

Chapter 12

Individual-Based Models

Hauke Reuter, Broder Breckling, and Fred Jopp

Abstract This chapter will describe the category of models that represent the behaviour and interaction of distinct individuals with specific properties. Models of this type can become very complex, but have the advantage that model structures operate on a low level of abstraction and represent ecological relations in a form similar to empirical assessment. Individual-based models facilitate studies of emergent properties, where characteristics of higher level entities like populations or communities can be generated on the basis of single actions of particular individuals. They allow to simultaneously investigate energetic and physiological aspects, behaviour, and relations to other organisms and heterogeneous environmental structures. As a technical background, object-oriented programming is frequently used for this model approach. This chapter introduces the conceptual background and describes two case studies, one that investigates spatial aspects of a predator–prey interaction, and a second one which depicts community interactions of Northern Scandinavian small mammals with oscillating population dynamics.

12.1 Introduction

Individual-Based Models (IBM) represent single organisms and their environment. They allow studying the implications of physiological processes, behavioural traits, and environmental interactions synchronously. This offers a structurally unique option for ecological modelling, because of the potential to join structural, functional, quantitative and qualitative aspects in a way that closely conforms to observation data and conceptual knowledge representation. In the context of ecological modelling applications, we use the term agent-based models synonymously.

H. Reuter (✉)
Leibniz Center for Tropical Marine Ecology GmbH (ZMT), Fahrenheitstraße 6, 28359 Bremen,
Germany
e-mail: hauke.reuter@zmt-bremen.de

Background and Development of the Approach

Early applications of this approach go back to the 1970s. They were a response to the requirements to include more biological realism and explicit spatial representations into ecological models (Łomnicki 1988). First models were introduced by Kaiser (1976; territoriality of dragonflies), Hogeweg and Hesper (1983; social interaction of bumblebees), Seitz (1984; life stage sequences in a *Daphnia* populations), and DeAngelis et al. (1979; development of cohort structure in small-mouth bass populations) and modelling forest stand dynamics (Botkin et al. 1972; Shugart and West 1977).

The minimum requirement for an individual-based model is the separate representation of individual entities, which can be distinguished in one or more characteristics. These characteristics of the individuals must be separately accessible and tracking of individual state changes must be possible during simulation. In most of the application cases, the number of states, and the repertoire to modify the states depends on the internal conditions. In relevant application cases, individual-based models attempt to provide a coherent picture of how particular organisms would act in a particular condition.

To use the full potential of the approach it is also possible to include different levels of entities. Complex modular organisms like trees can be represented as a set of individual branches, roots, leaves, fruits, etc. The individual organism is then a compound instance of sub-units. On the other hand, it is also possible – and sometimes useful – to operate other compound entities. For example, a representation of an environment can comprise a spatially differentiated structure where physical and chemical parameter differ locally and give rise to specific local responses of the organism's activity. Furthermore, abstract entities like populations can be represented, either as units with specific parameters like age distribution, biomass spectrum, which change during simulation, or as an aggregate that integrates over the individuals included in the model. Such an extended specification of an IBM may thus comprise configurations in which the components are not basic units but particular components of ecological systems. In principle, these may range from (sub-)cellular units, plant modules (Breckling 1996; Eschenbach 2005) to aggregations such as cohorts, social animals (nests, hives), populations, functional types, or spatial or temporal units of higher order (Köhler et al. 2003; Middelhoff et al. in print).

With such an extended understanding of how the IBM approach can be used, one can see that it is in fact structurally identical, *sensu strictu*, to agent-based models (ABMs). The term ABM originally emerged in a computational context with applications in physics as well as applications in social sciences and economics. ABM often describe robotic aggregates responding to a variable environment, or they simulate complex behaviour of humans in social networks. From an ecological perspective, it appears reasonable to use both terms (IBM and ABM) synonymously. In a similar way, the term multi-agent system or multi-agent simulations (MAS, e.g. Ferber 1999) bases on the same concept, however emphasizes the interactions of a larger number of autonomously acting software agents and is even more common in technical applications.

The approach becomes operational only if the computational challenges of such a model can be met. This was made possible by developments in computer programming and programme processing. The representation of a larger number of individuals with an interaction potential is feasible only with larger processing and data management capacities. The advancement in hardware and software development allowed more resource-demanding applications like Object-Oriented Programming (OOP). In the late 1960s, the programming language SIMULA (Dahl et al. 1968) provided the ground for the virtual representation of active agents, which was later adapted to various programming languages (Smalltalk, Delphi, C++, Java, and others).

