Composability: Perspectives in Ecological Modeling

Ozan Kahramanoğulları[1], Ferenc Jordán[1], and Corrado Priami[1,2]

[1] The Microsoft Research – University of Trento, Centre for Computational and Systems Biology
[2] Department of Information Engineering and Computer Science, University of Trento

Abstract. The multiplicity of ecological interactions acting in parallel calls for novel computational approaches in modeling ecosystem dynamics. Composability, a key property of process algebra-based models can help to manage complexity and offer scalable solutions in ecological modeling. We discuss and illustrate how composability of process algebra language constructs can be used as a language aid in the construction of complicated ecosystem models.

Keywords: ecology, modeling, stochastic process algebra, BlenX.

1 Introduction

The systems approach to biology [Kitano, 2002] is now broadly established. Catalyzed by the advances in computational capabilities and the introduction of promising technologies from various disciplines, formal modeling and analysis methodologies are now becoming one of the common instrument-ensembles in biological research. The contribution of the systems point of view to the experimental biology is twofold. Firstly, formal models enforce a rigorous representation of the biological knowledge. This results in disambiguous descriptions of the mechanistic behavior of the biological systems under study. Secondly, simulation and analysis with formal models often provide insights into implicit aspects of the biological systems, and deliver predictions that help biologists to design further experiments.

The algorithmic approach to systems biology [Priami, 2009], driven by the application of core computer science technologies, is based on describing the capabilities of the components of biological systems and their interactions in terms of discrete state spaces. The topological structure and quantitative aspects of algorithmic models mediate various simulation and analysis techniques that are adapted from computer science, and shed light to the mechanistic understanding of the biological systems they model. For example, stochastic simulations provide a means to observe the emergent behavior of the modeled systems, while static analysis capabilities, such as reachability queries on the state space, help to address topological properties.

One of the underlying metaphors of algorithmic systems biology is the perception of biological systems as complex, reactive, information processing systems, where system components interact with each other in diverse ways, and generate new patterns of interaction. Such a consideration makes concurrency theory an appropriate formal framework for studying biological systems with respect to the parallel, distributed and mobile interactions exhibited by these systems. In this regard, the field of *process algebra* provides the principles for defining specific programming primitives and draws the guidelines for designing algorithms that are tailored for biology [Regev and Shapiro, 2002]. In particular, *composability* of the algebraic operators, which is commonly exploited in modeling of computer systems with process algebras, becomes also instrumental while building biological models. This is because composability makes it possible to specify the meaning of a system component in terms of its components and the meaning of the algebraic operator that composes them. As a result of this, each component of a biological system can be modeled independently, allowing large models to be constructed by composition of simple components. Moreover, because one can work on individual components, modifications to the dynamics of a model can be made locally on the appropriate component of the model without modifying the rest of the model.

Although the pioneering efforts in systems biology can be attributed to the differential equation models of Lotka and Volterra of the predator-prey interactions in fisheries [Lotka, 1927,Volterra, 1926], the systems approach is rather underrepresented in ecology in comparison to molecular biology (see [Ulanowicz, 1986,Platt et al., 1981] for early discussions on ecological processes). This can be partly due to diverging considerations of ecosystems, on one hand, in terms of general principles and universal laws, and on the other hand, in terms of phenomena that emerge as a result of vastly parallel, stochastic processes, driven by local rules. This latter perspective, which is closer to the algorithmic approach to systems biology, is emphasized in ecology within *individual based models* (IBM) [DeAngelis and Gross, 1992,Grimm, 1999,Grimm et al., 2006] or *agent based* models (ABM). Another discussion in ecological modeling with parallels to both IBMs and algorithmic systems biology is based on the consideration of ecosystems as complex adaptive systems in which patterns at higher levels emerge from localized interactions and selection processes acting at lower levels [Levin, 1998].

IBMs build on the observations above by emphasizing the ideas that individuals of an ecosystem are different and the interactions between individuals take place locally. Based on these assumptions, IBMs describe populations of systems in terms of discrete and autonomous individuals with distinguished properties. Models built this way are then studied by tracking their individuals, also in terms of their collective behavior through space and time. The aim here is to understand the implications of the local interactions to the whole system with respect to the emerging patterns of behavior during stochastic simulations, and this way link mechanisms to behaviors [Seth, 2007].

