UNIVERSITÁ DEGLI STUDI DI PISA
Dipartimento di Informatica
Dottorato di ricerca in Informatica

Ph. D. Thesis

Methodologies and formalism for modeling macroscopic biological problems

Pasquale Bove

Supervisors
Prof. Roberto Barbuti
Dott. Paolo Milazzo

2016-04-22

Contents

Abstract

This work presents a new computational approach, based on the formalism of P systems, for modelling and running simulations of animal population dynamics phenomena.

Population and ecosystem dynamics models are defined by the description of processes that involve several elements to be taken into account. Populations include individuals that belong to the same species, while one ecosystem includes different species often split in various sub-populations with peculiar genetic features.

We use P systems as a starting point for the development of derived formalisms together with the study of specific population dynamics case studies. This to illustrate the features of the presented formalisms and the power of their modelling capability.

The three formalisms proposed are. MPP systems (Minimal Probabilistic P systems), APP systems (Attributed Probabilistic P systems) and MAPP systems (Multilevel Attributed Probabilistic P systems). All of them are formally defined by providing their syntax notations and formal semantics as inference rules.

The Minimal Probabilistic P systems are derived from flat P systems and enrich the evolution rules with functional rates, enforce the probabilistic maximal parallelism and use rule promoters.
The objects represent various individuals that belong to the modelled population. Special objects, named "control objects", are used as promoters to model the subsets of active rules at various stages of the computation process. Also control objects can be used to model abiotic factors of the ecosystems (e.g. temperature, humidity, pollution level and so on) which can affect the dynamics of the considered population.

The Attributed Probabilistic P systems are an extension of the previous model in which objects definitions are enriched with annotated attributes. In APP systems we still have rules with rating functions and control objects but we also include attributes to allow a better characterization of the models. The attributes of objects consumed by the rules influence the the rating obtained by rating functions associated with each rule. Thus correlation between attributes, functions and rate of rules, generate a flexible probabilistic system.

The APP systems keep the maximal parallelism like flat P systems and MPP systems. The main advantages of this formalism are the readability of the developed models and the unambiguous model description.
Also, the presence of the variables in the rules, greatly improves the compactness of the models.

The Multilevel Attributed Probabilistic P systems are based on a hierarchical system of membranes like the original P systems. Membranes are used to model various levels of aggregation of individuals, both spatial and logical, or social separation/aggregation. The

adoption of a multi-membrane perspective causes the loss of some features introduced in the previous formalisms (i.e. the maximum parallelism), however the use of multiple membranes allows the development of multilevel computational models to study hierarchical systems, a class of problems very interesting to study and model.
The study of dynamical effects that spread along different layers of the hierarchical organization of a population is an important ecological aspect that really deserves to be investigated and modelled.

The three case studies that have been chosen to go with the formalizations of these methodologies highlight the path that entwines their evolution.
For the first case study a probabilistc modeling of events in the stability of sympatric systems between *P. lessonae* e *P. Esculentus* frogs is required, while individuals need a very simple representation.
In the second case study the model highlights the relationship between the lemurs' troop topography and their social structure. In this case is necessary introducing a strong characterization of individuals, using attributes that influence the probabilistic structure of rules.

In the third case study we study a lion's pride social life, with data collected in the Serengeti park as a starting point. Beside the probabilistic structure of rules, the attribues for a detailed definition of individuals, we hereby introduce the possibilty of using among the rules groups of individuals considered as separate entities, and giving them specific attributes. This computational model will generate a a computation that is no longer parallel but cascading.

The most important aspect that we want to emphasize in this thesis is the idea behind all three formalism: we always try to introduce the least possible number of features needed to model the proposed case studies.
It is always possible to define a formalism richer of operators and constructs but, following this direction, we will lose the characteristic of minimality, readability and generality which are ours goals in this thesis.
It is obvious that a formalism ad hoc designed with one specific model already in mind, can exploit the structure of the model. What we try to define here is something fit for general purpose, able to deal with a large class of population problems, so the design process of these formalisms is a continuous trade-off between readability and minimality.
The expressive power of all the three formalism is equivalent to that of a standard P systems, which is proved to be able to emulate a register machine. Anything we will define in this thesis can be modelled in standard P systems form, however the use of the plain P systems to describe our models would have a huge cost in terms of readability and compactness.

Chapter 1

Introduction

1.1 Motivations

Biological problems are one of the most challenging and fruitful application fields for information technology. Population biology and ecology require specific modelling strategies and methodologies like population viability analyses (PVAs) [63] are used in critical conservation decision-making [62].
Biological systems typically involve large numbers of components with complex, highly parallel interactions and intrinsic stochasticity. The design of models about this kind of problems is one of the most effective ways to generate:

- Assumptions on system's evolution starting from initial conditions and a certain parameterization of a model.
- Development of plausible hypotheses and theories to guide the choice of real experiments to be performed, optimizing the resources in consideration of time and cost.
- A better comprehension of the reactions of complex systems to interferences, or the introduction of foreign elements as invasive species, or the appearance of a specific mutation or a sudden mutation in the habitat balance. Thanks to a model previously verified by the experimental data, it is possible to hypothesize and model a whole series of relevant hypothetical scenarios.
- Optimization of resources for intervention. A model can indicate the best action to be taken within a given ecosystem. For example, it can propose to maintain a stable ecosystem or propose changes for a compromised system. Resource optimization can help money and workforce management, and improve the effectiveness of interventions on the territory.

This short list of applications suggests that we refer, more than anything else, to macroscopic problems.
The cell biology and molecular biology are very large fields of application for these models and formalisms, which we will use as a starting point for our direction of research.
However, the models proposed by us are focused on macro-biology, since we work with animal populations, and their internal and external dynamics, at different levels of interaction between different species, on their adaptation to the environment and all that can be associated with the study of individuals in a specific population of the object model.

When attempting to create a model which describes the evolution of individuals composing a population, some approaches have been greatly exploited.
The first approach starts with individuals, chooses a set of attributes that define the main

features, gives to each individual a behavior algorithm and regulates the interactions between two or more individuals defining the proper protocols.
We generate an arbitrary number of these individuals, we choose whether or not to synchronize them to a universal clock, we define a way to represent space (continuous or discrete) and we start the simulation.
This approach is called "agent based" [61], where modeling consists in defining individuals and their interactions, generates a number of individuals and observes both numerically and statistically their behavior in a given habitat according to given parameters.

The strength of the agent based approach is its simplicity [60] following that approach is possible to define different elements of the model independently. The simple definition of a single agent with its simple rules of behavior determines the global behavior of the entire system. The simulation of phenomena such as the movement of a flock of birds, a social model on gorilla behavior or the simulation of a stock of fish is quite simple.
The weak point of this method is the lack of a standard. It is difficult to prove formal properties of this type of models as well as make a statistical analysis. Attempts have been made to define standards within the community that uses these models, or at least to create benchmarks to be followed in the creation of the various models [59],[58], but uncertainties persist for what concerns the definition of a unique method to model individuals.

Another classical approach is the analytical one, by means of systems of ordinary differential equations [57]. There is a vast literature in this regard with a number of well-studied mathematical models [56] [55].
Individuals are not handled individually but quantitatively as variables of the whole system. In this kind of approach we define a system of equations, some initials conditions evaluated at one specific time t_0, and then the system is able to provide a continuous function which describes the system's own evolution. It is possible to study the variables' values which define the system's state and their evolution up to a generic instant t_i which follows the initial state t_0.
This approach is definitely more structured than the previous one, it is easy to demonstrate the models' properties, and get a comprehensive study of its complexity.
All this is obtained at the cost of a much greater complexity of formulation. The systems of equations often require elaborate solutions through appropriate numerical methods. Another problem with this type of solutions is the lack of adaptability to some random elements intrinsic to these models and the difficulty to incorporate elements of probabilistic type.

Another approach is obtained considering the very strong parallelism between the process algebras and evolution of biological systems. One of the best ways to model the complexity of biological problems is using one of numerous formalisms based on process calculi.

A process can be seen as a living entity which has a creation moment, can reproduce creating copies of itself, can react to stimuli it receives on special channels and can change its behavior.
We have processes/animals that are born, live, reproduce and die, a series of interactions with other processes or with a permanent process/habitat.

There is another approach, which will be hereafter the field of interest of our work and will represent the starting point from which to build a formal apparatus of modeling. This kind of approach defines the class of "rewriting systems", [54] [53]. We will cover a particular computational model belonging to "rewriting systems" called P systems. For an

introduction to P System see chapter two.

1.2 Contribution and Thesis

This thesis is intended to show our contribution to the evolution of the P System formalism. The formalism variants we want to introduce in this work can be used as tools for modeling macro-biological phenomena.
We want to use as starting point the P systems with their high parallelism to obtain a formalism able to describe complex biological population problems. We first introduce a probabilistic structure to the rules of P systems, then we introduce elements which can be detailed and expanded with custom attributes. Interactions between elements are modelled by rules, to any possible outcome of a specific interaction is associated a specific rule. Rules associated to different interaction outcomes can be applied at same time on different elements.

Elements can be grouped in social or spatial structures. Set of individuals can interact each other in different stages of the simulation as a group rather than as single individuals. In case studies and examples we show step by step how it is possible to create a model. We start by describing which kind of elements we have, what attributes these elements have associated with them, choosing some empirical facts, defining these facts as interactions between elements and defining rules associated to these interaction outcomes.
In a simple way we can create models able to provide predictions and data output confirmed by empirical data sets of our case studies.
Our formalism can be used to model continuous space, successions of time-phases and other metrics. The rules, gathered in sets and subsets, provide a very modular and easy way to add new facts to a working model and extend the probabilistic structure of the formalism.

In chapter two we introduce some background notions. In particular, we formally define the P systems, explaining why we use this formalism as starting point.

We show that the most important feature of this formalism is given by its innate parallelism. A powerful characteristic that explains our efforts to adapt P systems to the study of groups of animals instead of chemical reactions into a cell.

In the third chapter a list of previous works on similar topics is presented, showing that scholars have already used P systems to represent population problems model, producing a significant number of results, papers and formalisms. We will focus on those papers that represent the basis of this work and part of the state of the art. Also, we will briefly explore other ways to model macroscopic biological phenomena like agent-based models and languages which take inspiration from process algebras.

From this chapter onwards we will show the evolution of our work which takes form in three formalisms, Minimal Probabilistic P systems (MMP), Attributed Probabilistic P systems (APP), Multilevel Attributed Probabilistic P systems (MAPP).
Each of these chapters will start with the reason why we need to add more features to the previous formalism and what new kind of phenomena we can model with such new tools. After a formal description of the proposed formalism we present one simple model to better understand the features of the formalism, then we go ahead with a real case study to show what we can do with our tools on a real problem, by comparing empirical observations on the field with our model-based predictions.

We take one step ahead with each formalism, and for each of them we have a dedicated chapter:

- in chapter four we show the first formalism, MPP systems, in detail. In this chapter we show the formalism associated with a real case study about European water frog populations. The proposed model generates the outcomes for many scenarios, and this outcomes are validated by empirical observation on the field.

- chapter five is dedicated to APP systems, reasons to pass from MPP to APP, a formal definition, an example and a case study about social structure in primates.

- chapter six is about MAPP systems, following the structure of the two previous chapters we introduce what we add to APP to obtain the MAPP systems, the reasons behind the new features, the formal definitions and two detailed case studies.

In the last chapter, the seventh, we show some possible future developments. In particular, a little discussion about the possible necessity of another step after MAPP systems to define a richer formalism, and what kind of problem can require this more advanced tool. Also a discussion about one particular way to use MAPP systems to produce models very similar to agent-based ones.

1.3 Publications

In "Barbuti R., Bove P., Schettini A. M., Milazzo P. and Pardini G. (2013). A computational formal model of the invasiveness of eastern species in European water frog populations. In Software Engineering and Formal Methods (pp. 329-344). Springer International Publishing" we provide in detail the definition of MMP systems in Chapter 4.

In "Barbuti R., Bove P., Milazzo P. and Pardini G. (2015). Minimal probabilistic P systems for modelling ecological systems. Theoretical Computer Science, 608, 36-56." we expand the MMP formalism and give a translation in to PRISM language.

In "Barbuti R., Bompadre A., Bove P., Milazzo P. and Pardini G. (2015). Attributed Probabilistic P systems and Their Application to the Modelling of Social Interactions in Primates. In Software Engineering and Formal Methods (pp. 176-191). Springer Berlin Heidelberg" we present the APP systems we will detail in Chapter 5

The paper "Multi-level Attributed Probabilistic P systems, a computational model for social and spatial ecological system (working title)" about the MAPP systems is still unpublished. We will introduce the MAPP systems in Chapter 6.

Chapter 2

Background and notations

In this chapter we will present a brief list of key concepts we use in the following chapters and also we will give formal definition of P systems. Moreover, we give background notions on probabilistic and stochastic model checking. Such form of model checking will be used to study properties of case studies in combination with Monte Carlo simulation.

2.1 Multisets and Indexing

The formalism of P systems, and everything that we develop from them, use a lot of the concepts of set, multiset, indexed set. In this section we define some notation we will use about those concepts:

- **free commutative monoid**: Given a set A, the free commutative monoid on A is the set of all finite multisets with elements drawn from A, with the monoid operation being multiset sum and the monoid unit being the empty multiset.
  For example, if $A = \{a, b, c\}$, elements of the free commutative monoid on A are of the form:
  $\{\epsilon, a, ab, a^2b, ab^3c^4, ...\}$
  The free commutative monoid extends the concept of free monoid and gives us the main tool we use in this thesis to work with sets and multisets in our formalisms.

- **multiset** is a generalization of the concept of a set that, unlike a set, allows multiple instances of the multiset's elements. The multiplicity of an element is the number of instances of the element in a specific multiset.
  We use strings to represent multisets of elements, where a character represents a symbol of an element of a multiset and we simply ignore the order of characters present in each string.
  So, to describe a multiset on the set of elements $A = \{a, b, c\}$ we use the strings s over A used as alphabet or $s \in A^*$. We use the operator star "*" (Kleene Star) over the alphabet of symbols to generate possibly infinite set of finite strings of elements. For our purpose the string "aaab", "abaa", "aaba" are equivalent and describe the same multiset.
  This way to describe multisets is useful when we use the operator "+" to append a string to another to describe the union over multisets $\uplus$.
  In this way to represent the operation $\langle a^2, b\rangle \uplus \langle a, b\rangle = \langle a^3, b^2\rangle$ We use "aab"+ "ab" = "aabab".
  With $|s|$ we denote the size (number of elements) of the multiset described by the

string s, and with $|s|_a$ the number of instances of element a of the set A contained in multiset.

- **one-to-one correspondences between sets and their indexing** In the following discussion we use finite ordered sets associated with other sets of the same cardinality. We use the elements of the ordered set to index the elements of associated set. The resulting indexing produces an order in the associated set. For instance the ordered finite set $A = \{a_1, \ldots, a_n\}$ is associated to the set $B = \{b_{a_1}, \ldots, b_{a_n}\}$ where elements of A are in one-to-one correspondence with elements of B so that b_{a_1} is associated with a_1, b_{a_2} is associated with a_2 and so on. Sometimes the associated set is a set of sets. For instance $B = \{B_1, \ldots, B_n\}$ is a set of sets where the j-th elements of the i-th set is referred as $B_{i,j}$.

2.2 Markov chain

A Markov chain (discrete-time Markov chain or DTMC [132]), named after Andrey Markov, is a random process that undergoes transitions from one state to another on a state space. It must possess a property that is usually characterized as "memorylessness": the probability distribution of the next state depends only on the current state and not on the sequence of events that preceded it. In the figure 2.1 we have a simple two state Markov chains.

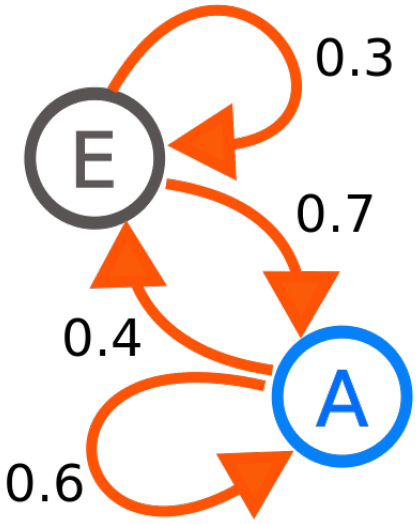


Figure 2.1: A simple two-state Markov chain

A Markov chain is collection of random variables X_t (where the index t runs through 0, 1, ...) having the property that, given the present, the future is conditionally independent of the past, in other words:

$P(X_t = j | X_0 = i_0, X_1 = i_1, \ldots, X_{t-1} = i_{t-1}) = P(X_t = j | X_{t-1})$.
If a Markov sequence of random variates X_n take the discrete values $a_1, \ldots, a_N$, then $P(X_n = a_n | X_{n-1} = a_{n-1}, \ldots, X_{t-1} = a_1) = P(X_n = a_n | X_{n-1} = a_{n-1})$.
and the sequence is called a Markov chain, a simple random walk is an example of a Markov chain.

2.3 P systems

P systems were introduced by Păun in [131] [130] as distributed parallel computing devices inspired by the structure and the functioning of a living cell.
A P system consists of a hierarchy of membranes, each of them containing a multiset of objects, representing molecules, a set of evolution rules, representing chemical reactions, and possibly other membranes. For each evolution rule there are two multisets of objects, describing the reactants and the products of the chemical reaction.
A rule in a membrane can be applied only to objects in the same membrane. Some objects produced by the rule remain in the same membrane, others are sent out of the membrane, others are sent into the inner membranes, which are identified by their labels.
Evolution rules can be applied more than once to different objects, with maximal parallelism, namely it cannot happen that some evolution rule is not applied when the objects needed for its triggering are available and not consumed by the application of any other rule.
Many variants and extensions of P systems exist that include features which increase their expressiveness and which are based on different evolution strategies.