Early IBMs often had a narrow focus and concentrated on single species investigations (e.g. Kaiser 1976; DeAngelis et al. 1979; Seitz 1984). These early models applied quasi-automatic transition between the single model-states (e.g. age, biomass, location). However, they could already illustrate the great potential of the IBM-approach. It thus added a new perspective to modelling in a close relation to the specific characteristics of ecological systems (see Chap. 4 on systems analysis), compared to the homogeneity requirements of variables as they were used in the classical systems dynamic approaches (Forrester 1968).

Further developments made IBMs applicable for investigations of behavioural decisions and interactions in social groups. Paulien Hogeweg and co-workers pioneered this field with their model on social interaction in bumblebee colonies (Hogeweg and Hesper 1983). A first paradigmatic overview was presented by Huston et al. (1988).

Facilitating a representation of variable environments, structured populations and behavioural traits, the modelling of complex life histories emerged (e.g. Wolff 1994; Colasanti and Hunt 1997). For instance, it became possible to simulate highly resolved time-energy-budgets as a basis for behavioural decision processes. The model on the reproduction phase of a robin population is such an example (Reuter and Breckling 1999) and allowed to investigate reproduction success under different environmental settings. A further development in IBM-methodology involved the number of considered and interacting species. In this context the inclusion of interaction rules plays a major role. These rules often refer to trophic relations (e.g. Kaitala et al. 2001), spatial competition or even to succession processes (Breckling 1990).

An increasing number of models combine sophisticated internal resolution of organismic processes with the representation of several species and their interactions to analyze e.g. food webs and community dynamics. Examples for this type are the simulation of plant competition including different herbivores by Parrot and Kok (2002) and the analysis of regular population cycles of small mammals (Reuter 2005, see Sect. 12.4). Often the simulated entities are designed to operate in heterogeneous environments including a spatially explicit habitat structure, seasonality and varying climate data. The models simulate explicitly designed scenarios directly, using empirical data involving e.g. GIS-derived maps and assumptions on local temporal and spatial developments. In the marine context we find successful attempts extending simulation models to include all relevant trophic levels (so-called end-to-end models, e.g. Travers et al. 2007). These approaches in marine

ecology propagate the coupling of different model types in which individual-based models are thought to play an integrating role (Cury et al. 2008) because of their flexible structure, which allows to combine knowledge and data from different sub-disciplines, that can be used to analyze findings on heterogeneous ecosystem levels and to understand the corresponding complex interaction and network structure.

In particular, since radio-tracking became possible, the modelling of the behavioural repertoire within a spatial context, and especially models that explicitly investigate animal movement and dispersal, have become an important domain of IBM applications (Gustafson and Gardner 1996; Broekhuizen et al. 2003; Jopp 2003; Pe'er and Kramer-Schadt 2008). IBMs have greatly contributed to the study of population dynamics in complex landscapes (Lima and Zollner 1996; Nathan et al. 2008; Revilla and Wiegand 2008). Population development may be simulated in dependence of complex behavioural modes or context dependent changes in reactions (Shin and Cury 2001) also including the field of population viability analysis (PVA, e.g. LePage and Cury 1997; Mazaris et al. 2005). IBM of invasion processes allow combining dispersal processes with species properties, individual behaviour and the properties of the invaded community (Higgins et al. 2001; Nehrbass et al. 2007).

Since their beginning, IBMs have undergone a rapid development and have been applied to almost all ecological topics and a large number of research questions (DeAngelis and Gross 1992; DeAngelis and Mooiji 2005; Grimm and Railsback 2005). In the following, we illustrate the basic structure of IBMs and demonstrate their functionality on the basis of simple model applications.

12.2 The Structure of Individual-Based Models

In order to give an overview on basic formal elements of individual-based models we begin with a short introduction into the programming background and an outline of the concept of object-oriented programming (OOP, e.g. Rumbaugh et al. 1991; Silvert 1993; Hill 1996). Then we look at the application of the OOP to conveniently describe structure and interaction of individual actors.

In OOP, the source code is organized in blocks which are delimited from each other. There are different types of blocks with the so-called CLASS as the most prominent one. A CLASS is a specific programme unit which consists of storage reservations for variables and may additionally contain code (statements) how to change the values of its variables. In such a CLASS it is possible to specify further sub-classes which allows to implement a hierarchic programme structure. During programme execution, a CLASS can be copied multiple times and these class instances may be kept available in the computer storage. This is the decisive feature in object-oriented programming. The copies of a class are called OBJECTS, thus being eponymous for the whole approach. Each of these objects, which are available during runtime of the programme, consists of the same code as the class from which it is derived, but may contain specific values stored in its variables. This

allows objects to differ from each other with respect to the role they play in model execution. The variations in variable values may trigger different parts of the internal code to be executed and thus may lead to a different behaviour and development of the respective object.