The ideal scenario in IBMs is that a model with as little detail as possible reveals as much as possible during simulation. As in molecular biology models, in IBMs there is often a trade-off between simpler, more abstract models and models that reflect more aspects of reality. The main challenge here is to summarize the knowledge on nature accurately into a model, also by resorting to an appropriate level of abstraction. The model should capture the key aspects of each individual's capabilities in terms of its interactions with others and the environment while remaining simple enough for a fruitful analysis. However, IBMs pose this challenge with an additional twist: while the complexity of cellular processes comes mostly from how many (and how many kinds of) molecules interact, an important component of ecological complexity is how many ways components can interact with each other. In ecosystems, several types of interactions act in parallel and they are also in interaction with each other (e.g. [Billick and Case, 1994]). Understanding and modeling the interactions of interactions, as well as finding appropriate common currencies for their quantification are among the most important motives in community and systems ecology [Vasas and Jordán, 2006].

Having emerged as an area of computer science, process algebras profit from a theoretical foundation that provides a rich arsenal of formal techniques and tools as well as a broadly expanding culture of software engineering. In the following, we argue that stochastic process algebra languages may contribute to ecology models [Priami and Quaglia, 2004,Priami, 2009], partly resolving the challenges that confront IBMs. As an evidence for this, we present process algebra representations of ecosystem models and primitives, where composability is the essential ingredient for extending and refining models at different levels, and for designing specialized modeling interfaces. For the models, we use the stochastic process algebra language BlenX [Dematté et al., 2008,Dematté et al., 2010].

2 The BlenX Language

In this section, we provide a brief introduction to stochastic process algebras, in particular, the BlenX language.

Process algebras are formal languages, which were originally introduced as a means to study the properties of complex reactive systems. In these systems, concurrency, that is, the view of systems in which potentially interacting computational processes are executing in parallel, is a central aspect. Due to their capability to capture such a form of concurrency, the process algebra languages qualify as appropriate tools for describing the dynamics of biological systems [Regev and Shapiro, 2002].

BlenX [Dematté et al., 2008,Dematté et al., 2010] is a stochastic process algebra language that shares features with stochastic pi-calculus [Priami, 1995] and Beta-binders [Degano et al., 2005]. As these other members of the family of process algebra-based languages, BlenX has a strong focus on the interactions of entities. BlenX is explicitly designed to model biological entities and their interactions. It is a stochastic language in the sense that the probability and speed of the interactions and actions are specified in the programs that are written in this language. In this respect, BlenX provides an efficient implementation of the Gillespie algorithm [Gillespie, 1977], the semantics of which is given by continuous time Markov-chains. BlenX is a part of the software platform *CoSBiLab*.

In BlenX, each individual is given with an abstract entity that we call a *box*. Each box has a number of connectivity interfaces called *binders*, and it is equipped with an internal program. The sites of interaction are represented as binders on the box surface. For example, in Figure 1, each box has only one binder. Binders are identified by their names, e.g., `x` and their types, e.g., `X`.

The mechanism, realized by the interfaces and the internal program govern the interactions of the box and their effects on the box: this mechanism describes a number of possible actions with which the individual can evolve to a new state possibly by interacting with others or on its own. At each simulation step, the simulation engine picks an action of the model in a manner which is biased by the rates of the actions with respect to the Gillespie algorithm. This gives rise to a model behavior in the form of a sequence of model actions that can be read as a time series, depicting the behavior of the model components.

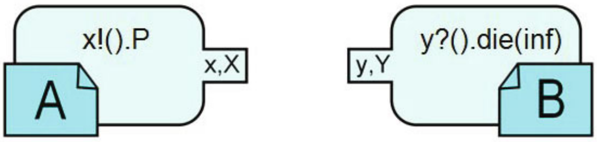


Fig. 1. Two BlenX boxes representing two interacting species A and B

A BlenX model consists of two parts, where the first part contains a description of all the boxes of the model, together with their binders. The second part of the model contains a list of compatibilities of different binders with respect to their types. With respect to the compatibilities described in this part, binders can bind or unbind to binders of other boxes, or perform communications with them to exchange information. For example, with respect to the model given in Figure 1, the compatibility expression $(X, Y, 0, 0, 1)$ indicates that the binders with types X and Y can communicate with a rate 1. The third and forth parameters of this expression state the binding and unbinding rates between these types, which are 0 in this case.