The formalisms that we define in this thesis are variants of P systems which include features that allow us to describe populations dynamics and ecosystems. In particular, maximal parallelism is one of the features of P systems that can be useful for the modelling of this kind of systems. Indeed, the population studied by means of modelling techniques often evolve by stages in which all of the individuals are involved in the same activity (e.g. reproduction stages, hibernation, survival selection in winter, etc.). Stages are often determined by year seasons, but can also be determined by specific biological aspects of the species under study. Maximal parallelism, together with the use of evolution rule promoters [129], offer a simple and effective way of dealing with stage-based evolutions.

Maximal parallelism is not the only feature of P systems that could be useful for modelling populations and ecosystems.
Evolution rules, for instance, can easily describe interactions between individuals of a population described as a multiset of objects. On the other hand, we believe that the membrane hierarchy of P systems is not really necessary for the ecological application domain.
In facts, the membrane structure of a P systems can often be *flattened* into a single membrane [119] as we do in MPP and APP systems. Moreover, if spatial (and social) aspects of ecosystems and populations have to be taken into account, it is possible to resort to formalisms that explicitly deal with spatiality [118, 117, 116, 115] like the last one of our formalisms, the MAPP systems.

A P System consists of a *hierarchy of membranes* that do not intersect, with a distinguishable membrane, called the *skin membrane*, surrounding them all. As usual, we assume membranes to be labeled by natural numbers. Given a set of elements V, a membrane m contains a multiset of *elements* in V^*, a set of *evolution rules*, and possibly other membranes, called *child* membranes (m is also called the *parent* of its child membranes). Elements represent molecules swimming in a chemical solution, and evolution rules represent chemical reactions that may occur inside the membrane containing them. For each evolution rule there is a multiset of elements representing the reactants, and a multiset of elements representing the products of the chemical reaction. A rule in a membrane m can be applied only to elements in m, meaning that the reactants should be precisely in m, and not in its child membranes. The rule must contain target indications, specifying the membranes where the new elements produced by applying the rule are sent. The new objects either remain in m, or can be sent out of m, or can be sent into one of its child membranes, precisely identified by its label. Formally, the products of a rule are denoted

with a multiset of *messages* of the following forms:

- $(v, here)$, meaning that the multiset of elements v produced by the rule remain in the same membrane m;
- (v, out), meaning that the multiset of elements v produced by the rule are sent out of m;
- (v, in_l), meaning that the multiset of elements v produced by the rule are sent into the child membrane l.

We can assume that all evolution rules have the following form, where $\{l_1, \ldots, l_n\}$ is a set of membrane labels in $\mathbb{N}$.

$$u \to (v_h, here)(v_o, out)(v_1, in_{l_1}) \ldots (v_n, in_{l_n})$$

An evolution rule in a membrane m is called *dissolving* if its application causes the disappearance of m. In this case, the elements in m and the child membranes of m remain free in the parent membrane of m, and the evolution rules of m are lost. The skin membrane cannot be dissolved. A dissolving evolution rule is denoted by adding to the products the special message δ such that $\delta \notin V$:

$$u \to (v_h, here)(v_o, out)(v_1, in_{l_1}) \ldots (v_n, in_{l_n})\delta$$

Application of evolution rules is done in parallel, parallelism is maximal: namely at each evolution step a multiset of instances of evolution rules is chosen non-deterministically, such that no other rule can be applied to the system obtained, by removing all the elements necessary to apply all the chosen rules. The application of rules consists of removing all the reactants of the chosen rules from the system, adding the products of the rules by taking into account the target indications, and dissolving all the membranes in which a δ message has been produced.

Now, we formally define P systems:

Definition 1. *A* P System Π *is given by*

$$\Pi = (V, \mu, (w_1, \ldots, w_n), (R_1, \ldots, R_n))$$

where:

- V *is an* alphabet *whose elements are called* objects*;*
- $\mu \subset \mathbb{N} \times \mathbb{N}$ *is a* membrane structure*, such that* $(i, j) \in \mu$ *denotes that the membrane labeled by* j *is contained in the membrane labeled by* i*;*
- w_i *with* $1 \leq i \leq n$ *are strings from* V^* *is the initial multisets over* V *associated with the membranes* $1, 2, \ldots, n$ *of* μ *;*
- R_i *with* $1 \leq i \leq n$ *are finite sets of* evolution rules *associated with the membranes* $1, 2, \ldots, n$ *of* μ*;*

We show in Fig. 2.2 an example of P System over the alphabet $V = \{a, c, d, e\}$ in which all the main features of the formalism are used.

In this example the alphabet is $V = \{a, c, d, e\}$, we have two membranes, 1 and 2, so $\mu = \{(1, 2)\}$ to model the fact membrane 1 is the skin membrane and contain membrane

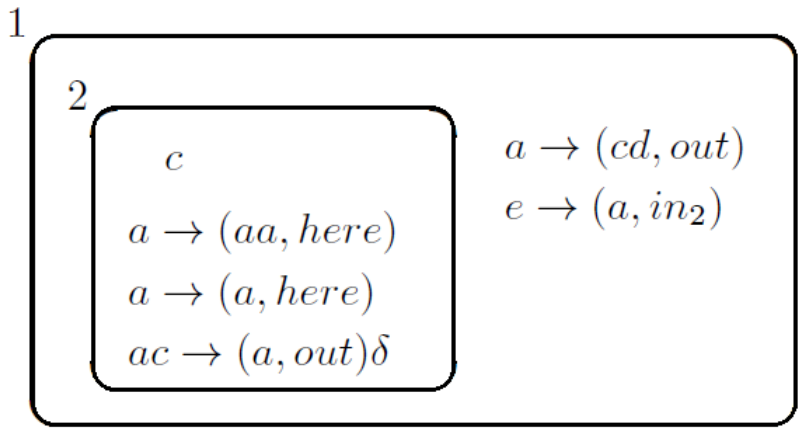


Figure 2.2: An example of P System with all kind of possible rules.

2.
The sets R_i are composed by the rules for membranes of the mode, $R_1 = \{a \to (cd, out), e \to (a, in_2)\}$ and the rule set for membrane 2 $R_2 = \{a \to (aa, here), a \to (a, here), ac \to (a, out)\delta\}$.
The initial tuple w is $w = (w_1, w_2)$ with w_1 empty and $w_2 = \{c\}$.
A computation works from an initial starting state towards an end state through a number of discrete steps. Each step involves iterating through all membranes in the P system and the application of rules, which occurs in both a maximally parallel and non-deterministic manner.
Working step-by-step, a computation halts when no further evolution can take place (i.e. when no rules can be applied). At this point whatever objects have been passed to the environment, or into a designated "result" membrane, are counted as the result of the computation. For istance the example system will use the rule $a \to (cd, out)$ to produce the output and push outside the skin membrane "cd" elements during the computation as his output.

2.4 Background on probabilistic and statistical model checking

In this section we present the PRISM probabilistic model checker as tool to apply probabilistic and statistical model checking to MPP models while still lot of work is required to adapt this tool to APP and MAPP systems too. Here we provide some background about this kind of model and a translation of MPP systems into the PRISM input language.

Model checking [113] is a technique for the verification of properties (expressed as logical formulae) holding in a system, based on the thorough exploration of its state space. Applicability of model checking is often limited to finite-state systems; in practice for many interesting systems the state space (albeit finite) is too big to be examined, therefore many techniques have been developed to tackle this problem.

Formally, properties are expressed in a *temporal logic*, such as the *Computational Tree Logic* (CTL) [112], constructed from a basic set of atomic propositions AP which are composed through the operators provided by the logic. A system's behavior is described by a *Kripke structure* M over AP, which is a tuple $M = (S, S_0, R, L)$ where: (i) S is the

finite set of states; (ii) $S_0 \subseteq S$ is the set of initial states; (iii) $R \subseteq S \times S$ is the transition relation; (iv) $L : S \to 2^{AP}$ is the labeling function associating, to each state, the set of atomic propositions which are true in that state.

The transition relation can describe non-deterministic behaviors, and it is assumed to be total ($\forall s \in S. \exists s' \in S. (s, s') \in R$). Executions of the system are represented by the infinite *paths* $\pi = s_0, s_1, s_2, \ldots$, where $s_0 \in S_0$, $s_1 \in S$ and $\forall i. (s_i, s_{i+1}) \in R$. In this paper we are mostly interested in analyzing *probabilistic* systems described by means of *Discrete Time Markov Chains* (DTMC), for which a probabilistic distribution is associated with the transitions exiting from a state. The memoryless property ensures that the probability of passing from one state to another depends only on the current state and, therefore, they are not affected by how the current state has been reached.

Temporal logics are used to describe sequences of transitions between states in the Kripke structure. In the non-probabilistic setting, we consider the Computation Tree Logic (CTL), which is composed of atomic propositions in AP, *path quantifiers* (**A**,**E**) and *temporal operators* (**X**,**F**,**G**,**U**). Path quantifiers are used to deal with branching in the Kripke structure, i.e. the different transitions exiting from a state.

A Kripke structure M verifies a CTL formula F over a path $\pi = s_0, s_1, s_2, \ldots$ ($(M, \pi) \models F$) according to the following rules.

$(M, \pi) \models \text{true}$		
$(M, \pi) \models ap$	iff	$ap \in L(s_0)$
$(M, \pi) \models \neg\varphi$	iff	$(M, \pi) \not\models \varphi$
$(M, \pi) \models \varphi \wedge \varphi'$	iff	$(M, \pi) \models \varphi \ \wedge \ (M, \pi) \models \varphi'$
$(M, \pi) \models \mathbf{A}\,\varphi$	iff	φ holds for all paths starting from s_0
$(M, \pi) \models \mathbf{E}\,\varphi$	iff	φ holds for some path starting from s_0
$(M, \pi) \models \mathbf{X}\,\varphi$	iff	$(M, \pi') \models \varphi$ with $\pi' = s_1, s_2, \ldots$
$(M, \pi) \models \mathbf{F}\,\varphi$	iff	φ holds for some state $s_i \in \pi$
$(M, \pi) \models \mathbf{G}\,\varphi$	iff	φ holds for all the states $s_0, s_1, s_2, \ldots \in \pi$
$(M, \pi) \models \varphi_1\,\mathbf{U}\,\varphi_2$	iff	there exist $s_i \in \pi$ such that φ_1 holds in $s_0, s_1, s_2, \ldots s_i$ and φ_2 holds in s_{i+1}

In our work, we make use of a probabilistic extension of the CTL, called *Probabilistic CTL* (PCTL) [111], which provides operators to query about quantitative properties of probabilistic systems. In particular, we are interested in the **P** operator, used to query about the probability of different executions. Among the different forms, we consider queries of the form $\mathbf{P}_{=?}[\varphi]$, which gives the probability that the evolution of the DTMC follows a path which satisfies the path property φ containing temporal operators.

We use the PRISM probabilistic model checker [109] to model and analyse the evolution of populations. The PRISM model checker is a mature tool which provides both *exact* model checking algorithms, based on iterative algorithms for approximating the measure of paths satisfying the formula being analyzed, and algorithms for *statistical* model checking, which are instead simulation-based. In statistical model checking [110], many random simulations are performed, and used to derive the probability measures of different system's executions, with some non-zero risk of erroneous results. However, simulation-based methods are far less demanding than exact methods (both in terms of space and time needed for their execution), and moreover the probability of erroneous results can always be bound by any small amount. Hence they constitute a powerful tool for the analysis of systems with a big state space on commodity hardware.

Chapter 3

State of the Art

The state of the art in the formal modelling of ecosystems consists of studies in which formal notations are used to model and simulate population and ecosystems dynamics. Also we will mention some notable examples of formalisms used over the years and which are part of the classical methods for modeling macro-biological systems.
Some of them, like rewriting systems, have been defined with the specific purpose of describing biochemical networks and activity of membranes inside cells.
Moreover, some of them have been inspired by process algebras based on language for concurrency theory.
A different approach to the problem is the use of the so called "agents". The agent-based models consist of a set of individual entities interacting with each other. The behavior of each agent is individually determined by an internal algorithm and external inputs.

According to this introduction we will divide this chapter in four sections:

- in the first section we present solutions derived by process algebra
- in the second section we present solutions derived by agent-based model
- in the third section we present solutions derived by rule based and rewriting systems
- in last section we present all other remarkable solutions which are not classifiable in previous sections

All these approaches have strong and weak points, to better understand what a "weak" or "strong" means in this context we will find a very interesting point of view in [133] where all these solutions are evaluated with properly reasoned criteria ("challenges").

3.1 Biology problems and process algebra

k-calculus was introduced in [143],[144] and is a calculus specifically designed for the task of representing and studying biological networks at the molecular level. Sites, proteins, protein complexes, and signatures have specific representations and operators. Definition of well-formedness for complexes are provided. Solutions are defined starting from sets of proteins and complexes. Reactions are defined by rewriting rules. We can have two kinds of basic reactions: activation and complexation with different activation condition. We can have basic reactions (all reactant are simple protein) or otherwise it will be said complex. In the end we define a k-system a pair $< S, R >$ where S is a solution and R is a finite set of basic reactions.
K-calculus can model commuting and competing behaviours and even some forms of synchronization, also a definition of bisimilarity is provided (barbed bisimilarity).

k-calculus is a simple language of complexations and activations targeting core molecular biology. Synthesis and degradation were not considered, neither were decomplexations. No locations were introduced in the language either: the idea here was to have the simplest meaningful language equipped with a barebone operational semantics and see if anything interesting happens. In conclusion a translation is provided for k-calculus in $\pi - calculus$, another more evolved process algebra formalism.

Bio-PEPA (2008) Developed by Ciocchetta and Hillston [42, 41], Bio-PEPA is a process algebra for the modelling and the analysis of biochemical networks. Bio-Pepa is

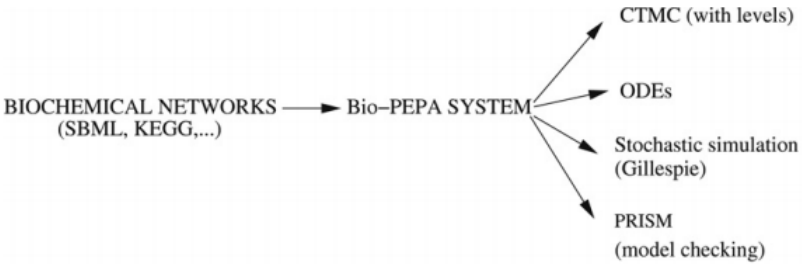


Figure 3.1: Schema of the Bio-PEPA framework

an extension of PEPA, a process algebra originally defined for the performance analysis of computer systems. A major feature of Bio-PEPA is the possibility to represent explicitly some features of biochemical models, such as stoichiometry and the role of the species in a given reaction. Furthermore functional rates are introduced to express general kinetic laws. Each action type represents a reaction and it's associated with a functional rate. Bio-PEPA is equipped with an operational semantic and a stochastic labeled transition system based on discrete levels of concentrations. The representation in terms of discrete levels of concentration is also reflected in the definition of the continuous time Markov chains (CTMC) derived from the system.
Bio-PEPA can be used, with some little foresights, to model macroscopic population problem too, but its features are better used for bio chemistry problems.

PALPS [48] [47] is a Process Algebra with Locations for Population systems. PALPS allows us to produce spatially-explicit individual-based ecological models and to reason about their behavior. PALPS has two abstraction levels: At the first level, PALPS may define the behavior of an individual of a population and, at the second level, it may specify a system as the collection of individuals of various species located in the space. In PALPS, individuals move through their life cycle while changing their location, and interact with each other in various ways such as predation, infection or mating. Furthermore, they propose a translation of a subset of PALPS into the probabilistic model checker PRISM.
The structure of PALPS is based on basic entities describing: species, locations (with topological relations), channels and sets of attributes. Expressions (both logical and arithmetic) over these entities define models following the syntax of the language. Technically, PALPS can be considered as an extension of CCS with probabilistic choice, locations and location attributes.
PALPS is another important inspiration for our work, we try to do the same by proposing a translation of subset of MPP systems (the first of our formalisms we propose in Chapter 4) into the probabilistic model checker PRISM. Also the idea of multilevel abstraction

used to model ecological models was important to define the MAPP System formalism (in Chapter 6).

Other relevant biological applications of process algebras include:
Bio-calculus (1999) Nagasaki, Onami, Miyano and Kitano in [4] define the Bio-calculus as an expression system designed to make a bridge between biology and computer science. It can be used to describe and simulate some kind of molecular interaction.
Applications of $\pi-$calculus (1992) More than a decade ago, Regev and Shapiro published their pioneer work on the representation of signaling pathways with $\pi-$calculus [22, 21]. Their idea is to describe metabolic pathways as $\pi-$calculus processes and in [26] they showed how the stochastic variant of the model (BioSpi), defined by Priami in [27], can be used to represents both qualitative and quantitative aspects of the systems described.

Brane Calculi (2005) More details of membrane interactions have been considered by Cardelli in the definition of Brane Calculi [1, 25], which are elegant formalisms for describing intricate biological processes involving membranes. Moreover, a refinement of Brane Calculi have been introduced by Danos and Pradalier in [24].