In fully featured OOP-languages, the command to instantiate an object may be triggered from any part of the programme. To access objects, a special kind of variable is needed. These so-called REFERENCE VARIABLES or POINTERS directly refer to a specific object and thus facilitate uni-lateral or mutual interactions between objects. OOP has revolutionized computer programming due to its more flexible design structure and clear organization of programme code. Moreover, the features of object orientation make even complex models easier to maintain and helps in tracing errors (“bug tracking”). Due to the flexible structure during programme run time (instantiation and deletion of objects, switching of pointers from one object to another) OOP easily allows to handle the structurally complex interaction networks required for advanced ecological applications (Reuter et al. 2008). This allows simulating a large variety of phenomena, in particular self-organizing spatiotemporal structures on different levels of organization (Reuter et al. 2005).

Most individual-based programmes contain relatively similar essential parts and processes which are common for this model type:

1. The representation of an individual entity as a class
2. The layout of a structured representation of the environment
3. The organization of the temporal model execution and interaction between the entities

These parts will be explained in the following.

12.2.1 Representation of Individual Entities

A class can be conveniently utilized to describe the life-history and interaction of an individual organism. Usually, it consists of three main parts (Fig. 12.1): (a) state variables which describe individual properties and attributes, (b) statements and code blocks which are used to update these variables, and (c) a scheduling mechanism to update the properties of the individuals.

a) Variables Describing Individual Properties

Which kind and how many variables are necessary to describe the properties of an individual, depends on the research question and the complexity of the individual life-history and activity repertoire in focus. In the simplest case, one property/variable is enough to be able to distinguish the individuals. For example, describing movement or dispersal patterns would necessarily require variables to store the

Fig. 12.1 Basic scheme of a class to represent an individual (Breckling et al. 2006)

```

Class Organism
  Declarations:
    Variables characterising individual state
    (e.g. location, biomass, age, alive)

    Activity Procedures („methods“) to update states
    (e.g. Move, Growth, Reproduction)

  Life Loop:
    While alive do ...
      Apply (call) Activity Procedures
      to update variables
      Hold [Delta t ] (i.e. detach this organism
      temporarily to update others)
    End life loop
  End organism

```

coordinates related to an individual's location (Jopp and Reuter 2005). In most cases, the number of variables describing individual properties is considerably higher. Besides location, they often comprise biomass, age, sex and a relational context. For instance, when the object refers to plant modules, information on neighbouring modules is decisive to determine transport processes. For many higher animals e.g. the information on a home range (or territory) or on eventual offspring that have to be fed, may be necessary.

b) Code to Update Individual Properties

It is useful to organize the updating of an individual's state in terms of a set of "activity procedures" (changing states relevant with regard to the environment) and "physiological processes" (changing states referring to the internal condition of the individual). Common examples for activity procedures are movement, reproduction, and feeding. Physiological processes represented in a model can be ageing and energy metabolism. The according procedures access and update the involved variables. For example, an activity "movement" should change the variables storing the location information. If energetic processes are included, "movement" may also change the energetic state to include the cost of the particular activity. The description of activities can be accomplished with very simple rules and can also integrate other mathematical approaches like fuzzy logic (see Chap. 10) or differential equations (see Chap. 6). The decision which details should be included in an activity procedure depends on the focus of the model, the available knowledge and information status of the ecologist. As the ecological quality of the input information basically decides the character of the model as a whole, we advocate that ecologists, with the necessary knowledge at hand, should be intensively involved in the programming process or better yet, learn to program their own models instead of relying on specialized programmers or pre-defined software tools which usually restrict the optimal adaptation to what the specific situation requires.

c) **Process Scheduling: How to Organize the Regular Update of Individual Processes and Properties**

The above part focussed largely on structural descriptions. Essential for the life-history model of an organism is how exactly the update process of all individual state variables is organized. This constitutes the dynamic part and can be referred to as “process scheduling”. To coordinate the concurrent execution of a larger number of entities in a model requires a loop control structure within the program code of the particular class (respectively the instantiated objects), which is iterated as long as the object has the internal status as a living entity during the simulation run. To distinguish active and no longer active objects requires the introduction of a Boolean variable, e.g. with the name “ALIVE” as one of the object’s states. A Boolean variable can store the values of either TRUE or FALSE. The control mechanism that uses the distinction, is referred to as “life loop”.