A box can stochastically interact with another box, and change state as a result of this interaction with respect to the actions specified in its internal program. Alternatively, a box can autonomously change state by stochastically performing an action that is given in its internal program. For instance, the interaction of a predator A and its prey B can be described in a BlenX model with the boxes depicted in Figure 1. The interaction rate, specified in the BlenX code, determines the rate of the predation being modeled. The internal program, which can be `nil`, describes this interaction and its consequences in terms of the *actions* the box can undertake. The `nil` action does nothing. Other stochastic *actions* that a BlenX box can perform are summarized as follows: a box can

(*i.*) communicate with another box (or with itself) by performing an input action, e.g., `x?(message)` that is complementary to the output action, e.g., `x!(message)`, of the other box, or vice versa, and this way send or receive a message;
(*ii.*) perform a stochastic `delay` action;
(*iii.*) change (`ch`) the type of one of its interfaces;
(*iv.*) eliminate itself by performing a `die` action;
(*v.*) `expose` a new binder;
(*vi.*) `hide` one of its binders;
(*vii.*) `unhide` a binder which is hidden.

In addition to these actions, there are also other programming constructs available such as if-then statements and state-checks. For example, let us consider the box A in Figure 1. We can define the program P such that it changes the type X to Z if this box is bound to another speices via its interface x:

```
if (x,X) and (x,bound) then ch(x,Z) endif
```

In BlenX, following the process algebra tradition, we can compose actions by using algebraic composition operators to define increasingly complex behaviors. We can sequentially compose actions by resorting to the prefix-operator, which is written as an infix dot. For instance, `ch(x,Z).hide(x).nil` denotes a program that first performs change action and then hides the changed binder. Programs can be composed in parallel. Parallel composition (denoted by the infix operator `|`, for instance `P|Q`) allows the description of programs, which may run independently in parallel and also synchronize on *complementary actions* (i.e., *input* and *output* over the same channel). Programs can also be composed by *stochastic choice*, denoted with the summation operator "+". The sum of processes `P` and `Q`, `P + Q` behaves either as `P` or as `Q`, determined by their stochastic rates, and selection of one discards the other forever.

In BlenX, we use *events*, which are programming constructs for expressing actions that are enabled by global conditions. For example, in ecosystem models, we use the `new` construct to introduce new individuals of a species to the system, for instance, to model migration or birth, or to implement global influences on the model individuals such as change of seasons.

3 Composability as a Modeling Aid

As we illustrate in the previous section, in BlenX, model expressions are written by resorting to the algebraic notion of composition. In a model, the capabilities of an individual is described by composing atomic BlenX actions within the internal program of the box that models the individual. This way, we can define each model individual in terms of its potential behavior with respect to its interactions

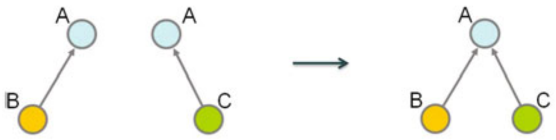


Fig. 2. Composition of two predator-prey interactions, providing an apparent competition module [Holt, 1977]. In the two simple models, A preys on either B or C. In the composed model, A preys on both B and C

and state changes as an algebraic expression. The meaning of this expression is thus delivered in terms of the meaning of its action components and the composition operators, giving rise to composability.

Composability of model expressions becomes an instrumental aid in modeling complex interactions and dynamics such as those that can be observed among individuals of ecosystems. From an analytic bottom-up point of view, composability of BlenX language allows the modelers to consider parts of a complex ecological system in isolation, and build models by composing these parts at the same level. This makes it easy to build larger models when required.