SpacePi (2008) John, Ewald and Uhrmacher devolepd an extension of π-calculus called SpacePi [5]. In this formalism π-processes are embedded into a vector space and move individually. Only processes that are sufficiently close can communicate. The SpacePi extends π-calculus also in time, the operational semantics of SpacePi defines the interactions between movement, communication, and time-triggered events.

3.2 Biology problems with agent-based solutions

VORTEX [46] is an individual-based simulation program that models the effects of mean demographic rates, demographic stochasticity, environmental variation in demographic rates, catastrophes, inbreeding depression, harvest and supplementation, and metapopulation structure on the viability of wildlife populations. The model facilitates analysis of density-dependent reproduction and changing habitat availability, and most demographic rates can optionally be specified as flexible functions of density, time, population gene diversity, inbreeding, age, and sex. VORTEX projects changes in population size, age and sex structure, and genetic variation, as well and estimates probabilities and times to extinction and recolonization.
More model which use VORTEX can be find in [45], [44], [43], all of them are very well documented and easy to be implemented.

Multi-scale DISPAS [51] is one of major sources of inspiration for the MAPP systems we present in Chapter 6. From the abstract of the papers we quote "DISPAS is an agent-based simulator for fish stock assessment developed as a decision making support for the sustainable management of fishery. In this work we enlarge the underlying model of DISPAS allowing it to model and simulate a multi-scale scenario.We retain the currently available spatial scale, able to represent a limited average region of the sea, and we introduce a new spatial macro-scale, able to represent the whole sea. At the macro-scale a single agent represents an area of five square nautical miles and manages groups of fish in different age classes. The interactions among the macro agents permit the exchange of individuals of each class among neighbor areas [..]". Multi-scale DISPAS propose a two level simulator model. The space is represent in a hexagon grid, inside any of the hexagon of the grid agent-fish are managed locally (first level of the simulation) then exchange of fish among neighboring hexagons is managed in the whole grid (second level of the simulation).

This management of the phenomena on multi-scale level is a very important feature we try to add to our formalism. The multi-level computational step allows us to represent very complex and structured problems like the case study we propose in Chapter 6.

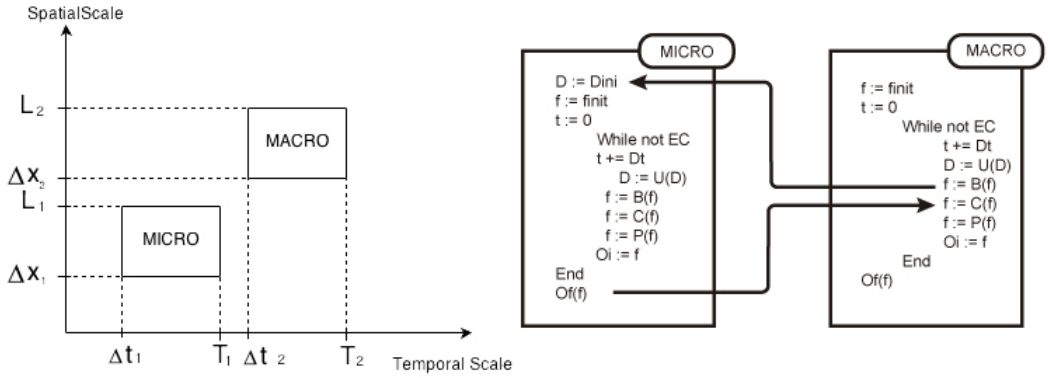


Figure 3.2: A Scale Separation Map (left) showing that the two CAs can be coupled by a micro-macro Coupling Template (right)

In [37] an individual based stochastic modelling approach to derive models for studying speciation by sexual selection is proposed. Specific models can be derived by instantiating genotypes of individuals, and production and cancellation rates relative to such genotypes. The methodology is based on Gillespie's algorithm for the simulation of chemical reactions [39]. The algorithm, unlike most procedures for the numerical solution of differential equations describing chemical kinetics, never approximates infinitesimal time increments by finite time steps. Consequently, the time is increased exactly to the time in which the next event occurs. Thus, the resulting simulation algorithm is characterized by a higher precision with respect to other simulation methods.
Therefore, in the paper it is proposed, rather than a model, a systematic methodology for the construction of computational models for sympatric speciation. The purpose of the paper is twofold: to provide biologists with a methodology for specifying individual based speciation models, and to show how, by means of a simulator, these models can be used to validate evolutionary hypotheses.
The proposed methodology and the purpose of the paper help us in this thesis to balance the complexity of our formalism, the goal is very similar and what we propose in this thesis is aimed to be used by biologist with some basic modeling knowledge.

In the paper [36] is described a stochastic, individual-based, explicit genetic model tailored for the cichlid system living in the lake Apoyo in Nicaragua.
The results show that relatively rapid ($<$ 20 000 generations) colonization of a new ecological niche and (sympatric or parapatric) speciation via local adaptation and divergence in habitat and mating preferences are theoretically plausible if: (i) the number of loci underlying the traits controlling local adaptation, and habitat and mating preferences is small; (ii) the strength of selection for local adaptation is intermediate; (iii) the carrying capacity of the population is intermediate; and (iv) the effects of the loci influencing nonrandom mating are strong.
This model is interesting because it is used to verify property of the system and to validate theories with in silico experiment. Is not the model itself but its use which perfectly shows what we want from this kind of solutions. What kind of data output we desire when we

apply our modelling tools to real problems with real data.

3.3 Biology problems by rule based and rewriting systems (P systems)

Rule base systems [139] roughly speaking consists of a knowledge base and an inference engine (see Figure 3.3). The knowledge base contains rules and facts. Rules always express a conditional, with an antecedent and a consequent component. The interpretation of a rule is that if the antecedent can be satisfied the consequent can too. When the consequent defines an action, the effect of satisfying the antecedent is to schedule the action for execution.

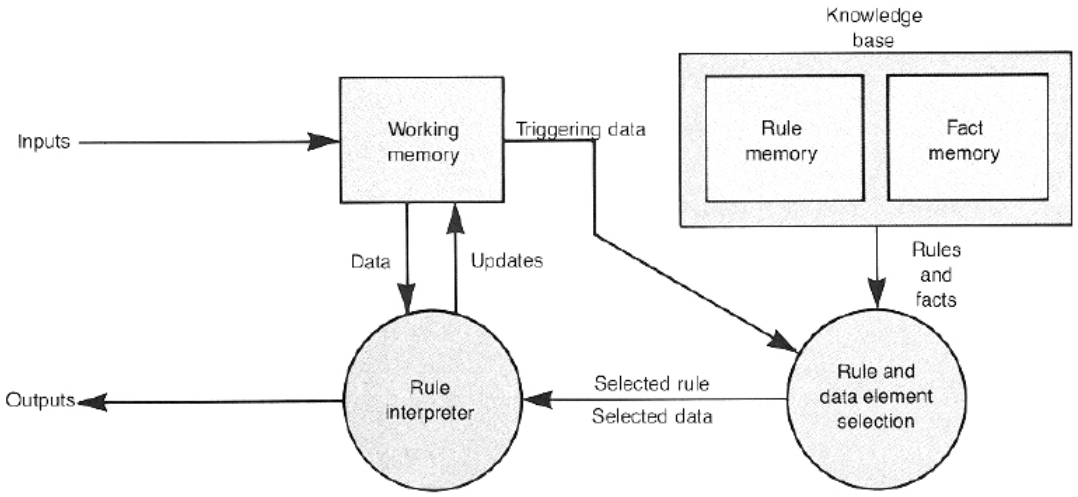


Figure 3.3: A simple RBS consists of storage and processing elements, which are often referred to respectively as the knowledge base and the inference engine. The basic cycle of a BBS consists of a select phase and an execute phase. During the execute phase, the system interprets the selected rule to draw inferences that alter the system's state. System storage includes components for long-term static data and short-term dynamic data. The long term store, which is this knowledge base, contains rules and facts. Rules specify actions the system should initiate when certain triggering conditions occur.

When the consequent defines a conclusion, the effect is to infer the conclusion.
Because the behavior of all RBSs derives from this simple regimen, their rules will always specify the actual behavior of the system when particular problem-solving data are entered. In so doing, rules perform a variety of distinctive functions:

1. They define a parallel decomposition of state transition behavior, thereby inducing a parallel decomposition of the overall system state that simplifies auditing and explanation. Every result can thus be traced to its antecedent data and intermediate rule based inferences.

2. They can simulate deduction and reasoning by expressing logical relationships (conditionals) and definitional equivalences.

3. They can simulate subjective perception by relating signal data to higher level pattern classes.

4. They can simulate subjective decision making.

Rewriting systems was a special class of rule-based systems and, in non-deterministic form, brings us to the core concept of P systems. The idea of *rules*, which modify a state following some sort of deterministic or probabilistic structure, was born with rule-based models and is the main idea on which we will be working in this thesis to model events. Some very interesting works about development of this formalism can be found in [140] and [142].

Probabilistic P systems [50], [49] is another important formalism for our work. Probabilistic P systems introduce probabilities associated to the rules of the system and others features we will propose in Chapter 4 with our MMP systems.

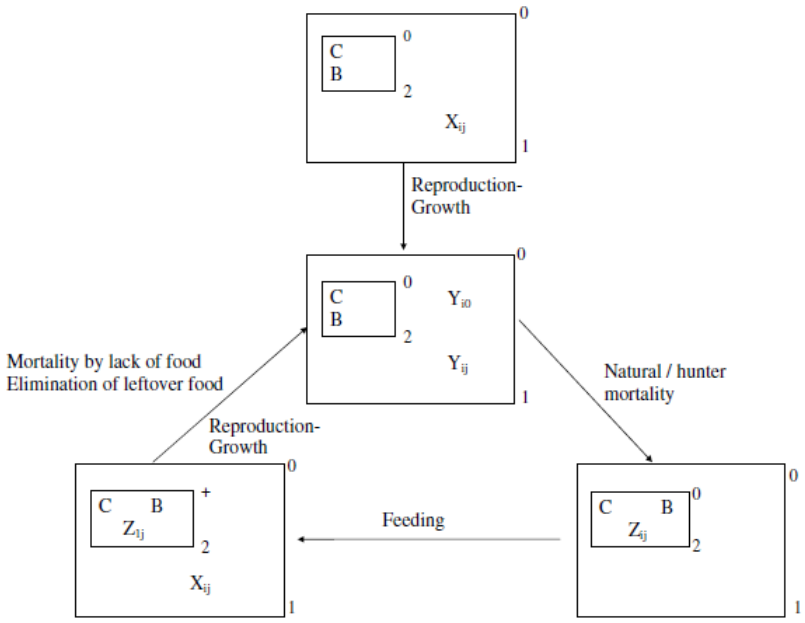


Figure 3.4: Structure of the P system computational step divided in the four stages

What we find in this interesting model is:
- constant probabilities associated to the rules
- The P system model of the case study implements a computational step composed by four stages (see fig3.4). The first one is devoted to the reproduction of the diverse species in the ecosystem. Then, the animals' mortality is analyzed according to different criteria. The third stage analyzes the amount of food in the ecosystem. In the last stage, the removal of animals due to lack of food takes place.
- in the alphabet of the system the individuals are represented by symbols X, Y and Z. Each of these symbols represented the same animal but in different states. Moreover, each symbol is indexed with two indexes i and j (index i is associated with the species and index j is associated with their age). In Chapter 5 with APP systems we want to extend this concept and to give to that kind of index the intuitive semantic of attributes associated to individuals of the system

In conclusion, probabilities associated to rules, multi stage running of simulation, index

(attributes) associated to symbol of the alphabet are all very important concept used in this work and which we expand in our formalisms in the following chapters.

In [40] the authors presented one P systems based general framework for modeling ecosystems dynamics.

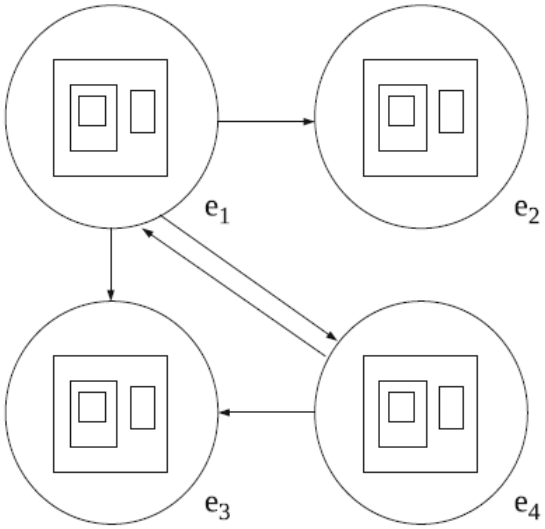


Figure 3.5: multienvironment structure

In this framework ecosystems are described by means of multienvironment P systems consisting of a finite number of environments, each of them having a specific P system with active membranes.
Inside these specific P systems each rule has associated a real number from $[0, 1]$ which depends on the left-hand side of the rule and the current state of the system run. This framework is an extension of the previous one [50] and adds the possibility to define more than one specific habitat with a specific set of rules to allow the elements to move between habitats.
This formalism is another interesting attempt to model population problems over a large territorial extension and with not only one kind of habitat. The problems the authors face in the paper are very well defined and helped us in the development of our formalisms.

Metabolic P systems presented in [38] is a variant of P systems used to define, in a biochemically realistic way, the evolution of P systems representing biological phenomena. This formalism is not related with population problems but gives an interesting point of view about the use of rules in P systems.
The leading principles of the MPA are the following:
- Reactants are distributed among all the rules step by step according to a "competition" strategy.
- If different rules need the same reactant, then each of these rules gets a portion of the available substance, in a percentage that is proportional to its reaction strength (reactivity) at that step.
- The reactivity of a rule at a given instant depends on the state of the system, defined as the concentration and localization of all substances.
- According to its stoichiometric "reading", any rule determines its own reaction unit and therefore the amount of substances which it consumes and produces.

We summarize these principles with four statements:
(1) Rules compete for object populations.
(2) Objects are allocated to rules according to a mass partition principle.
(3) Partition factors are determined by "reaction maps".
(4) A "Metabolic rule" r consumes/produces integer multiples of a reaction unit ur which generalizes the notion of molar unit (Avogadro's principle).
These pinciples can be not directly associated to population problems but the idea behind them can be ispiring in order to find new solutions.

in **Quorum sensing P systems** [135] is proposed one P systems variant by considering bacterium quorum sensing (QS) phenomena as the basis of the new approach. The QS mechanism is a communication strategy based on diffusible signals, which kick-in under high cellular density. In particular, it is generally observed that, when a QS process is activated, the concentration of the signal molecules reflects the number of cells in the colony, or at least the number of cells in a particular physiological state. Bacteria can then respond to variations of the concentration of signal molecules and, when this value exceeds a specific threshold limit that indicates the population is "quorated", they start to behave in a coordinated way. The resulting systems is a multi-environment models where single cells:
- a cell can store a natural number $s \geq 0$, which can become arbitrarily large and which represents the number of signal molecules currently present inside the cell
- a cell can store at most one symbol from a given alphabet Q, which represents the particular physiological state of the cell, we in fact call the alphabet Q the set of states
- a cell, depending on the value s, can change its state and simultaneously consume some signal molecules in order to produce some new ones inside the cell and/or release more signal molecules into a given environment
- an environment can store a natural number $t \geq 1$, which cannot exceed a given capacity, and which represents the number of signal molecules currently present in that environment. Environments are in fact considered as being passive repositories for signal molecules that act as finite buffers for communication between cells
- finite amounts of signal molecules can diffuse from an environment to a certain cell depending on the values s and t
The evolution of the system consist in steps where:
- all the cells that can evolve by means of at least one rule must evolve in parallel at the same time within the same transition step
- at most one rule per each cell can be used to modify the content of a cell within the same transition step
- the content of an environment j, with $1 \geq j \geq h$, can be modified by different releasing and diffusion rules, which may be applied in parallel to different cells connected to the environment j, these rules must be selected such as to make sure that, after their use, the value stored by this environment will be at least zero and no larger than the capacity c_j (capacity associated to the environment j).
The Quorum sensing P systems was important to help us to design the features of our MAPP systems (which will be shown in Chapter 6) and was useful to understand what kind of tools we will need when we want to model multi-environment phenomena. Also the signal structure between elements of the systems will be another important feature to manage in our work using a multilevel computation (only parallel among the membranes of the same level).

Dynamical Probabilistic P systems [145] are very interesting since they have to

face the same problem that led to the definition of the first formalism that we presented in this thesis (MPP systems).
With Probabilistic P systems (cited above) as a starting point, two assumptions which seem quite unnatural from a biological point of view are made:
(1) one uses the notion of priority relations among rules, which should be implicitly described by probabilities.
(2) probability values are initially assigned to rules and never changed during a computation, which means giving a static description of a dynamical system.
To overcome these two assumptions, a probabilistic structure which weighs the different rules on every interaction is created.
Summarizing, for each evolution step:
for each membrane, the probability distribution is computed.
for each membrane, the objects are assigned to the rules for as long as possible.
for each membrane, the multisets are updated accordingly.

The critical phase is when probability distributions are computed. This part is divided, as in MPP systems in:

Checking for rule applicability: we evaluate which is the applicable subset of rules, starting from the reactants multiset present in the current state.
Evaluation of pseudo-probabilities: we evaluate, in a parallel and independent way from any other rule, the probabilistic function of every rule that can be applied.
Evaluation of normalized probabilities: we normalize the weight of every rule, to have their overall probabilities ready to be used in the next phase.
In the related work some implementations of Lotka-Volterra models that show the system potential are presented. In conclusion, Dynamical Probabilistic P systems have been shown to be an extremely valid example of how non-static probabilistic structures can be attributed to P systems to describe events happening to biological populations. This model has also reinforced and validated the questions which gave birth to the idea of developing the formalisms present in this thesis.