The “life loop” of each currently active individual entity is repeated (iterated) as long as this variable has the Boolean status TRUE. Otherwise, the object will be terminated, deleted from the storage and the storage space released as being freely available. From a top-down point of view, the execution of any specific activity can be made dependent on distinctive conditions that relate to the internal state of the individual entity (like the energetic state, age, reproductive stage) and with the external situation (e.g. availability of food, presence of predators, daytime or time of the year). It is thus necessary to implement an algorithm to determine which of the possible activities is to be executed for a given situation.

These scheduling algorithms may range from very simply to very complex. A simple scheme would e.g. execute all activity and physiology procedures in the same order during each passing of the life loop as a “cyclic activity control”. This activity scheduling mechanism is adequate for configurations where it is not necessary to evaluate complex behavioural alternatives that may require changes of the sequence of activities (e.g. models concentrating on movement behaviour, see Sect. 12.3, the IPP example).

To model complex behavioural patterns, a further elaborated decision algorithm is required. For instance, with a “priority driven activity control” it is possible to assign a variable corresponding to each activity which indicates its execution priority. Consequently, during each life loop sequence the activity with the highest priority value gets executed. In the course of execution, the priority of the particular activity is reduced, while the idle activity alternatives may accumulate successively higher priority values. Thus complex behavioural decisions (including time-energy budgets, analysis of behavioural trade-off’s in life history) can be represented by considering external and internal states in relation to the supposed execution time for each activity (Breckling et al. 1997).

In an individual-based model, the ecologically relevant state of any simulated entity results from all performed activities in which the relevant inner states of the individual have been evaluated in feedback processes with all relevant external states (e.g. “environment”). The activities of an individual thus can alter its own states (e.g. if hungry through food search) but can also influence the environment,

and the state of other individuals (e.g. if predated then a predator will influence the “ALIVE” variable of a caught prey object).

Because the interaction structure becomes successively more complex with an increasing number of concurrent objects, it is usually not reasonable or possible to specify it directly. Instead, a self-scheduling mechanism is required. How to set-up such a mechanism is described in Sect. 12.2.3.

12.2.2 Representation of the Environment

One of the important potentials of IBMs is to facilitate an easy way for simulating spatial (and temporal) variability and heterogeneity. In the following, we focus on how to simulate heterogeneous environmental states. When starting on the organismic level, a heterogeneous spatial organization will, to some extent, already be reached by specifying location variables for the represented organisms and installing an activity procedure “movement” to change these variables adequately (see Sect. 12.3., the IPP example). When organisms interact (e.g. as predators or prey, or as schooling organisms), the presence/absence of other individuals structures an otherwise homogeneous environment. In addition, various other data structures can be used to represent spatial heterogeneities. Frequently, grid-based representations are used. Grid maps with the relevant information attached to each cell of the grid can store e.g. (water) depth and currents in aquatic surroundings, or altitude and habitat types in terrestrial environments. It is possible to include spatially heterogeneous resource levels, physical structures, local light intensity, eventually in relation to slope orientation. Often, this information is read into the programme from external sources at the beginning of a simulation run. Also, it can be generated and modified in the programme itself with particular updating routines. In more complex computer models, the environmental information is frequently generated by external modules or by other programmes. In these cases, the simulation requires the coordinated employment of an overall system of coupled models. For simulations in marine environments, an IBM can be coupled with a regional oceanography model (ROM), which provides regularly updated information on currents and physical water conditions (Penven et al. 2006). In a similar manner, it is possible to read-in weather data in order to specify seasonal changes. Sometimes, Cellular Automata (see Chap. 8) are employed to generate environmental structures (Breckling et al. 2006). The resource density and the way it is influenced by the organisms is a frequent topic considered in IBM (Reuter 2005; Charnell 2008).

12.2.3 Scheduling Programme Processes

After discussing the update of individuals, and aspects of the data management to provide dynamically changing environmental structures we now consider the

coordination of the different entities. With many concurrently active units, this is an important task. A coherent solution is required – and crucial for model execution.

Individual-based programmes usually employ a *discrete event scheduling* mechanism: Any calculation result is accessible at a specific point in simulation time. It is not reasonable to attempt a simulation of a continuous approximation, in particular, if many qualitative decisions are to be taken (an organism is alive or not, etc.). In OOP simulations, very large numbers of objects can be active at the same simulation time. In principle, real parallel processing is physically not possible if the number of processors is smaller than the number of objects. Therefore, an explicit handling of execution order is necessary. This task is a general one and does not need to be newly developed for each model. It can be generally solved on the level of the simulation environment and is employed for the specific scheduling requirements.