As for illustration for bottom-up composability, let us consider two simple predator-prey models in isolation as depicted in Figure 2. In the first model, species A exclusively preys only on species B. In the second model, species A preys on species C. In BlenX, we can consider these two models in isolation, and simply merge the two models in order to obtain a composed model as depicted in Figure 2 and Figure 3. Then in the composed model, the only additional requirement is the inclusion of the compatibility parameters, which contain the information on which boxes interact with each other.

From a dual top-down point of view, composability makes it possible to consider a component of a model as a black-box, or "open the black-box" to modify the model to include further aspects. This kind of composability becomes instrumental, for example, when a model is refined with a previously ignored detail of the modeled system. Here what we mean by refinement concerns the structure

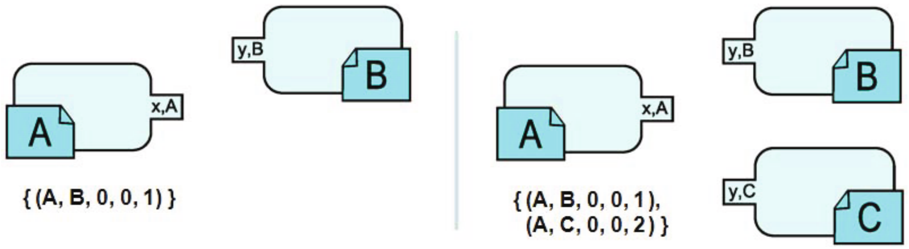


Fig. 3. Graphical representation of the composition in BlenX of the models depicted in Figure 2

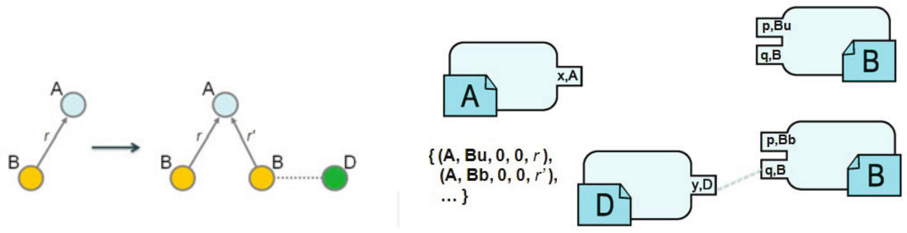


Fig. 4. Refinement of a simple model by further aspects of the predator-prey interaction. In the simple model A preys on B with rate r. In the refined model, A preys on B with rate r' if B is accompanied by D

of the model (rather than size), which boils down to choosing the appropriate component of the model and extending it. Within BlenX, algebraic constructs aid to refine the model by making it possible to work locally on the boxes representing the involved species and including the new data on the system. For example, consider a model, as depicted in Figure 4, where we have a species A that preys on species B. However, in this case, we refine the model with the information that in the presence of species D, the feeding rate of A on B changes from r to r' [Wootton, 1993].

By providing the means for these two dual perspectives, composability becomes instrumental in extending and refining the models to capture the optimal level of representation with respect to the goals of the model [Grimm et al., 2006]. Refined models reflect more aspects of the reality in the structure of the model with respect to the modeled ecosystem. However, sensitivity analysis and other static and dynamic analysis of the model and the simulations with it become more involved as the level of refinement increases. Moreover, the choice on which aspects of "reality" to include in the model in terms of qualitative and quantitative data is an important decision. In this respect, composability becomes an instrumental modeling aid that provides the means to move between different levels of abstraction and extend, and shrink the model with respect to requirements.

4 Composability and Language Design

The algebraic operators and the language constructs of specialized languages such as BlenX allow these languages to capture the mechanistic structure of the systems that they model. This way, for instance, BlenX can capture with its syntax the ecological phenomena that are otherwise challenging to model in other languages. This is an advantage in contrast to other programming languages that can provide a complex encoding of the desired behavior of these phenomena. However, the mathematical syntax of these languages makes them difficult to use for the people who lack training in these languages. This results in a barrier for these languages to be used effectively by a broader audience.