We can find a valid alternative to the use we make of the PRISM language in chapter 4 in article *Executable Specifications of P Systems* [146], where Maude is proposed as an analytic tool and P system emulator.
Maude is a software system developed around the Maude language. Core Maude is the Maude interpreter implemented in C++, it provides the Maude 's basic functionality. Full Maude is an extension written in Maude itself, allowing combination of various Maude modules to build more complex modules. Maude can be used for many applications with competitive performance and advantages over the conventional code. The current Maude implementation can execute syntactic rewriting with speeds from half a million to several million rewrites per second, depending on the particular application and machine. It is able to work well with multisets having millions of elements.
Maude is essentially a mathematical language. The basic programming statements are equations, membership assertions, and rules. Their rewriting semantics is given by the fact that instances of the left hand side pattern are replaced by corresponding instances of the right hand side. A Maude program containing only equations and membership assertions is called a functional module.
The equations are used as rules (equational rewriting), and the replacement of equals for equals is performed only from left to right.
A Maude program containing both equations and rules is called a system module. Rules are not equations, they are local transition rules in a possibly concurrent system. Unlike

for equations, there is no assumption that all rewriting sequences will lead to the same final result, and for some systems there may not be any final states. The functional modules define a functional sub-language of Maude, and the system modules extend the purely functional semantics of equations to the concurrent rewriting semantics of rules.

A P system can be naturally represented as a collection of Maude modules, the Maude semantics of the module M is not the same with the P system semantics, therefore we must associate with M the appropriate semantics based on the maximal parallel rewrite relation. The article shows some simple P systems translated in Maude and emphasizes its use as a complex tool, able to execute P systems specifications, and then to verify in a rigorous way the desired properties of the specified P system.

3.4 Other solutions for Biology problems

Obviously **differential equations** can be used to create models about a large array of phenomena. These models use mathematical equations that contain derivatives, either ordinary derivatives or partial derivatives. They describe the rate of change of continuous variables. They are typically used for modeling dynamical systems in several areas other than biology.
Systems of non-linear ordinary differential equations (ODEs) have been used in systems biology to describe the variation of the amount of species in the modeled system as a function of time. They have been applied to all kinds of biological pathways [35, 34, 33, 31]. With a fully detailed kinetic model, one can perform time-course simulations, predict the response to different inputs and design system controllers. However, building ODE models requires insight into the reaction mechanisms to select the appropriate rate laws, and experimental data to estimate the kinetic parameters. Other types of differential equations, such as stochastic differential equations (SDEs) and partial differential equations (PDEs) can be used respectively to account for stochastic effects and spatial distribution [30]. Piecewise-linear differential equations (PLDEs) have been used to integrate discrete and continuous features in gene regulatory networks [29, 32].

Cellular automata Were created by von Neumann and Ulam in the 40's [28]. They are discrete dynamic models that consist on a grid of cells with a finite number of states. A cellular automaton has an initial configuration that changes at each time step through a predefined rule that calculates the state of each cell as a function of the state of its neighbors at the previous step. They are specially suited for modeling complex phenomena in a scale-free manner and have been used in biological studies for a long time [13]. Due to their spatial features their main applications are related to molecular dynamics and cellular population dynamics.

Petri nets Were created by Carl Adam Petri in the 60's for the modeling and analysis of concurrent systems. They are bipartite graphs with two types of nodes, places and transitions, connected by directed arcs. Places hold tokens that can be produced (respectively, consumed) when an input (respectively, output) transition fires. The execution of a Petri net is non-deterministic and specially suited for distributed systems with concurrent events.
Their application to biological processes began in 1993, by the work of Reddy and coworkers, to overcome the limitations in quantitative analysis of metabolic pathways [20]. There are currently several Petri net extensions (e.g.: colored, timed, stochastic, continuous, hybrid, hierarchical, functional), forming a very versatile framework for both qualitative and quantitative analysis. Due to this versatility, they have been used in metabolic [3, 19, 18],

gene regulatory [137, 138], and signaling networks [17, 23, 6, 16]. Also, they are suited for integrating different types of networks, such as gene regulatory and metabolic [2].

Lindenmayer systems (or L systems) are one of the oldest formalisms introduced and developed in 1968 by Aristid Lindenmayer [15].
An L system is a formal grammar most famously used to model the growth processes of plant development.

Boolean networks In 1969 Kauffman introduced Boolean networks to model gene regulatory networks [14]. They consist of networks of genes, modeled by boolean variables that represent active and inactive states. At each time step, the state of each gene is determined by a logic rule which is a function of the state of its regulators. The state of all genes forms a global state that changes synchronously. For large network sizes (n nodes) it becomes impractical to explore all possible states cause the exponential time complexity$O(2^n)$.
This type of model can be used to find steady-states (called attractors), and to analyze network robustness [8]. Boolean networks can be inferred directly from experimental gene expression time-series data [12, 11]. They have also been applied in some studies to model signaling pathways [7, 136]. To cope with the inherent noise and the uncertainty in biological processes, stochastic extensions like Boolean networks with noise [10] and Probabilistic Boolean networks [9] were introduced.

Chapter 4

Minimal Probabilistic P Sysyems (MPP systems)

4.1 Introduction to MPP systems, formal definition and semantics

The variant of P systems we define in this chapter has a minimal set of features useful for modelling population dynamics. We consider for our purpose *flat P system* [114], namely P systems consisting of a single membrane. The key ingredients we consider are (i) evolution rules with functional rates, (ii) probabilistic maximal parallelism and (iii) rule promoters. The aim of this first variant of P systems is to make modelling of populations easier, by avoiding unnecessary functionalities in the modelling formalism.

In models of population dynamics, evolution rules are used to describe events such as reproduction, death, growth, and so on. In general, there may be several rules describing one of these events, and any of these rules can possibly be applied to the same individual but only one among them will be selected. In fact rules, to be applied, compete over individual (as reagents) with each other.

For instance, the same female individual may be involved in the event "reproduction" and the event is described by one rule for each possible outcome. Let us suppose there sexist one rule that models the birth of one male offspring and is in competition with one rule which models the birth of one female offspring and in our model just one female individual is present. These two rules compete with each other over that female individual to model the different outcomes of the mating.
If the first rule is selected then that rule consumes the female individual and one male offspring is added to the system, the second rule "loses" the chance to use the object representing the female individual. No female offspring will be born because the second rule cannot be applyed.

Some rules may be more likely to be applied than others since the events they describe are more likely than others (for instance, some females may have a sexual preference for some specific kind of male).
Associating rates with rules allows us to choose them in a probabilistic way, where probabilities are proportional to the rates. Moreover, by allowing rates to be functions, rather than constant values, we achieve that the probability of applying a rule can depend on the current state of the system (for instance on the size of the population, or on the number of individuals of a specific kind). Probabilistic choice of rules have been considered in many formalisms for modelling biological systems as we have seen in the previous chapter

[127],[126], [125], [124], [123], [122], [121]. In our formalism we use the probabilities of rules in conjunction with maximal parallelism.

Populations often evolve by stages (e.g. reproduction, selection, etc.) in which (almost) all of the individuals are involved. By combining maximal parallelism with probabilistic choice of reactions we allow the whole population to evolve in a coherent way and, at the same time, each individual to follow its own fate.

Finally, since in different stages of the evolution of a population different kinds of events may happen, we need a way to enable different sets of rules depending on the current stage. For instance, during a reproduction stage only reproduction rules should be enabled, whereas during a selection stage only death/survival rules should be enabled. In order to obtain this result we exploit rule promoters, which can be used to enable/disable a set of rules by simply including/removing an object from the state of the system.

Definition 2 (MPP system)**.** *A* Minimal Probabilistic P system *is a tuple* $\langle V, w_0, R\rangle$ *where:*

- V *is a possibly infinite alphabet of objects, with* V^* *denoting the universe of all multisets having* V *as support;*
- $w_0 \in V^*$ *is a multiset describing the initial state of the system;*
- R *is a finite set of evolution rules having the form*

$$u \xrightarrow{f} v \mid_{pr}$$

where $u, v, pr \in V^*$, $u \neq \emptyset$, *are multisets (often denoted as strings) of* reactants, products *and* promoters*, respectively, and* $f : V^* \mapsto \mathbb{R}^{\geq 0}$ *is a* rate function.

The semantics of MPP systems are defined as a Discrete Time Markov Chain, where a *state* (or *configuration*) is a multiset of objects in V^*, and each probabilistic transition models a maximally-parallel step according to the semantics presented in the following.

Apart from what concerns probabilities, the definition of the MPP semantics follows the lines of the semantics of P systems proposed in [120]. The idea is to represent a maximally parallel application of rules as a sequence of applications of single rules. During this sequence of rule applications, reactants are removed from (a copy of) the current state and added to a temporary multiset. When no more rules can be applied to the remaining objects, the temporary multiset is merged with remaining objects to obtain the next state of the system, ready for another maximally parallel step. The formal definition of this process requires the use of different transition relations: one for single rule applications, one for sequences of single rule applications, and one for maximally parallel steps.

Formally, the evolution of a MPP system $\langle V, w_0, R\rangle$ is given by a sequence of probabilistic maximally parallel steps described by transitions of the form

$$w \overset{\overline{r},\, \overline{p}}{\longmapsto}_R w'$$

where: w and w' are respectively the source and target configuration; $\overline{r}$ is a sequence of (instantiations of) evolutions rules from R; and $\overline{p} \in [0, 1]$ is the probability of the transition. In order to define the semantics, we make use of an auxiliary transition relation used to derive maximal multiset of rule applications. The inference rules defining the semantics are formally defined as follows:

(rule application)
$$\frac{r_i : \; u \xrightarrow{k} v \mid_{pr} \in R \qquad u \subseteq w_a \qquad pr \subseteq w \qquad K = \{\!|k' | u' \xrightarrow{k'} v' \mid_{pr'} \in R,\; u' \subseteq w_a,\; pr' \subseteq w|\!\} \qquad p = \frac{k}{\sum_{k' \in K} k'}}{(w_a, w_p) \xrightarrow{r_i,\, p}_{(R,w)} (w_a - u, w_p + v)}$$

(single rule sequence)
$$\frac{(w_a, w_p) \xrightarrow{r_i,\ p}_{(R,w)} (w'_a, w'_p)}{(w_a, w_p) \xrightarrow{[r_i],\ p}{}^{+}_{(R,w)} (w'_a, w'_p)}$$

(multi-rules sequence)
$$\frac{(w_a, w_p) \xrightarrow{r_i,\ p_i}_{(R,w)} (w'_a, w'_p) \qquad (w'_a, w'_p) \xrightarrow{\overline{r},\ \overline{p}}{}^{+}_{(R,w)} (w''_a, w''_p)}{(w_a, w_p) \xrightarrow{\overline{r}@[r_i],\ p_i.\overline{p}}{}^{+}_{(R,w)} (w''_a, w''_p)}$$

(step rule)
$$\frac{(w, \emptyset) \xrightarrow{\overline{r},\ \overline{p}}{}^{+}_{(R(w),w)} (w_a, w_p) \qquad (w_a, w_p) \not\longrightarrow_{(R(w),w)}}{w \overset{\overline{r},\ \overline{p}}{\longmapsto}_R w_a + w_p}$$

where $[r_i]$ denotes the sequence of rule application composed by the single element r_i, and @ denotes the concatenation of sequences. Given a system state, w, the rule (step rule) describes the evolution in a new state by the $\overset{\overline{r},\ \overline{p}}{\longmapsto}_R$ relation, where $\overline{p}$ is the probability of the transition, and $\overline{r}$ is the sequence of applied rules. According to (step rule), both $\overline{r}$ and $\overline{p}$ are derived from $(w, \emptyset) \xrightarrow{\overline{r},\ \overline{p}}{}^{+}_{(R(w),w)} (w_a, w_p)$, which represents the application of the sequence of evolution rules $\overline{r}$ from the state w. In this case, we denote as $R(w)$ the set of all rules in which each rate function f has been replaced by the constant obtained by applying f to the current state w. The function f is total on all possible multiset over V^* and depends only on the current state, when all constants for all rules are obtained applying the different f functions then the probability of each rule is obtained using some normalization process (where each single probability depends on the rate of all rules).

More formally, we have:

$$R(w) = \left\{ u \xrightarrow{f(w)} v \mid_{pr} \;\middle|\; u \xrightarrow{f} v \mid_{pr} \in R \right\}.$$

The use of such an instantiation function allows us to assume in the rest of the semantic rules that the rate associated with an evolution rule is simply a constant k.
Intuitively, the semantic definition states that all the rules to be applied in a step are selected in a probabilistic way, one by one, from the set of applicable rules. To keep track of the effect of the application of a rule, a pair (w_a, w_p) is modified after the application of each rule, where w_a denote the reactants still /empavailable in the current step, and w_p denote the *products* to be added at the end of the step. Therefore, each rule application causes their reactants to be removed from the available reactants w_a, and their product to be added to multiset w_p. When no further rule can be applied to w_a the new state, which is composed be the unused objects in w_a plus the suspended products in w_p, is produced. The transition relation $\xrightarrow{\overline{r},\ \overline{p}}{}^{+}_{(R,w)}$ used in the premises of (step rule) is defined in rules (multiple rules sequence) and (single rule sequence) as the transitive closure of $\xrightarrow{r,\ p}_{(R,w)}$. The latter is, in turn, defined in rule (rule application) and corresponds to the application of a single evolution rule $r_i \in R(w)$. When a rule is selected, its application consists in removing its reactants from w_a and adding its products to w_p. The w_p multiset will collect all products of all applied rules. Note that each rule is applied with a probability that is proportional to the rate of the rule computed in the original state w and properly normalized. Moreover, the conditions for applicability of a rule are evaluated with respect to the original state w as concerns promoters, and with respect to the intermediate state w' as concerns reactants. Once objects in w_p are such that no further rule in R can be applied to them, as captured by the premise $(w_n, \overline{w}_n) \not\longrightarrow_{(R(w),w)}$ of (step rule), then this means that the sequence of

applications $\overline{r}$ yields the configuration $w_a + w_p$, composed of the unused objects w_a and of the new products w_p. Finally, the probability of a transition between two states of the DTMC is given the following rule:

$$\text{(state transition probability)} \qquad \frac{PR = \{(\overline{r}, \overline{p}) |\ w \overset{\overline{r},\ \overline{p}}{\longmapsto}_R w'\} \qquad p\ =\ \sum_{(\overline{r},\overline{p}) \in PR} \overline{p}}{w \overset{p}{\Longrightarrow}_R w'}$$

which states that the probability to pass from a state w to a state w' is given by the sum of the probabilities to pass from w to w' by means of any possible sequence of rule applications. Note that relation $\overset{r_i,\ p}{\longrightarrow}_{(R,w)}$ essentially corresponds to the transitions of a Discrete Time Markov Chain, while relation $\overset{\overline{r},\ \overline{p}}{\longmapsto}_R$ corresponds to paths in such Markov Chain.
Consequently, in $w \overset{p}{\Longrightarrow}_R w'$, p is the sum of the probabilitis of all path leading from w to w' in the Markov Chain gived by $\overset{r_i,\ p}{\longrightarrow}_{(R,w)}$. Hence, $\overset{p}{\Longrightarrow}_R$ is, in turn, a Discrete Time Markov Chain.

4.2 Probabilistic Rules and Control Objects

In the formal definition of the formalism we see how probabilistic rules were introduced, and the promoters were used as control objects.
Probabilistic functions associated to the rules manage the non-determinism present in P systems. Facts and actions are described in the model by rules, how much likely this facts will happen within the population is described by the probabilistic function associated to the rules.

The probabilistic structure supervise both what facts will happen and the outcomes of the facts that happened.

They determine if a frog will go hunting or mating, and also what will be the outcome of the mating action. Rules can apply to a single individual (e.g. its survival) or model the interaction between two (or more) individuals (e.g. reproduction).
Probabilistic functions give rates from the state of the membrane, rates are transformed into probabilities by normalization, the probabilities determine how rules are selected to construct a maximal multiset for the computational step.
Different rates associated to the rules model how much one event is likely to happen with respect to another. For example a female can give a birth to one, two or three offspring. For each of these outcomes a rule is present with a rate empirically observed on the field. In this way we can model the outcomes of phenomena like: one youngling survive to a parasite, a new member is accepted in the pack, a genetic trait is passed down to new generation, a specific mutation is inherited by the offspring and so on.

Control objects use the idea of promoters to split the computational steps of the model into different phases. For each phase the control objects enable the associated subset of rules. For example in our case study we use the control objects REPR and SEL to switch between the subset of rules model the reproduction phase and the subset of rules model the survival and aging of the specimen.
Any phase has then associated one (or more) control object which enables the correct subset of rules associated to that phase. The subset of rules must also include rules to manage the control object and switch the current one into the correct next one.
To any subset of rules associated to a phase we add the rule to switch the control object into the next one.

For example the presence of control object SEL enables the subset of rules which models the selection of the population of frogs. For any species of frog we have two rules which model the outcome "the frog survives" and the outcome "the frog dies". We also add to this subset of rules one to switch SEL to REPR "$SEL \rightarrow REPR$".
We can use different control objects to have different and more accurate subset of rules. For example with the control objects "SEL" and "REPR" we can have "Dry_Season" and "Wet_Season". The presence at same time of the control object "SEL" and "Dry_Season" describes a specific subset of rules while "REPR" and "Wet_Season" describe another one. Of course it is responsibility of the creator of the model to avoid that two control objects present at the same time are in conflict using special rules to manage the transition between them.

