of testing which elements of a real population should be considered in more detail. Individual-based models, on the other hand, are bottom-up approaches that start at the bottom level of population ecology, that is, at the individual level. Individual-based models have, at least in principle, the potential to determine what individual properties and what elements of an individuals performance are essential for generating the characteristic features of the overall population dynamics. This approach includes the possibility of taking spatial dynamics and the dynamics of abiotic factors explicitly into account. However, if the research program offered by individual-based models is followed, methodological and practical problems may arise with respect to model resolution, parameter assessment, and computation time. The resolution of a model should be similar in all parts of the model, but often detailed knowledge about individuals is only partly available. This makes it difficult to extrapolate individual-based models from the behavioral timescale to the population timescale, that is, from fast changing variables describing behavior to slowly changing variables describing population dynamics. For example, for most fish species, the processes controlling recruitment are not well understood. Thus, most individual-based models about fish show the growth and development of only one cohort but not the entire population dynamics. Parameter assessment is the next problem. Individual-based models are supposed to be more testable because they are closer to reality, but more reality means more parameters and, in turn, more effort to determine these parameters. With individual behavior in a heterogeneous, randomly fluctuating environment in particular, the number of parameters will soon exceed manageable limits. In the end, computation time may explode as soon as one tries to model behavior because of the short time increments on which behavior operates. Thousands of individuals behaving to, say, 50 behavioral rules operating with time increments of days or even hours results, even with fast computers, in computation times that make the thorough analysis of the models almost impossible.

3.2 Pattern-Oriented Modeling [18]

We are surrounded by autonomous and adaptive agents: cells of the immune system, plants, citizens, stock market investors, businesses, etc. The agent-based complex systems (ACSs) around us are made up of myriad interacting agents. One of the most important challenges confronting modern science is to understand and predict such systems. Bottom-up simulation modeling is one tool for doing so: we compile relevant information about entities at a lower level of the system, formulate theories about their behavior, implement these theories in a computer simulation, and observe the emergence of system-level properties related to particular questions. Bottom-up models have been developed for many types of ACSs, but the identification of general principles underlying the organization of ACSs has been hampered by the lack of an explicit strategy for coping with the two main challenges of bottom-up modeling: complexity and uncertainty. Consequently, model structure often is chosen ad hoc, and the focus is often on how to represent agents without sufficient emphasis on analyzing and validating the applicability of models to real problems.

3.3 A Review of Ten Years of Individual-based Modeling in Ecology [17]

Each modeler who builds and analyses an individual-based model learns of course a great deal, but what has ecology as a whole learned from the individual-based models published during the last decade? It is suggested that in order to orient individual-based modeling more towards general theoretical issues, we need increased explicit reference to theoretical ecology and an advanced strategy for building and analyzing individual-based models.

To this end, this work presents a heuristic list of rules which may help us to advance the practice of individual-based modeling and to learn more general lessons from individual-based modeling in the future than we have during the last decade.

The main ideas behind these rules are as follows: (1) Individual-based models usually make more realistic assumptions than state variable models, but it should not be forgotten that the aim of individual-based modeling is not "realism" but modeling. (2) The individual-based approach is a bottom-up approach which starts with the "parts" (i.e. individuals) of a system (i.e. population) and then tries to understand how the system's properties emerge from the interaction among these parts. However, bottom-up approaches alone will never lead to theories at the systems level. State variable or top-down approaches are needed to provide an appropriate integrated view, i.e. the relevant questions at the population level.

What have we learned? Everybody who has used IBMs has of course learned a lot from their models, but nevertheless it is hard to say what ecology as a whole has learned from the individual-based approach.

Why do modelers use the individual-based approach? There are two mutually dependent motivations: one is that important features of individuals are not taken into account in state variable models (e.g. individual variability) and the other is that the theory emerging from state variable approaches has severe deficiencies. The first motivation may be referred to as "pragmatic" because it emphasizes pragmatic attitude that IBMs simply add a new tool to the toolbox of ecological modeling, whereas the second motivation may be referred to as "paradigmatic" because it emphasizes the attitude that something might be wrong with the paradigms of classical theoretical population ecology. Pragmatic IBMs study problems that cannot be addressed with state variable models, such as the dynamics of size hierarchies in plant monocultures, or recruitment in fish populations depending on size-dependent predation or feeding success. Although IBMs driven by pragmatic motivation are of course theoretical studies, the range of theoretical problems addressed is rather narrow. No reference is made to theoretical issues that are decisive in classical theoretical population ecology, e.g. regulation, persistence, resilience, or density-dependence. Paradigmatic motivation, on the other hand, means that IBMs are driven by the suspicion that much of what we have learned from state variable models about the theoretical issues mentioned above, e.g. regulation etc., would have to be revised if the discreteness, uniqueness, life cycles and variability of individuals were to be taken into account, as well as the fact that most interactions are local and that space matters. IBMs driven by paradigmatic motivation are thus studied with respect to the concepts and general lessons of classical theoretical population biology. The distinction between pragmatic and paradigmatic is not an objective distinction between different kinds of models, but a distinction of the general motivation behind the model.