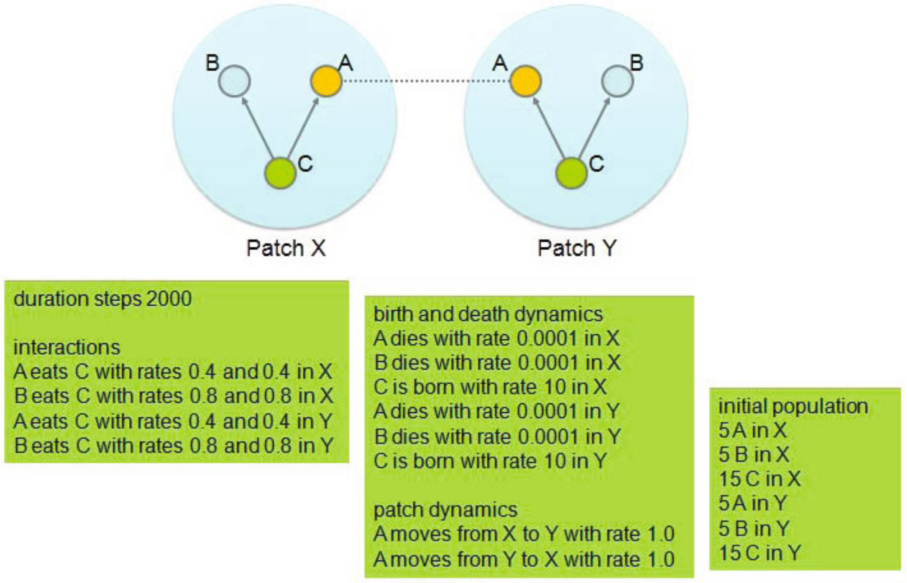
As a remedy for this barrier, one of the central objectives of algorithmic systems biology is the development of user-friendly interface languages for modeling. These interface languages profit from the expressive power of specialized languages, such as BlenX, while remaining accessible to domain specialists who are not familiar with formal languages. In this respect, when stochastic process algebra languages such as BlenX are considered as target languages for user-friendly interface languages, composability becomes a valuable feature that brings an ease to the design and development process.

The CoSBiLab LIME [Kahramanoğulları et al., 2011] is an example to such interface languages for ecosystem modeling. LIME is a language interface to BlenX for building ecosystem models. LIME allows the user to give a biologically intuitive model description in a narrative style controlled natural language. After performing static analysis on the model structure, the LIME translation software tool translates the model description into a BlenX program. This makes the BlenX and CoSBiLab modeling framework handy and intuitive for ecologists. Simulations can be run by the BetaWB and the output can be analyzed and visualized by both Plotter [Dematté et al., 2008] and CoSBiLab Graph [Valentini and Jordán, 2010].

An example LIME model is depicted in Figure 5. A LIME input file can consist of five parts that describe different aspects of the model.

1. The first part is a single statement on the simulation duration.
2. The second part consists of sentences that describe the interactions of the individuals of the modelled ecosystem. Each sentence describes an interaction in the ecosystem together with the ecological patch where it happens and its rate. The interactions can be of four different kinds: *predator-prey*, *plant-pollinator*, *direct competition*, and *facilitation*.
3. The optional third part of the input file collects the information on the birth and death rates of the species. Each sentence in this part describes the birth and/or death rates for each species in each habitat patch. If a habitat patch is not specified, the rate is distributed and applies to all the patches: this way, general rates can be defined.
4. The optional fourth part of the input file contains the information on patch dynamics of the ecosystem: each sentence here describes the migration rate between two particular patches of a given species.
5. The fifth part provides the information on the initial population sizes (at the beginning of the simulation).