4.3 A simple example

As an example, to show the features of the formalism, we consider a MPP system representing a reproductive event in a sexual population with XY sex-determination system. In the initial population there are females (f) and two types of males (m_1, m_2). Suppose that females prefer m_1 males with a preference value of 0.7, while they mate with m_2 males with a preference value of 0.3. We consider that the different traits of m_1 and m_2 are coded on the Y sexual chromosome. Thus the males in the offspring produced by m_1 and m_2 males are of kind m_1 and m_2, respectively. We consider also that each mating generates a single juvenile. The actual matings, in addition to female preferences depend on the availability of the two kinds of male.

The MPP system representing the described reproductive event is the triple $\langle V_{fm}, w_{0fm}, R_{fm} \rangle$. The alphabet V_{fm} is defined as follows:

$$V_{fm} = \{f, m_1, m_2, f^j, m_1^j, m_2^j\}$$

where the j superscript indicates juveniles.

The reproduction rules are presented in the form:

$$[femaleparent][maleparent] \overset{preference(female,male)}{\rightarrow} [femaleparent][maleparent][offspring]$$

Any of these rules consume as reagents one male, one female and put back as product one male, one female and one offspring to model the happened reproduction of the two parents.
The set of reproduction rules R_{fm} contains the following rules:

$$r_1 :\ f\ m_1 \xrightarrow{f_{m_1}} f\ m_1\ f^j \qquad r_2 :\ f\ m_1 \xrightarrow{f_{m_1}} f\ m_1\ m_1^j$$
$$r_3 :\ f\ m_2 \xrightarrow{f_{m_2}} f\ m_2\ f^j \qquad r_4 :\ f\ m_2 \xrightarrow{f_{m_2}} f\ m_2\ m_2^j$$

where the rates in $R(w_{fm})$ are: $f_{m_1}(w_{fm}) = 0.7 \cdot |w_{fm}|_{m_1} \cdot 0.5$ and $f_{m_2}(w_{fm}) = 0.3 \cdot |w_{fm}|_{m_2} \cdot 0.5$. Note that the result of $f_{m_1}(w_{fm})$ is given by the preference of females for m_1 males multiplied by the number of m_1 males in the population and the probability of producing a male or a female (0.5). $f_{m_2}(w_{fm})$ function is analogous.

Suppose also that the MPP system takes deaths of adult individuals into account. This is represented by the following rules:

$$r_5 :\ m_1 \xrightarrow{0.2} \epsilon \qquad r_6 :\ m_2 \xrightarrow{0.2} \epsilon \qquad r_7 :\ f \xrightarrow{0.2} \epsilon$$

The death rate for each individual is given by the constant 0.2, which is independent from the population consistency.

Given the following initial population:

$$w_{0fm} = m_1\, m_2\, m_2\, f\, f\, f\, f\, f$$

we obtain the following rates: $f_{m_1}(w_{0fm}) = 0.35$ and $f_{m_2}(w_{0fm}) = 0.3$. Note that at least two females cannot find a partner for reproduction in this event because there are not enough males.

A possible evolution step from w_{0fm}, according to the semantics defined in section 4.1, is the following. Rule (step rule) calls $(w_{0fm}, \emptyset) \rightarrow^{+}_{(R(w_{0fm}),w_{0fm})}$ which proceeds by choosing a rule to apply. In the premises of (rule application) the computation of the probabilities of the rules in $R(w_{0fm})$ is performed. This consists in normalizing the rates of the rules, thus obtaining a probability for each rule as follows.

$$r_1 :\ f\ m_1 \xrightarrow{0.184} f\ m_1\ f^j \qquad r_2 :\ f\ m_1 \xrightarrow{0.184} f\ m_1\ m_1^j$$
$$r_3 :\ f\ m_2 \xrightarrow{0.158} f\ m_2\ f^j \qquad r_4 :\ f\ m_2 \xrightarrow{0.158} f\ m_2\ m_2^j$$
$$r_5 :\ m_1 \xrightarrow{0.105} \epsilon \qquad r_6 :\ m_2 \xrightarrow{0.105} \epsilon \qquad r_7 :\ f \xrightarrow{0.105} \epsilon$$

The computed probabilities are then used, together with the applied rule, as labels of the transition described by (rule application). Suppose that rule r_1 is chosen, then the system will make the (internal) transition $(w_{0fm}, \emptyset) \xrightarrow{r_1,0.184}_{(R(w_{0fm}),w_{0fm})} (w_{0fm} - \{m_1, f\}, \{m_1, f, f^j\})$

For the second transition the probabilities of the rules must be updated (premises of (rule application)) because r_1, r_2 and r_5 are no longer applicable (the unique m_1 male was already used). The probabilities of the applicable rules become:

$$r_3 :\ f\ m_2 \xrightarrow{0.3} f\ m_2\ f^j \qquad r_4 :\ f\ m_2 \xrightarrow{0.3} f\ m_2\ m_2^j$$
$$r_6 :\ m_2 \xrightarrow{0.2} \epsilon \qquad r_7 :\ f \xrightarrow{0.2} \epsilon$$

We suppose to choose first rule r_6 and then r_4. Thiese two rules use all the males elements in the initial states. At this point only the rule for female death, r_7, is applicable (with probability equal to 1) and, because of maximal parallelism, it must be selected and applied three times.

With the above choice of the sequence of applicable rules, the (step rule) produces:

$$w_{0fm} \overset{\overline{r},0.011}{\longmapsto}_R\ m_1\, m_2\, f\, f\, f^j\, m_2^j$$

where $\overline{r} = [r_1, r_6, r_4, r_7, r_7, r_7]$, and 0.01104 is the product of the probabilities of the applied rules.

4.4 A case study for the MPP systems: Population dynamics of Lessonae-Esculentus complexes

Lake frog (*Pelophylax ridibundus* Pallas, 1771) and pool frog (*Pelophylax lessonae* Camerano, 1882) can mate producing the hybrid edible frog (*Pelophylax esculentus* Linneus, 1758). The edible frog can coexist with one or both of the parental species giving rise to mixed populations.

	LL	LR
L_yL	L_yL LL	L_yR LR
L_yR	LR	RR not viable

Table 4.1: Reproductive pattern of water frogs

Usually the genotypes of *P. ridibundus*, *P. lessonae* and *P. esculentus* are indicated by RR, LL, and LR, respectively. In Europe there are mainly mixed populations containing *P. lessonae* and *P. esculentus*, called L-E systems. Hybrids in these populations reproduce in a particular way, called *hybridogenesis* [108]. Hybridogenesis consists in a particular gametogenetic process in which the hybrids exclude one of their parental genomes premeiotically, and transmit the other one, clonally, to eggs and sperm.
This particular way of reproduction requires that hybrids live sympatrically with the parental species the genome of which is eliminated. In this way hybrids in a L-E system eliminate the L genome thus producing *P. esculentus* when mating with *P. lessonae*, and generating *P. ridibundus* when mating with other hybrids.
Usually *P. ridibundus* generated in L-E complexes are inviable due to deleterious mutations accumulated in the clonally transmitted R genome [107, 106, 105]. Because of inviability of *P. esculentus* $\times$ *P. esculentus* offspring, edible frog populations cannot survive alone, but they must act as a sexual parasite of one of the parental species. In L-E complexes the reproductive pattern is the one in Table 4.1 where the subscript Y indicates the male sexual chromosome.

Note that the Y chromosome, determining the sex of frog males, can occur only in the L genome, due to primary hybridization which involved, for size constraints, *P. lessonae* males and *P. ridibundus* females. Table 4.1 shows that only one of the three possible matings resulting in viable offspring produce LL genotypes. This would give an advantage to edible frogs which could outnumber *P. lessonae* and eventually eliminate them. This situation would result in an extinction also of *P. esculentus* which cannot survive without the parental species. In addition to their relative abundance which can be promoted by the above reproductive pattern, edible frogs show, by heterosis, a greater fitness than the parental species [104, 103, 102]. The sum of relative abundance and heterosis should outcompete *P. lessonae* in L-E complexes.
The widespread distribution of L-E complexes reveals the existence of mechanisms which contribute to the stability of such complexes, namely the ability of such populations to self-maintain their structure.
Among such mechanisms sexual selection seems to be one of the most important: *P. esculentus* females prefer *P. lessonae* males with respect to males of their own species [101, 100, 99, 98, 97]. Many mathematical and computational models were devoted to the study of the influence of sexual selection in the evolution of populations, the models in [95, 94] show how female preference is able to stabilize L-E complexes by counterbalancing both heterosis and reproductive advantage of edible frogs.

In this case study we are interested in modelling and simulating the dynamics of L-E complexes. Such dynamic is also studied in [96] by means of a computational model programmed in C, where also the genesis of L-E complexes is considered.
The study of conditions for population stability and of possible threats to endangered species is of particular importance for the maintenance of biodiversity.

In this section we use MMP systems for studying the dynamics of European water frog

populations. In particular, we show that female preferences and the inviability of *P. ridibundus* offspring can stabilize L-E complexes. Moreover, we show how the introduction of translocated *P. ridibundus* in stable L-E complexes can lead to the collapse of the systems.

4.4.1 Lessonae -Esculentus complexes: the MPP systems model

We model an L-E complex by means of an MPP system $\langle \Sigma_{LE}, w_{0LE}, R_{LE} \rangle$ in which each individual of the population is represented by an object in the state of the system. Hence, the alphabet Σ_{LE} contains one object for each possible genotype of an individual. We use different objects for juveniles (immature individuals) and adults. Moreover, the alphabet includes some control objects used to realize alternation of reproduction and selection stages. As a consequence, we define $\Sigma_{LE} = \Sigma_{LEa} \cup \Sigma_{LEj} \cup \Sigma_{ctrl}$, where Σ_{LEa} represents adults, Σ_{LEj} represents juveniles and Σ_{ctrl} are control objects.

Since the R genome may contain a deleterious mutation or not, we use different objects for representing *P. esculentus* and *P. ridibundus* individuals carrying or not a mutation in their genotype. Thus, the alphabet representing adults is

$$\Sigma_{LEa} = \{\, LL\,,\, L_yL\,,\, LR_*\,,\, L_yR_*\,,\, LR_\circ\,,\, L_yR_\circ\,,\, R_*R_\circ\,,\, R_\circ R_\circ\}$$

where y represents the Y chromosome, and $*$ and $\circ$ respectively represent the presence and the absence of a deleterious mutation. Note that according to the reproductive pattern of L-E complexes in Table 4.1 males with RR genotypes cannot be produced in a L-E complex. Moreover, note that object $R_\circ R_*$ is not present in V_{LEa} since the individual it represents is indistinguishable from the one represented by $R_*R_\circ$, and hence we use only one object to represent it.

The alphabet representing juveniles is

$$\Sigma_{LEj} = \{\, LL^j\,,\, L_yL^j\,,\, LR_*^j\,,\, L_yR_*^j\,,\, LR_\circ^j\,,\, L_yR_\circ^j\,,\, R_*R_*^j\,,\, R_*R_\circ^j\,,\, R_\circ R_\circ^j\,\}$$

where j denotes that the individual is a juvenile, and the other notations are as before. Note that $R_*R_*^j$ is allowed although it represents non viable genotype since in our model individuals with such a genotype will be allowed to be born, but they will not be allowed to become adults.

Finally, the alphabet of control objects is

$$\Sigma_{ctrl} = \{0, \ldots, n\} \cup \{\, REPR\,,\, SEL\,\}$$

where $REPR$ and SEL represent reproduction and selection stages, respectively, and natural numbers will be used as objects regulating the alternance of the two considered stages, where n is the largest number of iteration a stage of the model will take, for example if the computation alternates three reproduction stages and one selection stage $n = 3$ and $\Sigma_{ctrl} = \{0, 1, 2, 3\} \cup \{\, REPR\,,\, SEL\,\}$.

The set of evolution rules R_{LE} contains reproduction, selection and control rules. Hence, we have $R_{LE} = R_{LEr} \cup R_{LEs} \cup R_{ctrl}$.

Reproduction rules R_{LEr} are of the following form:

$$x\ y \xrightarrow{f_{xy}} x\ y\ z\ |_{REPR}$$

where $x \in \Sigma_{LEa}$ is any object representing a female and $y \in \Sigma_{LEa}$ is any object representing a male. Function f_{xy} gives the rate of mating of females of type x with males of type y by taking into account the sexual preferences of x females and the quantities of individuals of types x and y. In particular, given a multiset of objects w (a system state), we have

$$f_{xy}(w) = k_{mate}(x, y) \cdot |w|_x \cdot |w|_y \cdot 1/k_{o\_kind}(x, y)$$

where $k_{mate}(x, y)$ is the preference of a female x for a male y, and $k_{o\_kind}(x, y)$ is the number of possible kinds of offspring that can be generated by the mating of x with y. Please notice that $1/k_{o\_kind}(x, y)$ distributes the rate of the mating event of x and y over the rules for this mating.

Finally, $z \in \Sigma_{LEj}$ is an object representing the newborn, and it is related with x and y as described in Table 4.1. For example, for $x = LL$ and $y = L_yL$ there are two rules, one with $z = L_yL^j$ and the other with $z = LL^j$. On the other hand, for $x = LR_*$ and $y = L_yR_\circ$ there is one single rule with $z = R_*R_\circ^j$. As a consequence, $k_{o\_kind}(LL, L_yL) = 2$ whereas $k_{o\_kind}(LR_*, L_yR_\circ) = 1$. The other combinations of x and y are analogous.

Regarding selection rules R_{LEs}, they contain two rules for each individual of the population describing its survival and its death during the selection stage, respectively. The presence of these two rules for each type of individual, together with maximal parallelism, ensure that during a selection stage each individual will be faced with one of the two fates.

Survival and death rules are of the forms that follow. For each object $x \in \Sigma_{LEa}$ representing an adult individual we have:

$$x \xrightarrow{g_x} x\ |_{SEL} \qquad x \xrightarrow{g'_x} \epsilon\ |_{SEL}$$

where ϵ represent the empty multiset and g_x and g'_x give the probability of survival and death, respectively, of an individual of type x. Function g_x takes into account the size of the population, the carrying capacity of the environment and the fitness of the individual. More precisely, given $w \in \Sigma^*$ representing a system state, parameter cc representing the carrying capacity of the environment and parameter $k_{fit}(x)$ representing the fitness of individuals of type x, we have:

$$g_x(w) = \frac{1}{\sigma + \frac{|w|}{k_{fit}(x) \cdot cc}}$$

Function g'_x is such that for all $w \in V^*$ it holds $g'_x(w) = 1 - g_x(w)$.

For each object $x^j \in (\Sigma_{LEj} \setminus \{R_*R_*^j\})$ representing a juvenile (but not $R_*R_*^j$) we have:

$$x^j \xrightarrow{g_{x^j}} x\ |_{SEL} \qquad x^j \xrightarrow{g'_{x^j}} \epsilon\ |_{SEL}$$

where $x \in \Sigma_{LE}$ is the object representing the adult of the same type of x^j, and ϵ, g_{x^j} and g'_{x^j} are as before. In the case of $R_*R_*^j$ we consider only the death rule, since such a kind of juvenile is considered too unfit to be able to grow up. Hence, we have only

$$R_*R_*^j \xrightarrow{f_1} \epsilon\ |_{SEL}$$

where for all $w \in \Sigma^*$ it holds $f_1(w) = 1$.

Finally, regarding control rules R_{ctrl}, they are responsible for the appearance and disappearance of objects $REPR$ and SEL in order to activate alternatively reproduction and selection rules.

For the sake of simplicity, we assume that the offspring of each female in each reproductive stage are exactly n. We also assume that each of the offspring is the result of a different mating (this is a very rough simplification that however should not change significantly the global population dynamics). Hence, the object $REPR$ has to be present for n subsequent steps, then it has to be replaced by SEL for one step, and these $n + 1$ steps should be iterated forever. This result is obtained by ensuring that $REPR$ is in the initial state of the system and by using the following control rules:

$$1 \xrightarrow{f_1} 2 \qquad 2 \xrightarrow{f_1} 3 \qquad \cdots \qquad (n-1) \xrightarrow{f_1} n$$

	LL	LR	RR
L_yL	L_yL LL	L_yR L_R	L_yR L_R
L_yR	LR	RR	RR

Table 4.2: Reproductive pattern of water frogs without deleterious mutations.

$$n\ REPR \xrightarrow{f_1} SEL \qquad SEL \xrightarrow{f_1} 1\ REPR$$

where, as before, for all $w \in \Sigma^*$ it holds $f_1(w) = 1$.

The initial state w_{0LE} of the MPP system will change in different simulations. In general, it will contain the control objects 1 and $REPR$, and one object for each individual present in the considered initial population.

4.4.2 Parameters of the model

In order to perform simulations we consider the following initial parameters (some of them will be changed later on). Parameters are a very important element in the description of the model:

- No sexual preference: for every female x and male y we have $k_{mate}(x, y) = 1$.
- 10% higher fitness for hybrids (heterosis effect), namely

$$k_{fit}(x) = \begin{cases} 0.55 & \text{if } x \in \{LyR_*,\ LyR_\circ,\ LyR_*,\ LyR_\circ\}\ (\textit{hybrid specimen adults}) \\ 0.5 & \text{if } x \in \{LL,\ LyL,\ R_\circ R_\circ,\ R_*R_\circ\}\ (\textit{parental specimen adults}) \\ 0.88 & \text{if } x \in \{LyR_*^j,\ LyR_\circ^j,\ LyR_*^j,\ LyR_\circ^j\}\ (\textit{hybrid specimen juveniles}) \\ 0.8 & \text{if } x \in \{LL^j,\ LyL^j,\ R_\circ R_\circ^j,\ R_*R_\circ^j\}\ (\textit{parental specimen juveniles}) \end{cases}$$

- The carrying capacity cc is set to 400.
- The number of reproduction stages n is set to 3.