Often, programme environments have a central instance which activates the updating of all processes. This is acceptable if a fixed scheduling scheme can be described in advance and updating is quite regular (e.g. as in the case of cellular automata). However, we advocate a more flexible solution in which updating requests are controlled by the objects themselves and allow changes depending on the objects internal state. The objects already contain the code to calculate which activities are performed under the specific conditions. As one additional task it is possible to let them calculate a time interval of how long it would take in simulation time to have the particular activities done.

This time interval is returned to a central time management and can be used to establish a so-called “event queue”, which shifts programme control always to the first one in the queue and eliminates it after execution. Thus, the programme level of event control gets the function of organizing the updating process that originates from the requirements of the objects. SIMULA (Dahl et al. 1968), the first object-oriented programming language, provides a very efficient solution in form of a system class SIMULATION. The event organization is handled automatically in the background, while each object sends a message how long it is put on “HOLD” (being busy with the current activity). This event scheduling concept was revolutionary, and was the blueprint for many other programming languages, which at this time had a different concept of event handling, as the programmer explicitly had to take care of it. The different frameworks for IBM development provide specific inherent solutions for scheduling of active entities.

12.3 Application Example I: An Individual-Based Predator–Prey Model

The structure of an individual-based model will now be illustrated in detail with the Individual-based Predator–Prey Model (IPP). This model was developed

as a spatially explicit, individual-based implementation of a Lotka-Volterra predator-prey interaction with one prey and one predator species (Breckling et al. 2000). The Lotka-Volterra model is a frequently considered topic in differential equation based population modelling (see Chaps. 6 and 7). It is used here to demonstrate the change of perspective, when an individual-based point of view is adopted. Both simulated species have a limited activity repertoire consisting of feeding, growing, reproduction and movement. All activities are kept at a low level of complexity, thus serving as prototypes, however, illustrating the potential for further development.

The total numbers of both prey and predators are limited to a few thousand objects, which corresponds to the processing capacity of the environment to facilitate reasonably fast computation. The potential “movement” algorithms are the same for both species. They consist either of a Brownian (random) movement (for each movement step the new direction is chosen stochastically) or a directed walk (for the next step, the old direction and speed is maintained and a small random component is added). This leads to a higher autocorrelation of the overall movement direction. The choice of the movement algorithm (random vs. directed) has to be made for all organisms of one class by setting a specific value for a switch in a parameter file (“InFile”). It then applies to the whole simulation run. The length of each movement step is calculated in relation to the biomass of the respective organism.

Feeding and energy physiology differ between the simulated prey and predators. The prey grow independently from any external influence. This simulates unlimited resources. However, the predators search a specified radius around their current location during each time step and feed on the prey which they find within this radius. Then, the biomass of the identified accessible prey (multiplied with a conversion factor between 0 and 1) is added to the current biomass of the predator individual. The predator loses biomass according to a biomass dependent respiration function which leads to starvation below a certain threshold, if no prey individuals are met.

Reproduction for both, prey and predators, is biomass dependent and implemented as a fission-like process. If an individual reaches a specified biomass threshold, new objects of the same species are instantiated at the same location, and adult biomass is then distributed between juveniles and the adult.

The environment for the simulation run is established as a homogeneous area, with each edge being connected to the opposite site and thus leading to torus boundary conditions (see Fig. 8.3 in Chap. 8). These conditions minimize boundary side effects (Jopp 2003).

In the beginning of each simulation run, a specified number of prey and predator individuals are distributed randomly across the simulation area. Generally, all relevant parameters describing the behaviour of prey and predator as well as other parameters of the specific simulation (e.g. duration, size of area, etc.) are set in a parameter file.

Though individual actions do not include any directional preference, and only random components change the movement paths, we obtain emerging large-scale

spatial structures as a result of the interactions of the model components. The type of pattern depends largely on the combination of the involved movement processes. In the following, we will elucidate some of the characteristic emerging patterns.

In the first example prey and predator individuals both move according to a random walk. As a consequence, after the initial random distribution prey and predators exhibit a spatial segregation. The population of the predators can only grow in the proximity of the area which is dominated by the prey. As the predator aggregations successively shift towards the prey areas, spatial dynamics result in a kind of travelling wave pattern (see also Sect. 8.4) that involves both prey and predator individuals (Fig. 12.2).

The second example (Fig. 12.3) illustrates the results when the prey exhibits a Brownian (random) movement, whereas the predators move according to a correlated random walk. This scenario leads to a remarkable aggregation of the prey. We find temporarily stable prey clusters with roaming predators which rarely meet a cluster while roaming the overall area. The predators can feed during a few time steps when passing a prey cluster, however will leave it again because they maintain the momentum of their movement.