4 Open Problems and Proposed Work

4.1 Problems and Goals

The main goal is to obtain an easy to use formalism (a language of some kind) to describe large populations of interactive individual, whose identity is different from one to another and who interact both with each other and with their environment. This language must be intuitive and simple, so that complex systems can be modeled even from scientists without a computer science background. We would aim also to develop automatic tools for this language: syntax analyzers, simulators, model checkers, graphical interfaces, etc...

4.2 Achieved Results and Experience

In 2009, in cooperation with dr. Zuffi from the Calci Natural Museum, I created a very simple simulator to study how a population of Emys Orbicularis (pond turtle) evolved through the years, given certain rates of predation and reproduction and given some environmental conditions. To be noted is the fact that not every needed parameter was known and so the model was meant also to try to infer these data. While the program itself is quite simple, the experience showed two interesting characteristic: first of all, it demonstrated how little the methods of population ecology are spread in the natural science community, since this very simple individual-based simulation was a hit in an international conference on herpetology; second, it showed us which are the requirements when dealing with this kind of problems: 1) unknown parameters are unavoidable, 2) some measure of the population must always exists to judge the results and, even more important, 3) the direct involvement of natural scientists is essential in all the phases of the simulation.

The results of the simulation are shown in figure 1, where we can see how the population remains dynamically stable over a period of 500 years, by carefully tweaking the unknown parameters: and since this kind of population dynamic is the one that is actually observed in nature, our result both gave us confidence in the simulator and also allowed us to have a good estimation of the formerly unknown parameters.

4.3 Methods and Results

A sketch of the formalism, both graphical and as a language, has already been created, although still in a very early stage of development. This work is based on the principal characteristics of the systems that we want to model: large populations of unique individuals, each similar but not identical to the others, with capabilities that change in time and whose interactions with each other influence their short and long term behaviors and their enclosing environment. In the beginning, the formalism will be an ad-hoc solution for this kind of simulations; subsequently, we plan to extend it step by step by introducing other interesting and useful sides, like the possibility of having probability ranges and

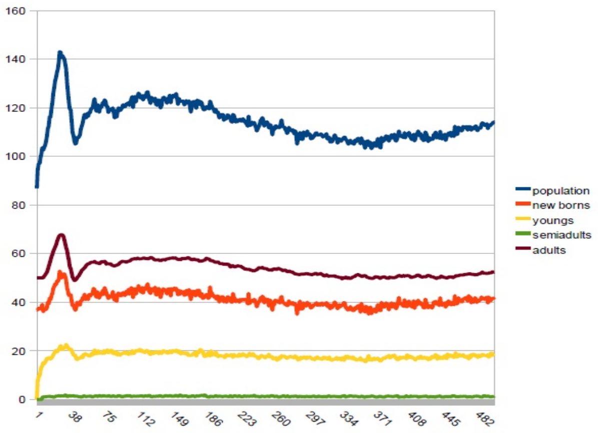

Figure 1: Emys Simulator results for a 500 year period.

spatiality (to model populations distributed in an environment whose individual can move through environments). The last part of my work will be to develop some analysis tools and techniques (like general purpose simulators based on the formalism, translators into other languages, model checkers, etc...) Different case studies are needed in order to verify both the ease of use and the usefulness of this proposed formalism. To this aim, we have two ongoing cooperations with biologists and natural scientists: the one with dr Zuffi explained before and another one (yet to be launched) to create model for the selection mechanisms of the Thropeus Moori (a species of fish native to central Africa lakes).

Starting from the Emys simulator, I build a possible schema of FSMs that represented the male and the female turtles in their life cycle, interacting for reproduction and thus producing nests which in turn produces newborns. uch a schema is shown in fig. 2. This FSMs can change state through interaction with others, or by an internal move. In both cases, the arc can augmented with the rate representing the probability that such a state change happens with respect with all the other possible state changes ready to happen in the system. The communication channel is represented with an arrow with a white head linking two arcs, meaning that change state is brought by the communication event happening. The dashed arrow represent a side effect of the state change: through that state change, a new instance of the automata pointed by the dashed arrow is created and inserted in the system.