4.4.3 Results

We study the stability of L-E complexes by considering populations without deleterious mutations in the R genome of *P. esculentus*. We performed 1000 simulations with initial populations composed by *P. lessonae* frogs and a share of 10% of mutation-free edible frogs. The initial state of the system is hence described by the multiset w_{0LE} consisting of 90 instances of LL, 90 of L_yL, 10 of $LR_\circ$, 10 of $L_yR_\circ$ and of the control objects 1 and $REPR$.

We observe that, in all the simulations, the population evolves towards a mono-specific population of viable all-females *P. ridibundus* which eventually collapses for the absence of males (recall that the Y chromosome can occur only on the L genome). Figure 4.1 shows the outcome of a typical simulation. If viable *P. ridibundus* females are produced, the reproductive pattern becomes the one depicted in Table 4.2. Edible frogs are numerically advantaged from possible mating between *P. ridibundus* females and *P. lessonae* males. It is clear from the table that this reproductive pattern generates a numerical disadvantage for pool frogs, the population of which decreases. The decrease in the *P. lessonae* population

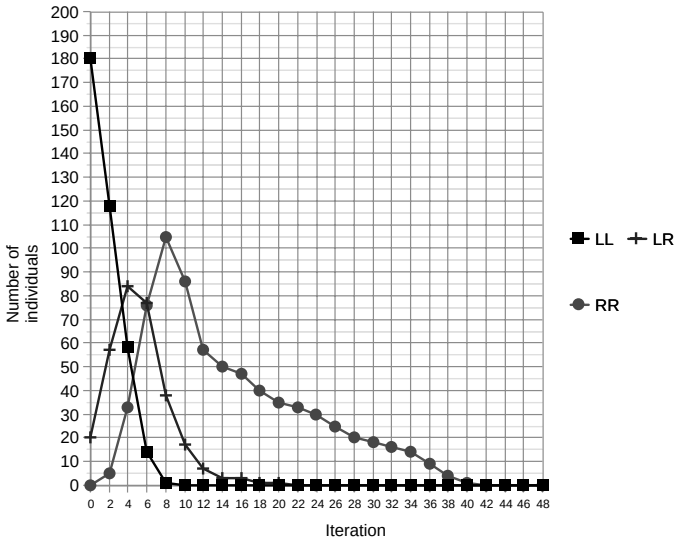


Figure 4.1: Result of a simulation of a L-E complex without deleterious mutations.

has, as a consequence, a decrease of produced L gametes, which, in turn, results in a bigger production of lake frogs. Thus the population of *P. ridibundus* females grows and eventually they out-compete the other species.

Let us now consider an initial population with the same percentages of edible frogs (10%), but in which all the *P. esculentus* individuals carry the deleterious mutations on the R genome, so that *P. ridibundus* females are not viable and they do not appear in the population. The initial state of the system is hence described by the multiset w_{0LE} consisting of 90 instances of LL, 90 of L_yL, 10 of $LR_\circ$, 10 of L_yR_* and of the control objects 1 and $REPR$.

We performed 1000 simulations. We observe that also in this case the population collapses in all simulations. The cause of the collapse is due to the fact that both the reproductive pattern of Table 4.1 and the greater fitness of edible frogs give an advantage to *P. esculentus* frogs forcing the complex towards a mono-specific population. A population with *P. esculantus* alone cannot survive. Figure 4.2 shows the outcome of a typical simulation in this case.

Finally we introduce in the population a female preference towards L_yL males (observed experimentally in [101, 97, 99]). In particular, we set $k_{mate}(LL, L_yL) = 6$ and $k_{mate}(LR, L_yL) = 2$. Also in this case we performed 1000 simulations with the same initial state as before.

We observe that in all simulations the complex evolves towards a stable L-E complex. Figure 4.3 shows the outcome of a typical simulation in this case.

Note that we do not show the outcome of simulations in a population with female preferences but without deleterious mutations in the R genome. Actually in this case the population also evolves towards a all-females *P. ridibundus* population.

4.4.4 Invasion of translocated *P. ridibundus*

The main point that we study with our model is the consequence of the introduction of *P. ridibundus* in stable L-E complexes. *P. ridibundus* can mate both with *P. esculentus*, producing *P. ridibundus*, and with *P. lessonae* (primary hybridization), producing *P. esculentus*.

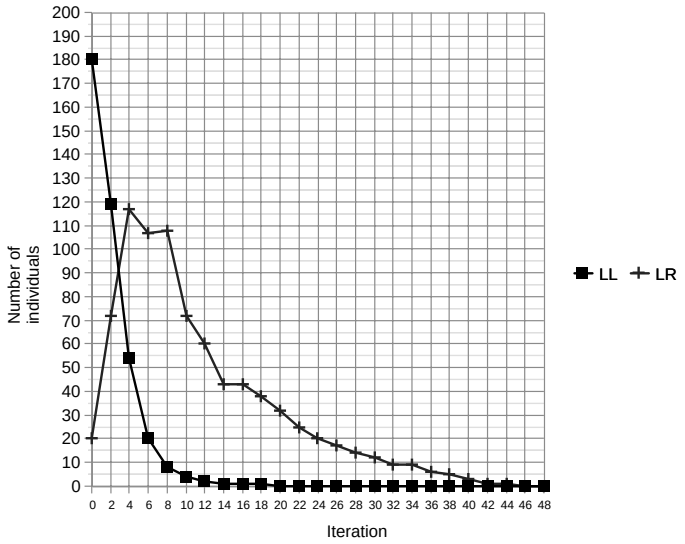


Figure 4.2: Result of a simulation of a L-E complex with deleterious mutations.

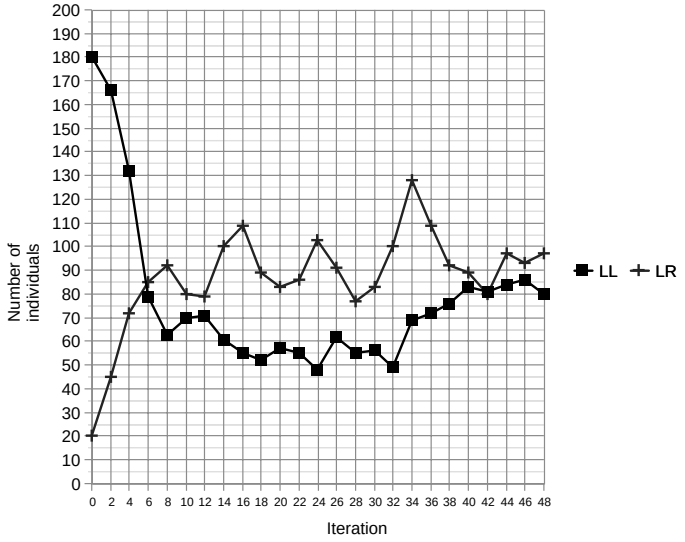


Figure 4.3: Result of a simulation of a L-E complex with deleterious mutations and sexual selection.

4.4.5 The MPP systems model of invasion.

In order to study the dynamics of a L-E complex in which *P.ridibundus* can be introduced we need to extend our previous model including objects and rules describing the behaviour of *P.ridibundus* males.

Consequently, we define a MPP system $\langle \Sigma_{LER}, w_{0LER}, R_{LER} \rangle$ where $\Sigma_{LER} = \Sigma_{LERa} \cup \Sigma_{LERj} \cup \Sigma_{ctrl}$ and $R_{LER} = R_{LERr} \cup R_{LERs} \cup R_{ctrl}$, where, in turn, we have:

- $\Sigma_{LERa} = \{LL\,,\, L_yL\,,\, LR_*\,,\, L_yR_*\,,\, LR_\circ\,,\, L_yR_\circ\,,\, R_*R_\circ\,,\, R_{y*}R_\circ\,,\, R_{y\circ}R_*\,,\, R_\circ R_\circ\,,\, R_{y\circ}R_\circ\}$

- $\Sigma_{LERj} = \{LL^j\,,\, L_yL^j\,,\, LR_*^j\,,\, L_yR_*^j\,,\, LR_\circ^j\,,\, L_yR_\circ^j\,,\, R_*R_*^j\,,\, R_{y*}R_*^j\,,\, R_*R_\circ^j\,,\, R_{y*}R_\circ^j\,,\, R_{y\circ}R_*^j\,,\, R_\circ R_\circ^j\,,\, R_{y\circ}R_\circ^j\}$

- R_{LERr} and R_{LERs} extend R_{LEr} and R_{LEs}, respectively, with analogous rules for *P.ridibundus* males

- Σ_{ctrl} and R_{ctrl} are as before

Note that, given the impossibility of mating between *P. ridibundus* male with *P. lessonae* females (for size reasons), $L_*R_{y\circ}$ individuals cannot be produced. Note also that in reproduction rules involving *P.ridibundus* males with one mutation, namely $R_*R_{y\circ}$ we have to consider more possibilities for the genotype of the offspring than in the previous cases. Indeed, by means of recombination a male of this type can produce four kinds of gametes: $R_*, R_\circ, R_{y*}$ and $R_{y\circ}$.

4.4.6 Parameters of the invasion

In order to perform simulations we consider the following initial parameters (some of them will be changed later on) that we know, from the previous model, could lead to a stable L-E complex if deleterious mutations are present.

- Sexual preference:

$$k_{mate}(x,y) = \begin{cases} 6 & \text{if } x = LL \text{ and } y = L_yL \\ 2 & \text{if } x \in \{\, LR_*\,,\, LR_\circ\,\} \text{ and } y = L_yL \\ 0 & \text{if } x = LL \text{ and } y \in \{\, R_{y*}R_\circ\ \,,\, R_{y\circ}R_*\ \,,\, R_{y\circ}R_\circ\,\} \\ 1 & \text{otherwise} \end{cases}$$

- 10% higher fitness for hybrids (heterosis effect), namely

$$k_{fit}(x) = \begin{cases} 0.55 & \text{if } x \in \{\, LR_*,\, LR_\circ,\, LyR_*,\, LyR_\circ\,\}\ (\textit{hybrid specimen adults}\,) \\ 0.5 & \text{if } x \in V_{LERa} \setminus \{\, LR_*,\, LR_\circ,\, LyR_*,\, LyR_\circ\,\}\ (\textit{parental specimen adults}\,) \\ 0.88 & \text{if } x \in \{\, LR_*^j,\, LR_\circ^j,\, LyR_*^j,\, LyR_\circ^j\,\}\ (\textit{hybrid specimen juveniles}\,) \\ 0.8 & \text{if } x \in V_{LERj} \setminus \{\, LR_*^j,\, LR_\circ^j,\, LyR_*^j,\, LyR_\circ^j\,\}\ (\textit{parental specimen juveniles}\,) \end{cases}$$

- The carrying capacity cc is set to 400.

- The number of reproduction stages n is set to 3.

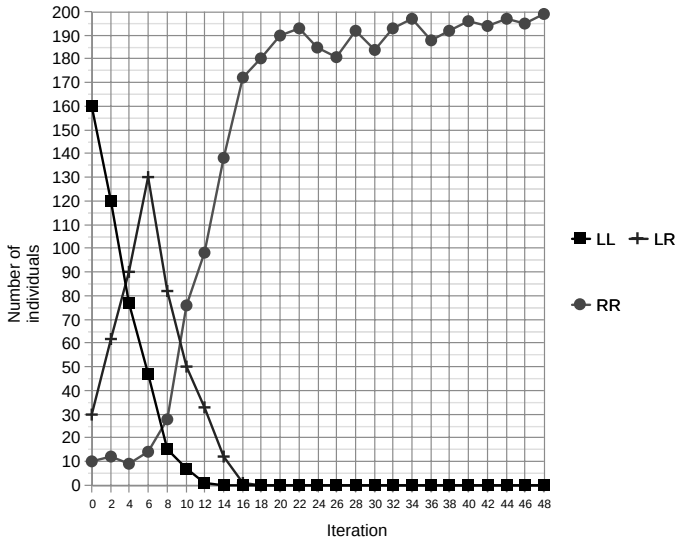


Figure 4.4: Simulation of invasion leading to replacement of the population by lake *P. ridibundus* frogs.

4.4.7 Results of the invasion

We performed 1000 simulations with initial populations composed by 80% of *P. lessonae* frogs, 15% of mutation-free edible frogs and 5% of *P. ridibundus* frogs. The initial state of the system is hence described by the multiset w_{0LER} consisting of 80 instances of LL, 80 of L_yL, 15 of LR_*, 15 of L_yR_*, 5 of $R_\circ R_\circ$, 5 of $R_{y\circ}R_\circ$, and of the control objects 1 and $REPR$.

The results in this case are of two kinds: 73% of simulations result in a mono-specific *P. ridibundus* population while 27% of simulations result in a collapse of the whole population. Figure 4.4 and 4.5 show typical population dynamics. Because the introduce lake frogs are mutation-free and because they can mate with *P. esculentus* frogs, deleterious mutations are gradually purged. Thus, the population evolves towards a mono-specific *P. ridibundus* system. In this situation, if males are present, the *P. ridibundus* population can survive, otherwise it will collapse. The survival of *P. ridibundus* males is threatened by female preferences towards LL males and the advantage of *P. esculentus* for their heterosis. In all cases, the initial L-E complex is destroyed, as predicted in [52].

4.5 Translation of MPP systems into the PRISM input language

The PRISM translation of a MPP system follows the operational semantics of MPP systems defined above. The translation of MPP systems in to the PRISM input language is rather simple, although the obtained PRISM code is quite complex and difficult to read.

Multiset of objects of MPP systems are translated into integer PRISM variables, one for each kind of object in the multiset. The value of each variable represents the number of instances in the multiset of the corresponding kind of object.

In accordance with the semantics of MPP systems, the states of the considered transitions systems require different multisets of objects to be represented. In particular, this in necessary to represent the pairs of multisets corresponding to the states on which the semantic rules (rule application), (single rule sequence) and (multiple rule sequence) operate. Consequently, in the PRISM translation we have to include three different sets of

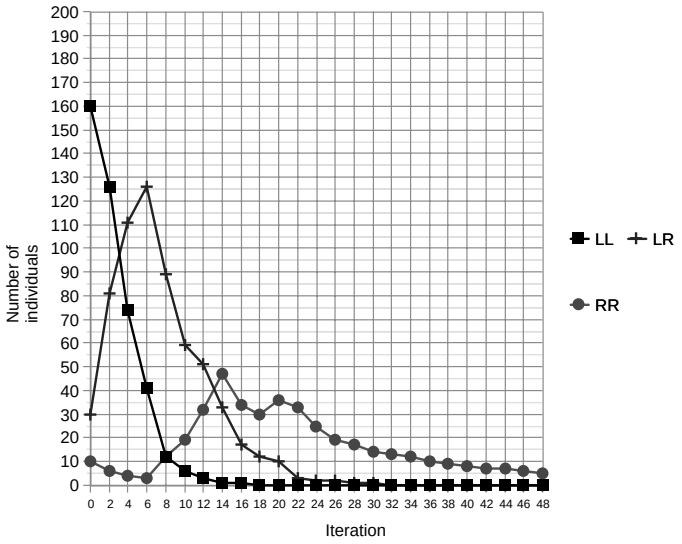


Figure 4.5: Simulation of invasion leading to replacement of the population by lake *P. ridibundus* frogs.

variables: one for representing the state of (step rule) and two for representing the pairs used as states of the other rules. Evolution rules of MPP systems are then translated into transitions in the PRISM language that change the state of the system in accordance with what specified in (rule application). Before showing the formal definition of the translation, we introduce (a subset of) the syntax and semantics of PRISM.

PRISM syntax and semantics

A PRISM model is defined as a tuple $(Var_{\mathrm{P}}, max_{\mathrm{P}}, R_{\mathrm{P}}, init_{\mathrm{P}})$ where Var is a finite set of variable names; $max_{\mathrm{P}} : Var \to \mathbb{N}$ is a function giving, for each variable, the maximum value it can hold; R_{P} is a finite set of *probabilistic rules*; $init : Var \to \mathbb{N}$ is a function giving the initial value of each variable. For the sake of simplicity, only integer variables are allowed, where each variable $v \in Var$ can hold values in the set $\{0, 1, \ldots, max(v)\}$.

Definition 3. *The syntax of the probabilistic rules in R_{P} is defined by the following grammar:*

$$
\begin{aligned}
Rule &::= Cond \to \big(Exp : Update\big) + \cdots + \big(Exp : Update\big) \\
Cond &::= Atom\ Rel\ Atom \mid Cond \vee Cond \mid Cond \wedge Cond \mid \neg Cond \\
Atom &::= Var \mid \mathbb{N} \\
Rel &::= = \mid > \mid < \\
Update &::= \text{true} \mid (Var' = Exp) \mid Update\ \&\ Update \\
Exp &::= Var \mid \mathbb{N}t \mid Exp + Exp \mid Exp - Exp \mid Exp/Exp \mid Expr * Exp \mid Cond\ ?\ Exp : Exp
\end{aligned}
$$

where $Cond$ represents boolean conditions, and Exp represent arithmetic expressions (which also includes the ternary operator $\_\ ?\ \_ : \_$ to denote conditional expressions). A rule $Cond \to \big(Exp : Update\big) + \cdots + \big(Exp : Update\big)$ is composed of a boolean condition defining its applicability, followed by a set of pairs $\big(Exp : Update\big)$ describing probabilistic transitions, where Exp is an expression giving the probability of the transition, and $Update$ describes how the variables are to be updated. In any update $(x'_1 = e_1)\& \ldots \&(x'_n = e_n) \in Update$, each variable x_i is assumed to occur at most once among the left-hand sides of all the updates.