When all other factors remain unchanged (i.e. *ceteris paribus* condition), the degree of autocorrelation, which is represented in the value for directedness of the predator movement, is the key factor that enables transferring one class of spatial pattern (travelling wave phenomenon) into another (random distribution, see Turchin 1998). Also, further variations of different movement factors that can be specified over the parameter file between prey and predators lead to different spatial distribution patterns.

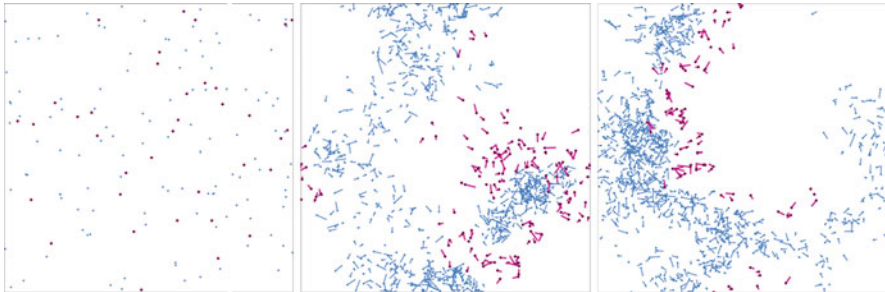


Fig. 12.2 Simulation results of the IPP model simulating a simple individual-based predator–prey interaction. *Points* indicate the current positions of prey and predators, the *line* shows the movement from the previous position. *Lighter shades* and *smaller points* indicate the prey, *darker* and *larger points* indicate the predator. *Left*: The initial distribution is random. *Center*: After 200 time steps – if both types of organisms move randomly, according to a Brownian movement, a characteristic spatial self-organization occurs. After the initial phase a dynamic change of border structures occurs where predator and prey interaction is highest. *Right*: After 400 time steps

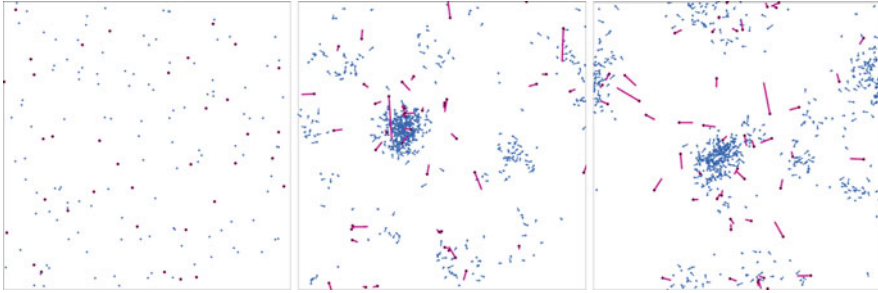


Fig. 12.3 Simulation results of the IPP model simulating a simple individual-based predator–prey interaction. For a description of symbols see Fig. 12.2. *Left*: The initial distribution is random. *Center*: After 100 time steps – if the prey moves randomly, and the predators exhibit a directed movement with a partial auto-correlation, a pronounced cluster structure of the prey occurs. *Right*: after 400 time steps

12.4 Application Example II: Cyclic Rodent Communities in Northern Scandinavia

In order to illustrate the potential of the individual based-modelling approach we outline a more complex example relating to population cycles in Northern Scandinavian rodents. In community ecology, cyclic population dynamics constitute an interesting example for complex dynamics often involving a multitude of species-intrinsic factors and environmental influences (Myers 1988; Bascompte et al. 1997; Sanderson et al. 1999; Haydon et al. 2002; Bauer et al. 2002).

Rodents, mostly in the Northern Hemisphere, often exhibit drastic changes in population size with numbers at peak times reaching up to 500-fold of numbers in the minimum phase. In Northern Scandinavia, where the changes in abundances are most regular with peaks every 3–5 years, cyclic dynamics impact the whole local biocenosis and are synchronous over large areas (Huitu et al. 2003). These community interactions of small rodents have fascinated ecologists for many decades (e.g. Elton 1927; Chitty 1960; Stenseth 1999; Korpimaeki et al. 2005) and gave rise to many controversial discussions (Rosenzweig and Abramsky 1980) on the driving factors. Numerous hypotheses have been put forward including abiotic influences (Aars and Ims 2002; Sundell et al. 2004) and biotic intrinsic factors (Chitty 1967; Boonstra 1994; Oli and Dobson 2001). In the last years, biotic extrinsic interactions (mostly trophic relationships) have been widely recognized as the most important processes. However, it is still discussed, whether rodent population dynamics are controlled by bottom-up causation (e.g. Jedrzejewski and Jedrzejewska 1996; Selas 1997) or are top-down limited (e.g. Norrdahl 1995; Klemola et al. 2003). Further more it is not clear how important the role of pathogens is (e.g. Hoernfeldt 1978; Cavanagh et al. 2004) and if driving factors change with cycle phase. Despite the long lasting controversy and the numerous field investigations, the causalities for population cycles are not yet

entirely known. Especially restricting in this context are the inherent limitations of field work in relation to the temporal and spatial extent of the investigated phenomenon.