Note that although the schema represent each and every possible automata in the system and its behavior, as usual the system can have multiple copies of each one of them, with the detail that each copy is slightly different, because of some kind of private (or internal) unique name. Even more, because these instances can start their life at different times thus interacting with different sets of automata in the system, even without the rates their development can differ.

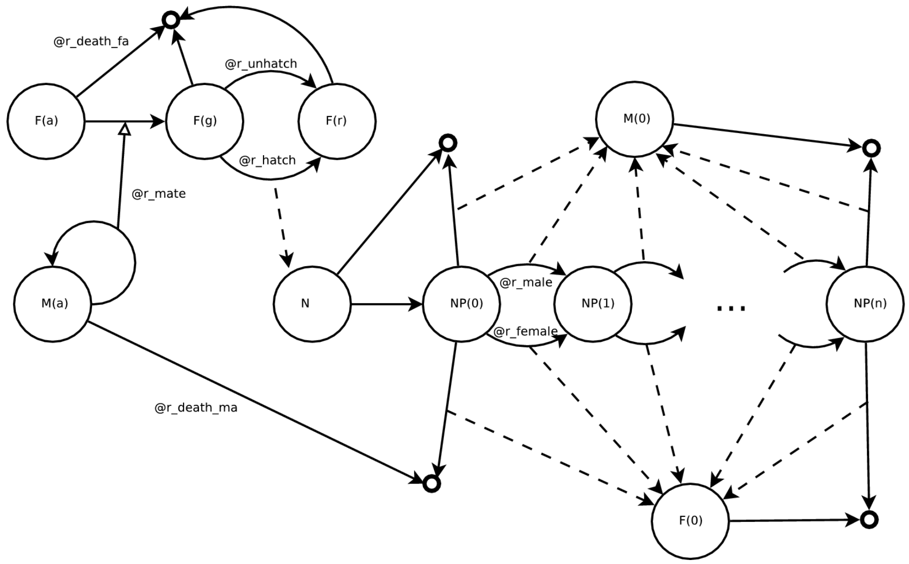

Figure 2: Finite State Automata with special arcs for communications and creations.

Another peculiarity of these FSMs is that they will be hierarchical. At the very least, the system will be partitioned in large containers of groups of slightly different group of automata, with only one container at the time being the working one and with rules to move control from each automata in a container to another automata in the next. There will have to be rule of development for these containers, thus making them another (special or not) automata in itself.

An important feature that will be present are the global broadcast signals: most (if not all) of the automata in the system will be able to receive these signals, that originate somewhere (probably from another automata, possibly some special ones, maybe the container automata, see fig 3) and instead of being delivered to a single receiver they get delivered to all the individuals in the system. This broadcast signals provoke some global state change (not shown in the picture), like the changing of time (days, seasons, years, etc...) or they could be responsible for some collective behavior that originates from single individuals, like a storm that fly away because of a single warning launched from a sentry.

A proposed change for these broadcast signals would be to have them adorned with a scope: we could have the whole system divided in two subsystem, lake and land, exchanging automata and communicating only through a small frontier (the shore line), see fig. 4. Then both the lake and the land could be divided in small colonies of animals, with each colony communicating and exchanging

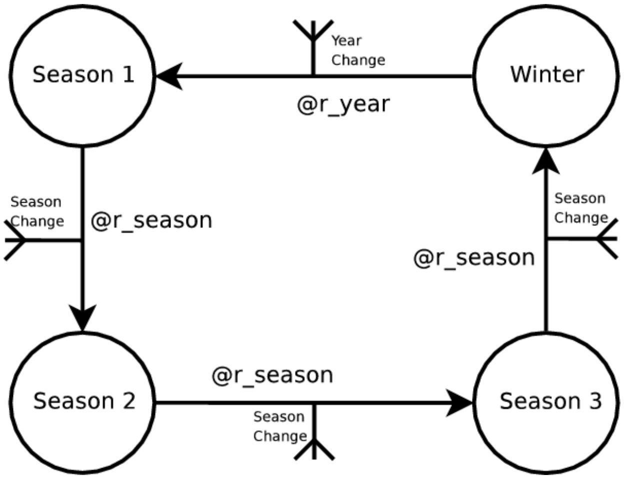

Figure 3: Finite State Automata emitting broadcast signals.