The semantics of a PRISM model $(Var_{\mathrm{P}}, max_{\mathrm{P}}, R_{\mathrm{P}}, init_{\mathrm{P}})$ is defined as a Discrete Time Markov Chain (DTMC) where states are functions $\sigma : Var_{\mathrm{P}} \to \mathbb{N}$ such that $\forall a \in Var_{\mathrm{P}}.\, \sigma(a) \in \{0, \ldots, max(a)\}$. Given a condition $C \in Cond$, we denote as $C\sigma$ the truth value obtained from the evaluation of the condition in state σ, while, given an expression $e \in Exp$, we denote as $e\sigma$ the real value obtained from its evaluation in state σ.

We assume that the set of rules R_{P} is such that the conditions of all rules do not overlap, namely:

$$\forall r_1, r_2, \sigma.\ C(r_1)\sigma \wedge C(r_2)\sigma = \text{false}.$$

where $C(r_i)$ denotes the condition for the application of rule r_i, $i \in \{1, 2\}$. Moreover, if a rule $r = C \to \sum_{i=1}^{n}(e_i : upd_i)$ is enabled in a state σ, then the sum of the probabilities of all cases must be equal to 1, namely:

$$\forall \sigma.\ (C\sigma = true) \implies \sum_{i=1}^{n} e_i\sigma = 1.$$

Before defining the probability of transition between states, we introduce the following transition relation describing the application of a rule:

$$\frac{r_k = C \to \sum_{i=1}^{n}(e_i : upd_i) \quad C\sigma = true \quad \exists j.\ e_j\sigma = p}{\sigma \xrightarrow{k,j,p}_{\mathrm{r}} \sigma \lhd upd_j}$$

where the application of upd to σ, $\sigma \lhd upd$, is defined as follows:

$$\sigma \lhd \text{true} = \sigma;$$
$$\sigma \lhd ((x' = e)\,\&\,upd) = (\sigma \lhd upd)[e\sigma/x].$$

Finally, the probabilistic transitions of the DTMC are derived as follows:

$$\frac{p = \sum\{p' \mid \sigma \xrightarrow{k,j,p}_{\mathrm{r}} \sigma'\}}{\sigma \xrightarrow{p} \sigma'} \qquad \frac{\sigma \not\to_{\mathrm{r}}}{\sigma \xrightarrow{1} \sigma}$$

The first inference rule allows deriving transitions corresponding to the application of one or more rules, whose probability is given by the sum of the probabilities to pass from σ to σ' by means of the application of that rules. The second inference rule states that, if no rules are applicable, the probability to remain in the same state is 1. Recall that at most one rule is ever enabled in a state, and the sum of the probabilities of all the cases of a rule must sum up to 1. Thus, it is easy to see that, according to the above inference rules, the sum of the probabilities of all the transitions exiting from a state σ is equal to 1, as required for DTMCs.

4.5.1 Translation

Let $\langle V, w_0, R\rangle$ be an MPP system where $R = \{r_1 : u_1 \xrightarrow{f_1} v_1|_{pr_1}, \ldots, r_s : u_s \xrightarrow{f_s} v_s|_{pr_s}\}$. Since PRISM models are finite, we assume a function $max : V \to \mathbb{N}$ giving the maximum number of occurrences of each symbol. The translation gives a PRISM model $(Var_{\mathrm{P}}, max_{\mathrm{P}}, R_{\mathrm{P}}, init_{\mathrm{P}})$ where:

- $Var_{\mathrm{P}} = \{a_0, a_1, a_2 \mid a \in V\}$;
- $\forall a \in V.\ max_{\mathrm{P}}(a_0) = max_{\mathrm{P}}(a_1) = max_{\mathrm{P}}(a_2) = max(a)$;
- $\forall a \in V.\ init_{\mathrm{P}}(a_0) = |w_0|_a$, $init_{\mathrm{P}}(a_1) = |w_0|_a$, $init_{\mathrm{P}}(a_2) = 0$.

The translation uses, for each symbol $a \in V$, three variables a_0, a_1, a_2 to describe, respectively, the (i) the multiset w of symbols at the beginning of the step, (ii) the multiset w' of symbols not yet consumed in the current step, and (ii) the multiset $\overline{w}'$ of products created by the rules being applied. We assume that, for each evolution rule $r_k \in R$, the rate function f_k has an equivalent definition $\tilde{f}_k$ defined as a PRISM expression over the variables $\{a_0 \mid a \in V\}$. Finally, the set of rules R_{P} is composed of the following two rules:

$$rp_1: \quad \left(\bigvee_{t=1}^{s} cond(u_t, pr_t)\right) \to \sum_{k=1}^{s} \left(\frac{cond(u_k, pr_k)?\tilde{f}_k : 0}{\sum_{z=1}^{s} cond(u_z, pr_z)?\tilde{f}_z : 0} : update(u_k, v_k)\right);$$

$$rp_2: \neg\left(\bigvee_{t=1}^{s} cond(u_t, pr_t)\right) \to 1 : \mathop{\&}_{a \in V} \begin{pmatrix} a_0' = a_1 + a_2 \leq max_{\mathrm{P}}(a_0)?a_1 + a_2 : max_{\mathrm{P}}(a_0)) \ \& \\ a_1' = a_1 + a_2 \leq max_{\mathrm{P}}(a_1)?a_1 + a_2 : max_{\mathrm{P}}(a_1)) \ \& \\ a_2' = 0 \end{pmatrix};$$

where

$$cond(u, pr) = \left(\bigwedge_{a \in u} a_1 \geq |u|_a\right) \wedge \left(\bigwedge_{a \in pr} a_0 \geq |pr|_a\right)$$

$$update(u, v) = \left(\mathop{\&}_{a \in u} a_1' = a_1 - |u|_a\right) \ \& \ \left(\mathop{\&}_{a \in v} a_2' = a_2 + |v|_a \leq max_{\mathrm{P}}(a_2)?a_2 + |v|_a : max_{\mathrm{P}}(a_2)\right)$$

The rules rp_1, rp_2 are mutually exclusive, since their conditions are complementary. Intuitively, rule rp_1 is used to model the application of a single evolution rule as described by (rule application) from the semantics of the MPP systems, while rule rp_2 models the actual maximally-parallel step as described by (step rule). In particular, $cond(u, pr)$ encodes the condition needed for the application of an MPP rule with reactants u and promoters pr, while $update(u, v)$ describes how the variables a_1, a_2 are updated to reflect the application of a rule having reactants u and products v. Note that, in both rp_1 and rp_2, the variables cannot exceed their upper bounds.

Lemma 1. *Let $P = \langle V, w_0, R\rangle$ be an MPP system, where $R = \{r_1, \ldots, r_s\}$ with $r_i = u_i \xrightarrow{f_i} v_i|_{pr_i}$, and let $max : V \to \mathbb{N}$ be a function. Let $M_{\mathrm{P}} = \langle Var_{\mathrm{P}}, max_{\mathrm{P}}, R_{\mathrm{P}}, init_{\mathrm{P}}\rangle$ be the translation of P. Let $w, w', \overline{w}' \in V^*$ be such that $\forall a \in V$. ($|w|_a < max(a)$, $|w'|_a < max(a)$, $|\overline{w}'|_a < max(a)$), and $\sigma : Var_{\mathrm{P}} \to \mathbb{N}$ such that $\forall a \in V$. $\sigma(a_0) = |w|_a$, $\sigma(a_1) = |w'|_a$, $\sigma(a_2) = |\overline{w}'|_a$. We have*

$$\begin{array}{ll} (w', \overline{w}') \xrightarrow{r_i, p}_{(R,w)} (w'', \overline{w}'') & \text{such that } \forall a \in V.\ |w''|_a < max(a) \wedge |\overline{w}''|_a < max(a) \\ \iff & \\ \sigma \xrightarrow{1, i, p} \sigma' & \text{such that } \forall a \in V.\ \sigma'(a_1) < max_{\mathrm{P}}(a_1) \wedge \sigma'(a_2) < max_{\mathrm{P}}(a_2) \end{array}$$

where $\sigma'(a_1) = |w''|_a$ and $\sigma'(a_2) = |\overline{w}''|_a$, for all $a \in V$.

Proof. **Case** $\Longrightarrow$. From the semantics of MPP systems, (rule application) is applied to derive the transition describing the application of rule r_i. From rule application we know that $u_i \subseteq w'$, $pr_i \subseteq w$, and $p = \frac{k}{\sum_{k' \in K} k'}$ where $K = \{\!|k' | u' \xrightarrow{k'} v' \ |_{pr'} \in R,\ u' \subseteq w',\ pr' \subseteq w|\!\}$. This implies that $cond(u_i, pr_i)\sigma$ is true and that

$$e_i\sigma = \left(\frac{cond(u_i, pr_i)?\tilde{f}_i : 0}{\sum_{z=1}^{s} cond(u_z, pr_z)?\tilde{f}_z : 0}\right)\sigma = \left(\frac{\tilde{f}_i}{\sum_{z=1}^{s} cond(u_z, pr_z)?\tilde{f}_z : 0}\right)\sigma.$$

Let $J = \{j \mid r_j \in R \wedge cond(u_j, pr_j)\sigma = \text{true}\}$, we have $e_i\sigma = \tilde{f}_i\sigma / (\sum_{j \in J} \tilde{f}_j\sigma)$. Since $j \in J$ iff r_j is applicable, we have $\tilde{f}_i\sigma = k$ and $\sum_{j \in J} \tilde{f}_j\sigma = \sum_{k' \in K} k'$. Thus $e_i\sigma = p$.

It is easy to see that $\sigma'(a_1) < max_{\mathrm{P}}(a_1)$ and $\sigma'(a_2) < max_{\mathrm{P}}(a_2)$, then $\sigma'(a_1) = |w''|_a$, $\sigma'(a_2) = |\overline{w}''|_a$.

Case $\Longleftarrow$. Suppose $\sigma \xrightarrow{1,i,p} \sigma'$ such that $\forall a_1, a_2 \in Var_{\mathrm{P}}.\ \sigma'(a_1) < max_{\mathrm{P}}(a_1) \wedge \sigma'(a_2) < max_{\mathrm{P}}(a_2)$. This means that $cond(u_i, pr_i)$ is true, and

$$p = \left(\frac{\tilde{f}_i}{\sum_{z=1}^{s} cond(u_z, pr_z)?\tilde{f}_z : 0} \right) \sigma.$$

Analogously to the previous case, $p = e_i\sigma = \tilde{f}_i\sigma / (\sum_{j \in J} \tilde{f}_j\sigma)$, where $J = \{j \mid r_j \in R \wedge cond(u_j, pr_j)\sigma = \text{true}\}$. Thus, $p = \frac{k}{\sum_{k' \in K} k'}$ where $K = \{|k' | u' \xrightarrow{k'} v' \ |_{pr'} \in R,\ u' \subseteq w',\ pr' \subseteq w|\}$. Moreover, because $\sigma'(a_1) < max_{\mathrm{P}}(a_1)$ and $\sigma'(a_2) < max_{\mathrm{P}}(a_2)$, for all $a_1, a_2 \in Var_{\mathrm{P}}$, we have each expression $(a_2' = a_2 + |v|_a \leq max_{\mathrm{P}}(a_2)?a_2 + |v|_a : max_{\mathrm{P}}(a_2))$ is equivalent to $a_2 + |v|_a$. Then $|w''|_a < max(a)$, $|\overline{w}''|_a < max(a)$, $\sigma'(a_1) = |w''|_a$, and $\sigma'(a_2) = |\overline{w}''|_a$, for all $a \in V$. □

Theorem 1. *Let $P = \langle V, w_0, R \rangle$ be an MPP system, and let $max : V \to \mathbb{N}$ be a function. Let $M_{\mathrm{P}} = \langle Var_{\mathrm{P}}, max_{\mathrm{P}}, R_{\mathrm{P}}, init_{\mathrm{P}} \rangle$ be the translation of P. Let $w \in V^*$ be such that $\forall a \in V.\ |w|_a < max(a)$, and $\sigma : Var_{\mathrm{P}} \to \mathbb{N}$ such that $\forall a \in V.\ \sigma(a_0) = |w|_a$ and $\sigma(a_1) = 0$ and $\sigma(a_2) = 0$. We have*

$$\begin{aligned}
&(w, \emptyset) \xrightarrow{r_{i_1}, p_1}_{(R(w),w)} (w_1, \overline{w}_1) \xrightarrow{r_{i_2}, p_2}_{(R(w),w)} \cdots \xrightarrow{r_{i_n}, p_n}_{(R(w),w)} (w_n, \overline{w}_n) \\
&\qquad \textit{such that } \forall i \in \{1, \ldots, n\}.\ |w_i|_a < max(a), |\overline{w}_i|_a < max(a), \\
&\qquad \textit{and } (w_n, \overline{w}_n) \not\longrightarrow_{(R(w),w)} \\
&\Longleftrightarrow \\
&\sigma \xrightarrow{1,i_1,p_1} \sigma_1 \xrightarrow{1,i_2,p_2} \cdots \xrightarrow{1,i_n,p_n} \sigma_n \\
&\qquad \textit{such that } \forall i \in \{1, \ldots, n\}.\ \sigma_i(a_1) < max_{\mathrm{P}}(a_1),\ \sigma_i(a_2) < max_{\mathrm{P}}(a_2) \\
&\qquad \textit{and } \sigma_n \xrightarrow{2,1,1} \sigma_f
\end{aligned}$$

where $\forall a \in V.\ \sigma_f(a_0) = \sigma_f(a_1) = |w_n + \overline{w}_n|_a, \sigma_f(a_2) = 0$.

Proof. Follows from Lemma 1 and from the definition of rules rp_1, rp_2. In particular, rp_1 cannot be applied to σ_n because there no transitions from $(w_n, \overline{w}_n)$ in P, while rp_2 is applicable to σ_n because its condition is the negation of the one of rp_1. The relation between σ_f and $(w_n, \overline{w}_n)$ follows from the definition of rp_2. □

As a consequence of Theorem 1, we have that if there is a sequence of (rule application) in P such that for all $a \in V$, $max(a)$ is never reached in any step, then a step $w \overset{\overline{r},\ \overline{p}}{\longmapsto}_R w'$ corresponds, in the PRISM translation of P, to a sequence of applications of rp_1 followed by an application of rp_2. Moreover, the probability $\overline{p}$ is equal to the probability of passing from σ to σ_f through the path $\sigma \xrightarrow{1,i_1,p_1} \sigma_1 \xrightarrow{1,i_2,p_2} \cdots \xrightarrow{1,i_n,p_n} \sigma_n \xrightarrow{2,1,1} \sigma_f$. Consequently, if is possible to derive $w \overset{p}{\Longrightarrow}_R w'$ using the rule (state transition probability) with PR consisting only of transitions $w \overset{\overline{r},\ \overline{p}}{\longmapsto}_R w'$ derived without reaching $max(a)$ in any step, for all $a \in V$, then we have a transition in the DTMC of the translation of P from σ to σ_f with probability p, namely $\sigma \xrightarrow{p} \sigma_f$.

4.5.2 Results of the translation of the frog model

The dynamics of the population in the different cases is studied both by means of standard stochastic simulation with the MPP systems interpreter and by means of statistical model

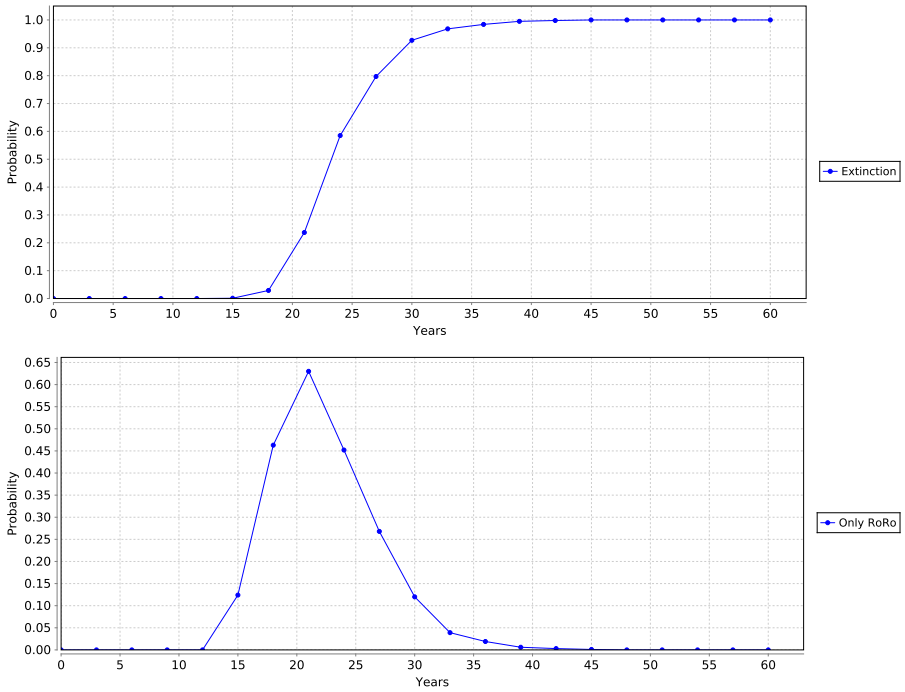


Figure 4.6: Result of a statistical model checking of a L-E complex without deleterious mutations. Probability of extinction (above) and of a population consisting of all *RoRo* individuals (below).

checking with PRISM with the translation we propose in the section above.