In order to analyze large-scale effects that result from complex interactions in variable cause-effect networks, an individual-based model was developed (Reuter 2005). The model allowed integrating the most essential components and their interaction structure on different integration levels. It represents small mammals' communities as a food web which is composed of three trophic levels: (1) rodent food, (2) rodents and (3) predators (Fig. 12.4). Rodents and predators were described as individual organisms with a detailed life history including an activity repertoire and physiological processes. The modelled organisms interact in an environment with a spatial arrangement of habitats under seasonally changing conditions. This concept extended previous differential equation based modelling approaches (Hanski and Korpimäki 1995; Turchin and Hanski 2001) by integrating most aspects from the ongoing debate that are relevant for rodent population dynamics.

Simulations with the model allow covering different scenarios with respect to the environment and the parameterized species. The investigated scenario included the parameterization for two rodent species, field vole (*Microtus agrestis*) and bank vole (*Clethrionomys glareolus*), and two predator species, the least weasel and the

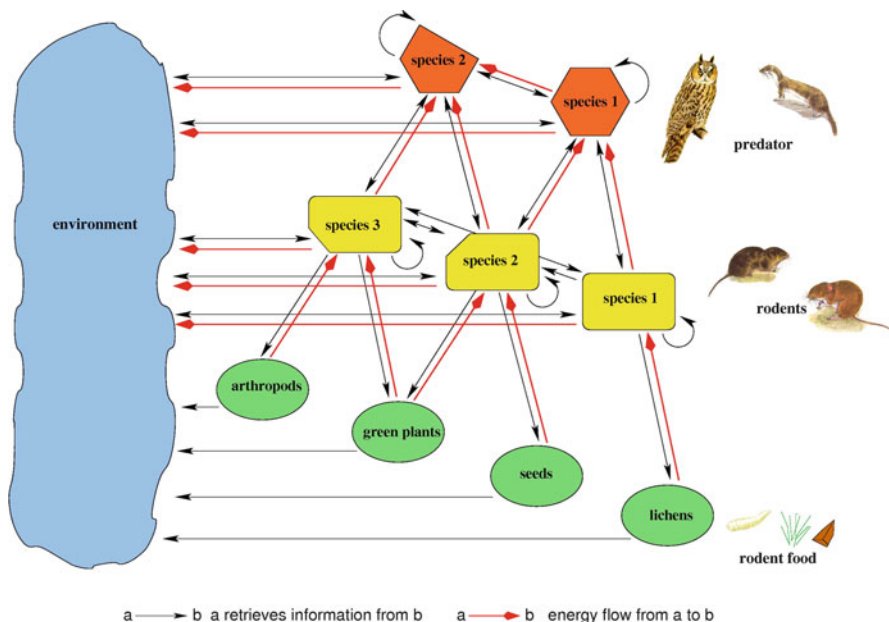


Fig. 12.4 The components and actors of the rodent cycle model: Predators (least weasel, *Mustela nivalis* and long-eared owl, *Asio otus*) and rodents (field vole, *Microtus agrestis* and bank vole, *Clethrionomys glareolus*) are represented as individual objects. The environment is represented as a grid-map which also contains the food resources for the rodents (adapted from Reuter 2005)

long-eared owl (*Mustela nivalis*, *Asio otus*), with respect to their different ecologically relevant properties (e.g. territoriality, food specialization, migration behaviour). Simulations take place on a grid map, with an extent of 150 ha and a spatial resolution of 30×30 m. The resources for the rodents are calculated for each grid cell and exhibit seasonal dynamics and allow for a feedback process with exploitation by rodents.

The model represented dynamics and interactions on different integration levels including individual life history traits, population development and community interactions. The individual level included e.g. ontology, reproduction sequences, weight development, habitat use and interaction with other rodents and predators. As a result of the individual interactions cyclic population dynamics emerged, as they are typical for Scandinavian rodent communities with an average cycle length slightly below 4 years. These cyclic population level dynamics were not implemented in the programme specification but are emergent properties produced by the model components during execution (Breckling et al. 2005; Reuter et al. 2005, 2008).