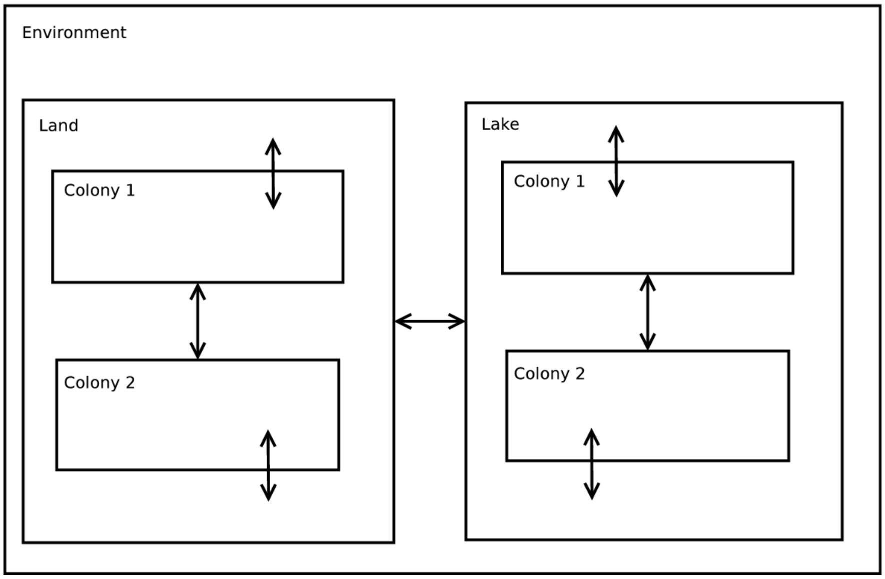

Figure 4: Containers for automata grouping and broadcast signals scoping.

automata through small trails. This way, a sentry could warn its own colony with a broadcast without that signal reaching the next colony. Or a strong sound could warn all the land colonies without it propagating in the lake. The broadcast signals could be adorned with the scope of their broadcast, getting through to that container (and its underlings) but not to any outside of it or incorporating it.

Another proposed feature, more syntactic than the previous ones, could be to augment the creating side effect with a multiplicity scope thus making the "..." part in fig 2 useless.

4.4 Impact

What impact such formalism could achieve?

The question is non trivial, since so many alternatives are already available. Yet, as of now, only a handful of natural scientists, chemists and biologists really use one of these tools, hence making their availability irrelevant. By working side by side with these researchers and by observing how they work and what they need to do, I plan on crafting a mechanism that, while retaining its formal value and its computer science interest, is really adapted to what the final users require, in this case natural scientists who have to build and run simulations and projections.

By means of formal methods, like model checking, I plan to build formalism and automatic tools that could achieve that synthesis between the "pragmatic" and the "paradigmatic" aspect of simulation mentioned in [17]. In order to do this, this framework must help the scientist to extract knowledge and infer structure from individual-based models with hundreds of parameters and thousands of individuals.

Such an achievement would most probably be very useful to field scientists, to wild-life preservers, to natural scientists that aim to re-integrate fauna in an open environment, etc... And with the help of the case studies, such a framework could be spread to a large audience of scientists that still today rely on manual analysis, an error prone and very time consuming activity.

References

[1] M. Abadi and L. Cardelli. A theory of primitive objects: Second-order systems. In *In Proc. ESOP'94, European Symposium on Programming*, pages 1–25. Springer-Verlag, 1994.

[2] M. Abadi and L. Cardelli. A theory of primitive objects: Untyped and first-order systems. In *Proc. Theoretical Aspects of Computer Software*, pages 296–320. Springer-Verlag, 1994.

[3] M. Abadi and L. Cardelli. *A theory of objects*. Springer, 1996.

[4] M. I. Aladjem, S. Pasa, S. Parodi, J. N. Weinstein, Y. Pommier, and K. W. Kohn. Molecular interaction maps–a diagrammatic graphical language for bioregulatory networks. *Science's STKE*, 2004(222):pe8, 2004.

[5] R. Alur, S. Kannan, and M. Yannakakis. Communicating hierarchical state machines. In *in 26th International Colloquium on Automata, Languages and Programming (ICALP*, pages 169–178. Springer-Verlag, 1999.

[6] P. C. Attie and N. A. Lynch. Dynamic input/output automata: a formal model for dynamic systems (extended abstract). In *In CONCUR01: 12th International Conference on Concurrency Theory, LNCS*, pages 314–316. Springer-Verlag, 2001.

[7] R. Barbuti, A. Maggiolo-Schettini, P. Milazzo, G. Pardini, and A. Rama. A process calculus for molecular interaction maps. *CoRR*, abs/0911.4854, 2009.

[8] R. Barbuti, A. Maggiolo-Schettini, P. Milazzo, and L. Tesei. Timed p automata. *Fundam. Inform.*, 94(1):1–19, 2009.