For an illustrative example, consider the LIME model in Figure 5 that consists of two habitat patches, X and Y, with identical parameters. The local communities consist of two predators, A and B, sharing a single prey, C. The consumption rate of A and B on C are 0.4 and 0.8, respectively. However, A migrates between the patches, with rate 1. A and B have a death rate of 0.0001, and C has a birth rate of 10. Finally, the initial population size is 5 for the predators and 15 for the prey. The LIME description of this model consists of the frames given in Figure 5, which is much easier to write and work with in comparison to the BlenX code that it generates. In order to see this explicitly, let us consider the



duration steps 2000

interactions
A eats C with rates 0.4 and 0.4 in X
B eats C with rates 0.8 and 0.8 in X
A eats C with rates 0.4 and 0.4 in Y
B eats C with rates 0.8 and 0.8 in Y

birth and death dynamics
A dies with rate 0.0001 in X
B dies with rate 0.0001 in X
C is born with rate 10 in X
A dies with rate 0.0001 in Y
B dies with rate 0.0001 in Y
C is born with rate 10 in Y

patch dynamics
A moves from X to Y with rate 1.0
A moves from Y to X with rate 1.0

initial population
5A in X
5 B in X
15 C in X
5A in Y
5 B in Y
15 C in Y

Fig. 5. A graphical representation of an ecosystem model and its CoSBiLab LIME representation. The species A and B are two predators which both prey on C. The species A can move between the patches X and Y

following sentence:

A eats C with rates 0.4 and 0.4 in X

This sentence is translated into BlenX code, a part of which is

```
... if (ex,Aex_X) then ex!().start!() endif
  + if (ey,Aey_X) then ey!().ch(r,ArRep_X) endif ...
```

and the sentence

A dies with rate 0.0001 in X

is translated to the following code:

```
... + if (ex,Aex_X) then die(0.00010) endif ...
```

Migration between patches in the narrative form

A moves from X to Y with rate 1.0

is translated to the following BlenX expressions:

```
... + if (ex,Aex_X) then delay(1.0).ch(r,Ar_Y).
                   ch(ex,Aex_Y).ch(ey,Aey_Y).start!() endif ...
```

In comparison to BlenX, LIME has a limited expressive power, since it can be used to model only certain kinds of ecological interactions and dynamics. However, LIME enjoys a higher level of composability in comparison to BlenX. This is because LIME models can be written, extended and modified with a great ease with almost no prior knowledge of this language. For example, the food web models that we borrow from [Pimm, 1980] in Figure 6 can be written within few minutes. This capability makes it very easy to experiment with different models.

Modeling different types of concurrent interspecific interactions is a big challenge in ecology [Olff et al., 2009]. There are many works in ecology literature on single interaction types, with an emphasis on food webs, and an increasing focus on plant-pollinator networks in isolation. However, in ecosystems, different kinds of interactions always happen in parallel, thus it is paramount to model them simultaneously. In LIME, composability of the process algebra constructs play a key role in expressing these different kinds of interactions in a unified manner in a single model. Moreover, composability brings a great ease into the design and maintenance of such interface languages.

5 Discussion

For both technical and historic reasons, individual based models (IBMs) still face major computational challenges [Gronewold and Sonnenschein, 1998]. Some of these challenges are linking elegantly mechanisms and behavior [Seth, 2007], minimizing the combinatorial explosion in state space of complex models, and modeling common currency-based integration of several, multi-level and multi-scale processes [Allen and Starr, 1982,Levin et al., 1997]. In particular, a desired advancement in IBMs is the development of generic mechanisms for integrating multiple, parallel ecological processes [Levin et al., 1997,Olff et al., 2009]. Similar to processes addressed by the language LIME, these processes can act at the same level, for example, as in predation and facilitation interactions among the species of an ecosystem [Bertness and Callaway, 1994], or they can act at different organizational levels, for example, as in dispersal in the metacommunity and competition in the local community. Progress in these problems could help in better understanding the relationship between patterns and processes in ecology [Pimm, 1991].