The model also allows analyzing the population structure with respect to age structure, reproduction rates and mortalities for different phases of the cycle. Specialized predators shared the cycles frequency with the rodents but the phase lags behind. The factors that cause this sudden decline of rodents are believed to be of crucial importance for the whole system dynamics (Batzli 1996).

By analyzing mortalities in the phase between maximum abundance and the following minimum, the model gave new insights into the driving forces: The overall model results showed that mortalities due to intrinsic factors like senescence did not increase distinctly. In contrast, trophic interactions that are based on the lack of food and predation pressure contributed to about 90% of mortality of rodent individuals in this phase. The results clearly emphasize that food web interactions constitute the essential driving force of the cyclic dynamics. Further investigations however yielded another surprise: The trophic control and its relative strength, which was calculated as the difference between normalized mortality rates in relation to either of the two factors, varies unpredictably in the model (Fig. 12.5) and cannot be correlated to specific properties of the respective cycle.

With respect to the identification of these two factors that have a varying impact, the model results give important new conclusions for the discussion of the driving forces in Scandinavian rodent cycles and may thus help to explain the differing results of numerous empirical investigations.

12.5 Conclusions

The relation of modelling on the one side and empirical information and biological knowledge on the other is different for IBM than it is for other modelling approaches. While the modelling approaches usually employ a specific abstraction

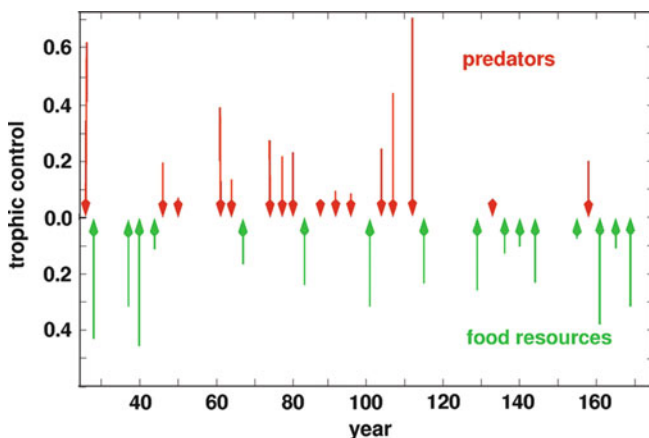


Fig. 12.5 In the rodent population oscillation model it turns out that top-down and bottom-up control of the rodent cycles change in an unpredictable way. This yields an explanation for why empirical investigations led to contradictory results (from Reuter 2005)

pattern that captures ecological relation only according to a particular scheme, IBM has the advantage to represent observation and knowledge in a form that is highly congruent with how we understand existing interaction. The level of abstraction is low, and the model represents elementary interactions that aggregate in the course of model execution in the same form as observable phenomena in empirical investigations. The iterative character of the models allows to “sample” during simulation. In that respect IBM is a crucial tool in testing consistency of ecological knowledge. Due to their potential to represent detailed biological knowledge and small-scale mechanisms, IBMs tend to have a complex model structure. This requires a particular attention to model documentation and evaluation (see Chap. 23).

The generic applicability of the structure of IBMs allows simulating a wide range of issues in terrestrial as well as in aquatic ecology. The illustrated scheme of programme organization is sufficiently flexible to capture organismic development and behaviour, environmental conditions, and the interaction of both. It is suitable to specify e.g. predator–prey interactions (Charnell 2008), schooling (Reuter and Breckling 1994), and behavioural shifts under varying conditions (Peacor et al. 2007), the formation of colonies and the description of structural–functional development of modular organisms (Eschenbach 2005).

IBMs allow to represent interaction of structural and functional features across different scales. Thus situations which do not only involve quantitative transitions but in parallel also qualitative or structural changes can be studied. Simulation results on higher organization levels emerge from the self-organizing interactions of basic units.

Further Readings

- Breckling B (ed.) (2005) Emergent properties in individual-based models – case studies from the Bornhöved Project (Northern Germany). *Ecol Model* 186(4):375–510
- Breckling B, Middelhoff U, Reuter H (2006) Individual-based models as tools for ecological theory and application: understanding the emergence of organisational properties in ecological system. *Ecol Modell* 94:102–113
- DeAngelis DL, Gross L (1992) Individual-based models and approaches in ecology: populations, Communities and ecosystems. Chapman & Hall, New York
- Grimm V, Railsback S (2005) Individual-based modelling and ecology. Princeton University Press, Princeton
- Hill DRC (1996) Object oriented analysis and simulation. Addison Wesley, Harlow