[9] L. Cardelli. Abstract machines of systems biology. *Transactions on Computational Systems Biology*, 3737:145–168, 2005.

[10] L. Cardelli. Artificial biochemistry, 2009.

[11] L. Cardelli and A. D. Gordon. Mobile ambients. In *In Proceedings of POPL'98*. ACM Press, 1998.

[12] R. Fagin, J. Y. Halpern, and M. Y. Vardi. What can machines know? on the properties of knowledge in distributed systems. *Journal of the ACM*, 39:328–376, 1996.

[13] L. Fahse, C. Wissel, and V. Grimm. Reconciling classical and individual-based approaches in theoretical population ecology: A protocol for extracting population parameters from individual-based models. *The American Naturalist*, 152(6):838–852, 1998.

[14] C. Fournet, G. Gonthier, J.-J. Lvy, L. Maranget, D. Rmy, and I. Rocquencourt. A calculus of mobile agents. pages 406–421. Springer-Verlag, 1996.

[15] D. T. Gillespie. Exact stochastic simulation of coupled chemical reactions. *Journal of Physical Chemistry*, 81(25):2340–2361, 1977.

[16] A. Girault, B. Lee, and E. A. Lee. Hierarchical finite state machines with multiple concurrency models. *IEEE Transactions on Computer-aided Design of Integrated Circuits and Systems*, 18:742–760, 1999.

[17] V. Grimm. Ten years of individual-based modelling in ecology : what have we learned and what could we learn in the future? *Ecological modelling*, 115, 1999.

[18] V. Grimm, E. Revilla, U. Berger, F. Jeltsch, W. M. Mooij, S. F. Railsback, H.-H. Thulke, J. Weiner, T. Wiegand, and D. L. DeAngelis. Pattern-Oriented Modeling of Agent-Based Complex Systems: Lessons from Ecology. *Science*, 310(5750):987–991, 2005.

[19] J. Y. Halpern and Y. Moses. Knowledge and common knowledge in a distributed environment. *Journal of the ACM*, 37:549–587, 1984.

[20] D. Harel. Statecharts: A visual formalism for complex systems, 1987.

[21] J. Hillston. A compositional approach to performance modelling, 1996.

[22] K. W. Kohn. Molecular interaction maps as information organizers and simulation guides. *CHAOS*, 11(1):84–97, 2001.

[23] K. W. Kohn and M. I. Aladjem. Circuit diagrams for biological networks. *Molecular Systems Biology*, 2006.

[24] K. W. Kohn, M. I. Aladjem, J. N. Weinstein, and Y. Pommier. Molecular interaction maps of bioregulatory networks: A general rubric for systems biology. *Molecular Biology of the Cell*, 17:1–13, 2006.

[25] R. Lanotte, A. M. Schettini, A. Peron, and S. Tini. Dynamic hierarchical machines.

[26] D. Lepri. A formal semantics for molecular interaction maps. Master Thesis in Computer Science, University of Pisa, 2008.

[27] P. Milazzo. *Qualitative and Quantitative Formal Modeling of Biological Systems*. PhD thesis, Computer Science Department - University of Pisa, 2007.

[28] R. Milner. *A Calculus of Communicating Systems*, volume 92 of *Lecture Notes in Computer Science*. Springer, 1980.

[29] R. Milner. The pi calculus and its applications (keynote address). In *IJCSLP*, pages 3–4, 1998.

[30] R. Milner. *Communicating and mobile systems: the pi-calculus.* Cambridge University Press, New York, NY, USA, 1999.

[31] R. Milner, J. Parrow, and D. Walker. A calculus of mobile processes, part i. *I and II. Information and Computation*, 100, 1989.

[32] R. Milner, J. Parrow, and D. Walker. A calculus of mobile processes, i. *Inf. Comput.*, 100(1):1–40, 1992.

[33] M. Mitchell. *An introduction to genetic algorithms.* A Bradford book. MIT Press, Cambridge, Mass. [u.a.], 1996.

[34] A. Regev, E. M. Panina, W. Silverman, L. Cardelli, and E. Y. Shapiro. Bioambients: an abstraction for biological compartments. *Theor. Comput. Sci.*, 325(1):141–167, 2004.

[35] A. C. Uselton and S. A. Smolka. A compositional semantics for statecharts using labeled transition systems. In *In CONCUR '94, vol. 836 of LNCS*, pages 2–17. Springer-Verlag, 1994.

[36] Y. B. Zion, G. Yaari, and N. M. Shnerb. Optimizing metapopulation sustainability through a checkerboard strategy. *PLoS Comput Biol*, 6, 2010.