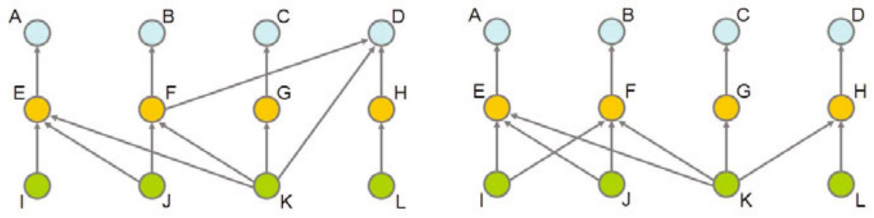


Fig. 6. Graphical representation of two food webs that are variations of each other

The view of ecological systems as complex reactive systems, similar to now broadly established consideration of molecular biology systems, provides the means to model, simulate and analyze these systems, and also indicates directions which can contribute to the solution of above mentioned challenges that confront IBMs. In fact, the consideration of ecosystems as complex reactive systems has parallels also with the complex adaptive system view of of ecosystem models, which are summarized by three properties [Levin, 1998]: (*i.*) diversity and individuality, (*ii.*) qualitative and quantitative aspects of the localized interactions, and (iii:) autonomous processes that reflect the effects of the interactions to their replication or removal, and to the enhancement of their interactions. As we have demonstrated above, composability properties and language constructs of the language BlenX are promising tools for addressing these properties.

The computer simulations in ecology promise the further added value of filling the void due to the impossibility of experiments within the study of certain ecosystems. This is simply because it is not plausible, even if possible, to 'knock out' all the members of a species in a certain ecosystem in order to see the effect. However, certain experiments can be easily designed in silico by resorting to languages such as BlenX and LIME.

An important aspect of the BlenX language with respect to ecological modeling is its stochastic semantics. Stochasticity, which manifests itself as fluctuations in simulations, is an instrumental feature for studying the inherent noise in ecosystems, both at individual level and at the population level. Certain sources of noise are much more important in small populations [Powell and Boland, 2009]. For example, as opposed to deterministic population-level models, stochastic IBMs make it possible to model actual extinction events. While the extinction of rare species is at the frontier of applied ecological research and conservation biology, more generalistic discussions of ecosystem models are limited in both handling the noisy behavior of small populations and modeling extinction. For instance, the same amount of living biomass can behave quite stochastically if it corresponds to two whale individuals, and quite deterministically if it corresponds to millions of organisms in the zooplankton. Availability of stochasticity together with composability is an additional asset from the point of view of IBMs.

References

Allen and Starr, 1982. Allen, T.F.H., Starr, T.B.: Hierarchy: Perspectives for Ecological Complexity. The University of Chicago Press (1982)

Bertness and Callaway, 1994. Bertness, M.D., Callaway, R.: Positive interaction in communities. Trends in Ecology and Evolution 9, 191–193 (1994)

Billick and Case, 1994. Billick, I., Case, T.J.: Higher order interactions in ecological communities: what are they and how can they be detected? Ecology 75, 1529–1543 (1994)

DeAngelis and Gross, 1992. DeAngelis, D.L., Gross, L.J.: Individual-based Models and Approaches in Ecology. Chapman and Hall, New York (1992)

Degano et al., 2005. Degano, P., Prandi, D., Priami, C., Quaglia, P.: Beta binders for biological quantitative experiments. ENTCS 164(3), 101–117 (2005)

Dematté et al., 2010. Dematté, L., Larcher, R., Palmisano, A., Priami, C., Romanel, A.: Programming biology in BlenX. Systems Biology for Signaling Networks 1, 777–821 (2010)

Dematté et al., 2008. Dematté, L., Priami, C., Romanel, A.: The beta workbench: a computational tool to study the dynamics of biological systems. Briefings in Bioinformatics 9, 437–449 (2008)

Dematté et al., 2008. Dematté, L., Priami, C., Romanel, A.: The Blenx Language: A Tutorial. In: Bernardo, M., Degano, P., Tennenholtz, M. (eds.) SFM 2008. LNCS, vol. 5016, pp. 313–365. Springer, Heidelberg (2008)

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

Grimm, 1999. Grimm, V.: Ten years of individual-based modelling in ecology: what have we learned and what could we learn in the future? Ecological Modelling 115, 129–148 (1999)

Grimm et al., 2006. Grimm, V., Berger, U., Bastiansen, F., Eliassen, S., Ginot, V., Giske, J., Goss-Custard, J., Grand, T., Heinz, S.K., Huse, G., Huth, A., Jepsen, J.U., Jørgensen, C., Mooij, W.M., Müller, B., Peer, G., Piou, C., Railsback, S.F., Robbins, A.M., Robbins, M.M., Rossmanith, E., Rüger, N., Strand, E., Souissim, S., Stillman, R.A., Vabø, R., Visser, U., DeAngelis, D.L.: A standard protocol for describing individual-based and agent-based models. Ecological Modelling 198(1-2), 115–126 (2006)

Gronewold and Sonnenschein, 1998. Gronewold, A., Sonnenschein, M.: Event-based modelling of ecological systems with asynchronous cellular automata. Ecological Modelling 108(1-3), 37–52 (1998)

Holt, 1977. Holt, R.D.: Predation, apparent competition, and the structure of prey communities. Theoretical Population Biology 12(2), 197–229 (1977)

Kahramanoğulları et al., 2011. Kahramanoğulları, O., Jordán, F., Lynch, J.: CoSBi-Lab LIME: A language interface for stochastic dynamical modelling in ecology. Environmental Modelling and Software 26, 685–687 (2011)

Kitano, 2002. Kitano, H.: Systems biology: A brief overview. Science 295, 1662–1664 (2002)

Levin, 1998. Levin, S.A.: Ecosystems and the biosphere as complex adaptive systems. Ecosystems 1(5), 431–436 (1998)

Levin et al., 1997. Levin, S.A., Grenfell, B., Hastings, A., Perelson, A.S.: Mathematical and computational challenges in population biology and ecosystems science. Science 275(5298), 334–343 (1997)

Lotka, 1927. Lotka, A.J.: Fluctuations in the abundance of a species considered mathematically. Nature 119, 12 (1927)

Olff et al., 2009. Olff, H., Alonso, D., Berg, M.P., Eriksson, B.K., Loreau, M., Piersma, T., Rooney, N.: Parallel ecological networks in ecosystems. Philosophical Transactions of Royal Society B 364(1524), 1755–1779 (2009)

Pimm, 1980. Pimm, S.L.: Food web design and the effects of species deletion. Oikos 35, 139–149 (1980)

Pimm, 1991. Pimm, S.L.: The Balance of Nature? Ecological Issues in the Conservation of Species and Communities. The University of Chicago Press (1991)

Platt et al., 1981. Platt, T., Mann, K.H., Ulanowicz, R.E.: Mathematical Models in Biological Oceanography. The UNESCO Press (1981)

Powell and Boland, 2009. Powell, C.R., Boland, R.P.: The effects of stochastic population dynamics on food web structure. Journal of Theoretical Biology 257(1), 170–180 (2009)

Priami, 1995. Priami, C.: Stochastic π-calculus. The Computer Journal 38(6), 578–589 (1995)

Priami, 2009. Priami, C.: Algorithmic systems biology. Communications of the ACM 52(5), 80–89 (2009)

Priami and Quaglia, 2004. Priami, C., Quaglia, P.: Modelling the dynamics of biosystems. Briefings in Bioinformatics 5(3), 259–269 (2004)

Regev and Shapiro, 2002. Regev, A., Shapiro, E.: Cellular abstractions: Cells as computation. Nature 419, 343 (2002)

Seth, 2007. Seth, A.K.: The ecology of action selection: insights from artificial life. Philosophical Transactions of Royal Society B 362(1485), 1545–1558 (2007)

Ulanowicz, 1986. Ulanowicz, R.E.: Growth and Development: Ecosystems Phenomenology. Springer, Heidelberg (1986)

Valentini and Jordán, 2010. Valentini, R., Jordán, F.: CoSBiLab Graph: the network analysis module of CoSBiLab. Environmental Modelling and Software 25, 886–888 (2010)

Vasas and Jordán, 2006. Vasas, V., Jordán, F.: Topological keystone species in ecological interaction networks: considering link quality and non-trophic effects. Ecological Modelling 196(3-4), 365–378 (2006)

Volterra, 1926. Volterra, V.: Fluctuations in the abundance of species considered mathematically. Nature 118, 558–560 (1926)

Wootton, 1993. Wootton, J.T.: Indirect effects and habitat use in an intertidal community: interaction chains and interaction modifications. The American Naturalist 141(1), 71–89 (1993)