In Figure 4.6 we show results of statistical model checking of the same model in the case in which deleterious mutations ae not present (Fig.4.1). In particular, we study the probabilities of two properties. The first is the probability of extinction of the population over time. This is expressed in PRISM by the following PCTL formula, where `years_counter` is a variable, added to the translation of the MPP model, counting the number of elapsed years, and `total_population` is a PRISM formula that corresponds to the sum of all variables representing individuals. The graph in Figure 4.6 is obtained by varying the value of constant `Years` from 0 to 60 by steps of 3.

```
P=? [ F total_population=0 & years_counter<=Years ]
```

The results show that the population become extinct with probability 1.0 in nearly 40 years.

The second probability we studied is the one of having a population composed only of *P. ridibundus* females (`RoRo`) at a given year. This is expressed by the following PCTL formula in which `Years` is varied as before.

```
P=? [ F total_population=RoRo & years_counter=Years ]
```

Obviously, the fate of a population of only *P. ridibundus* females is extinction. Figure 4.6 shows that such a situation has the higher probability around 20 years. From the two graphs in the figure it can be concluded that the extinction of the population in this case has as intermediate step the formation of a population of only *P. ridibundus* females.

Now consider an initial population with the same percentages of edible frogs (10%), but in which all the *P. esculentus* individuals carry the deleterious mutations on the *R*

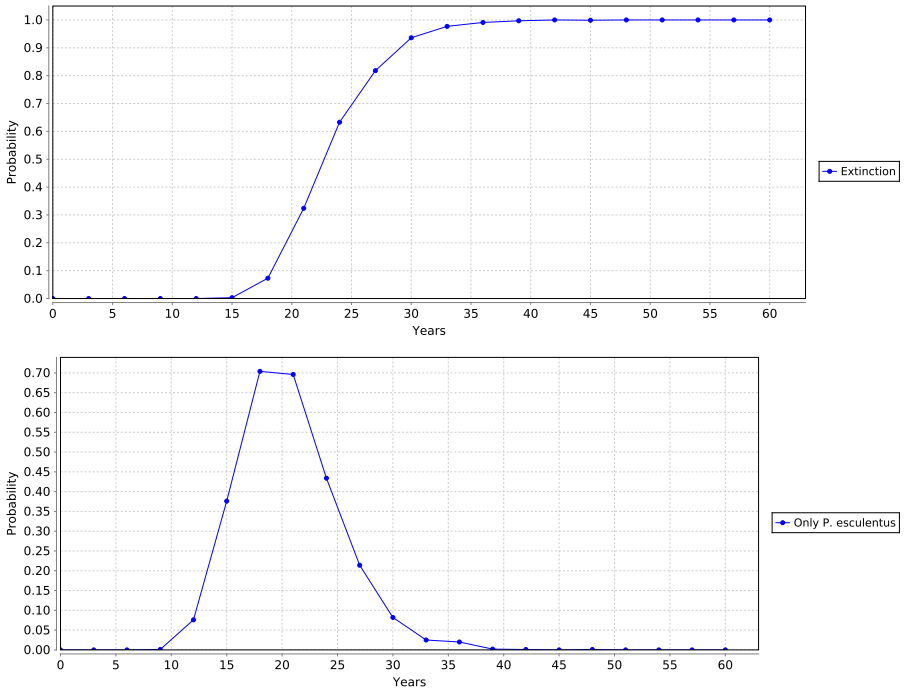


Figure 4.7: Result of a statistical model checking of a L-E complex with deleterious mutations. Probability of extinction (above) and of a population consisting of all P. esculentus individuals (below).

genome (Fig. 4.2, that is *P. ridibundus* females are not viable and they do not appear in the population. In Figure 4.7 we show the results of statistical model checking for this model. As before, we first study the probability of extinction of the population over time (same PCTL formula as before). The results show that also in this case the population become extinct with probability 1.0 in nearly 40 years in according with the MPP previsions.

The second probability we studied in this case is the one of having a population composed only of *P. esculentus* frogs (females `LRm` and males `LyRm`)at a given year. This is expressed by the following PCTL formula in which `Years` is varied as before.

```
P=? [ F total_population=LRm+LyRm & years_counter=Years ]
```

Figure 4.7 shows that a population with only *P. esculentus* individual is reached with high probability after around 20 years. Similarly as before, from the two graphs in the figure it can be concluded that the extinction of the population in this case has as intermediate step the formation of a population of only *P. esculentus* individuals.

When we introduce in the population a female preference towards L_yL males (Fig. 4.3) results of statistical model checking show that the probability of extinction in 60 years (same PCTL formula as before) is equal to 0.01.

Finally when we model the invasion of *P. ridibundus* (Fig. 4.5) results obtained by statistical model checking (Figure 4.8) substantially confirm that the probability of extinction of the population in 60 years (same PCTL formula as before) is nearly 20%. From simulation results we formulated the hypothesis that the extinction of the whole population can be predicted by the early extinction of the *P. ridibundus* males. In order to verify this hypothesis, we computed the probability of the following formula:

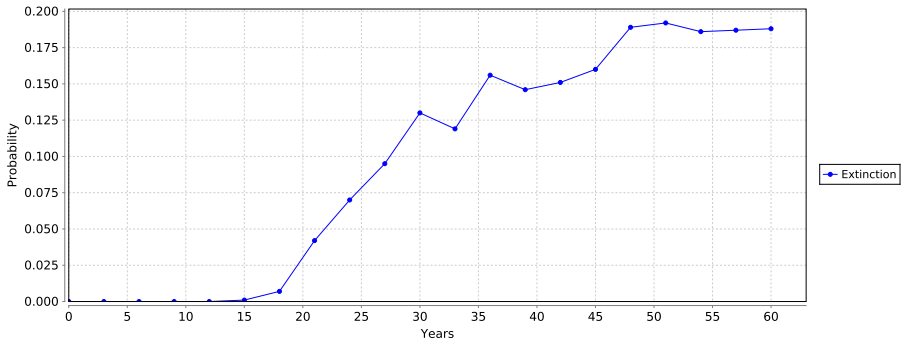


Figure 4.8: Result of a statistical model checking of a L-E complex invaded by a small population of *P. ridibundus* (both males and females): probability of extinction.

```
P=?[F !flag_RR_all_females & total_population=0 & years_counter<=60]
```

In this formula, `flag_RR_all_females` is a boolean variable, added to the PRISM translation of the MPP system, that becomes true if all of the *P. ridibundus* males die within the first 10 years. Hence, the formula gives the probability of extinction of the population when the *P. ridibundus* males survive after the first 10 years. If this probability is very low, than there is a strong correlation between the death of *P. ridibundus* males in the first years and the extinction of the whole population. Actually, the result of statistical model checking of this formula gives 0.04, thus confirming our hypothesis. Moreover, we have checked also the probability of *P. ridibundus* males in the first 10 years, that is:

```
P=?[F flag_RR_all_females]
```

The result we have obtained is nearly 0.16, which agrees with the previous results and is a further confirmation of the hypothesis.

4.6 Conclusion and general overview about MPP systems

In conclusion MPP systems extend P systems with probabilistic rates associated to the rules and with control objects to manage the different phases of the computation.
These two features are enough to allow us to obtain a useful tool, which can describe and model ecological population problems, like the case study presented in this chapter.
The computation rewrites the state of the system. Within that internal state individuals are represented by interchangeable objects, and in each computational step probabilistic functions and rates act together with control objects to select one possible set of rules to be applied.
It's possible to realize a code which puts into practice a model defined by this formalism. We can add some specific output functions to this code, functions which can map, step by step, the state of the system and elaborate useful data.
All these tools together allow models to answer questions like :
- What is the male/female ratio of the population over time?
- How fast the population grows?
- What are a Population viability analysis (PVA) results after 100 steps?
- Does one observed mutation help the spread of a species in a specific habitat?

The raw output of the MPP systems is the sequence of states of the system (i.e multi-sets of elements in the membrane) reached after any computational step. In a real coded

implementation of a MPP systems formalized model we can apply to this output, composed by the sequence of states, appropriate functions in order to produce refined data like the graph showed in the chapter above.

Chapter 5

Attributed Probabilistic P systems (APP systems)

5.1 From MPP to APP systems

We have seen how MPP systems objects represent individuals in populations, while control objects are used as promoters to model subsets of active rules at various stages of computation.

Individuals are often associated with attributes such as gender, age, morph. In order to describe all possible kinds of individual, in MPP systems, it is necessary to have an alphabet which includes one object for each possible combination of attribute values.
For example, let us assume an ecosystem with lions and sheep where individuals are distinguished by age, gender and (only for sheep) presence of a specific DNA mutation.
The MPP System describing such ecosystem should have an alphabet like the following:
$V = \{LionM0, LionM1, \ldots, LionMk, LionF0, LionF1, \ldots, LionFk, SheepM0,$
$SheepM1, \ldots, SheepMw, SheepF0, SheepF1, \ldots, SheepFw, SheepM0*,$
$SheepM1*, \ldots, SheepMw*, SheepF0*, SheepF1*, \ldots, SheepFw*\}$

where M|F denote Male or Female gender, k is the maximum possible age for a Lion in the model, w is the maximum posssible age for a Sheep in the model and * denotes the presence of the mutation in the Sheep DNA.

If attributes could assume too large an array of values, the resulting alphabet would be so big that its management would be impractical. To solve this issue we introduce a record of attributes associated to the object rather than introducing different elements for any possible combination of attributes value.
If, for example, we linked to the item "lion" attributes (age, sex) elements get annotated like this: $Lion_{(3,m)}$ or $Lion_{(8,f)}$ indicating a male lion 3 years old or a female lioness 8 years old.
When the need to associate attributes to the elements arises, it becomes necessary to move from MPP system models to something more complex. This leads to the formulation of APP systems. In APP systems we keep using the promoters and probabilistic functions but at the same time we add attributes to enrich the model.
The attributes of objects consumed by the rules greatly influence the probabilistic functions associated with each rule, therefore generating a much more flexible and powerful probabilistic system.
Attributes assist also in writing aggregate rules, for example, let us assume we have to model Male Sheep which can reach the maximum age of four years old. In order to model the aging, year by year, we will need rules like:

$Sheep_{(0,M,\_)} \rightarrow Sheep_{(1,M,\_)}$
$Sheep_{(1,M,\_)} \rightarrow Sheep_{(2,M,\_)}$
$Sheep_{(2,M,\_)} \rightarrow Sheep_{(3,M,\_)}$
$Sheep_{(3,M,\_)} \rightarrow Sheep_{(4,M,\_)}$
$Sheep_{(4,M,\_)} \rightarrow \_$

The use of attributes to define rules leads to model aging as follows:

$Sheep_{(age<4,M,\_)} \rightarrow Sheep_{(age+1,M,\_)}$
$Sheep_{(4,M,\_)} \rightarrow \_$

Another example can be some protozoans with a single attribute *size*, a protozoan bigger than another one can eat the smaller to increase his size, one rule to describe this behavior can be:

$Protozoan_{m>n}, Protozoan_n \rightarrow Protozoan_{m+n}$

this rule models a protozoan of any size which eats another smaller protozoan for each couple of values m and n with $m > n$.

5.2 Introduction to APP systems

Like MPP systems, APP systems are intended to be used to model the dynamics of populations and ecosystems.

We used APP systems to model social behavior of some primates species as a case study. In particular, as an application we developed a model to compare despotic and egalitarian behaviors in /emphlemur catta.
Social systems of this kind are usually approached by means of agent-based models, which are often poorly documented and ambiguous. Conversely, since both syntax and semantics of APP systems are formally defined, a model based on APP systems is unambiguous.
We plan to adapt our general model to represent the behavior of a large number of specific primates species by changing parameters values, thus showcasing our formalism flexibility.
In this chapter we aim to show how APP systems can be used to describe real ecological systems, and to understand the behavior of self-organizing animal populations, like schools of fishes or flocks of birds, just as other models already proposed do [93, 92]. Modelling social interactions in primates is an interesting research field which has been usually tackled by means of agent-based models [91, 90, 89, 88, 87, 86, 85].
Such models have been often criticized because, in many cases, they were so poorly documented that the models themselves could not be evaluated. For these reasons some protocols have been defined in order to create a standard structure by which all the agent-based models could be documented [84, 83]. Such protocols document a model by providing a check list of questions which must be answered by the model such as:

- Who (i.e. what entity) does what, and in what order?
- When are state variables updated?
- What kind of entities are in the model?
- By what state variables, or attributes, are those entities characterized?

We present as a case study an APP systems model of social interactions in primates

(*lemur catta*). The model is composed of quite a few unambiguous rules. Those rules can be seen as an implementation of the rules used in agent-based models which, however, are usually programmed "ad hoc" and their effect needs to be documented separately. On the contrary, the rules of our APP-system-based model are almost self-explanatory, showing how easy to use are formalisms for modelling real-world systems.
Simulations of our model are presented as well. We show that the obtained results are compatible with the results of agent-based models described in literature.
Nevertheless, the use of APP systems for modelling real systems provides important advantages, mainly with respect to models readability, and the compact and unambiguous descriptions which can be obtained.

It is worth noting that attributes can also be used successfully to model space and locations. For instance, an attribute could be used to denote the place where the individual is located ("in the wood", "in the nest", ...), or a pair (or triplet) of attributes (say x, y and z) could be used to describe the position of individuals in 2D|3D space. The latter approach could allow rules to be applied when, for instance, individuals are close enough to each other.

We will give now a formal definition of Attributed Probabilistic P System, then we will show a toy model. We will continue with some considerations about the main features of the formalism and a case study to show this kind of model full potential.

5.3 Attributed Probabilistic P systems, formal definition and semantics

As reminder, in this definition too, we denote with $\{a_1, \ldots, a_n\}$ the set or the multiset of objects $a_1, \ldots, a_n$. Moreover, we denote with $|w|$ the size (number of elements) of the multiset w, and with $-$ and $+$ the difference and the union of multisets, respectively.

Definition 4. *An* Attributed Probabilistic P system*, P, is a tuple $\langle V, arity, D_{a_1}, \ldots, D_{a_n}, w_0, R\rangle$ where:*

- *V is an ordered finite alphabet of symbols, $\{a_1, \ldots, a_n\}$;*
- *$arity : A \to \mathbb{N}$ is a function which for each $a_i \in A$ gives the arity of D_{a_i};*
- *each D_{a_i} is a set of tuples, $D_{a_i} = I_1 \times \ldots \times I_{arity(a_i)}$, where each I_j is a (possibly infinite) set of unstructured values; the set D_{a_i} is called the set of attributes of a_i;*
- *w_0 is a multiset of values in $\Sigma = \{\langle a_i, d_i\rangle \mid a_i \in A,\ d_i \in D_{a_i}\}$ describing the initial state of the system, where Σ is called the set of* objects *of P. Then we will write $w_0 \in \Sigma^*$.*
- *given a set of variables V, R is a finite set of evolution rules having the form*

$$u_{\mathrm{V}} \xrightarrow{f} v_{\mathrm{V}} \mid_{pr_{\mathrm{V}}}$$

where $u_{\mathrm{V}}, pr_{\mathrm{V}} \in \Sigma_{\mathrm{V}}^*$ are multisets (often denoted without brackets) of objects and variables denoting reactants *and* promoters*, respectively; $v_{\mathrm{V}} \in \Sigma_{\mathrm{EV}}^*$ is a multiset of objects and* expressions *with variables denoting* products*; and $f : \Sigma^* \mapsto \mathbb{R}^{\geq 0}$ is a* weight function*. Precisely:*

$$\Sigma_{\mathrm{V}} = \{(a_i, d_i) \mid a_i \in A, d_i \in D_{a_i}^{\mathrm{V}}\} \qquad \Sigma_{\mathrm{EV}} \qquad = \{(a_i, e_i) \mid a_i \in A, e_i \in E_{a_i}^{\mathrm{V}}\}$$

where $D^{\mathrm{V}}_{a_i} = (V \cup I_1) \times \ldots \times (V \cup I_{arity(a_i)})$*; and* $E^{\mathrm{V}}_{a_i} = Exp(V, I_1) \times \ldots \times Exp(V, I_{arity(a_i)})$*, with* $Exp(V, I)$ *denoting the set of well-typed expressions built from operators, variables* V*, and values of* I*. Moreover, we have* $Vars(v_{\mathrm{V}}) \subseteq Vars(u_{\mathrm{V}}) \cup Vars(pr_{\mathrm{V}})$*, where* $Vars(t)$ *denotes the set of variables occurring in* t*. Rules without variables are called* ground rules.

In what follows we will denote an (attributed) object $\langle a, d \rangle$ as $a_{(d)}$. A state (or configuration) of an APP system is a multiset of objects in Σ^*. By definition, the initial state is w_0, and we denote a generic state as w.

The evolution of an APP system is a sequence of probabilistic maximally parallel steps. We formally define the semantics of APP systems as a transition relation in the style of [79]. In each step a maximal multiset of evolution rule instances is selected and applied as described by the following semantic rules:

$$\text{(rule application)} \qquad \frac{r_i = \; u \xrightarrow{k} v \in R \qquad u \subseteq w_a \qquad K = \{|k' | u' \xrightarrow{k'} v' \in R,\; u' \subseteq w_a|\} \qquad p = k / \sum_{k' \in K} k'}{(w_a, w_p) \xrightarrow{r_i,\; p}_R (w_a - u, w_p + v)}$$

$$\text{(single rule sequence)} \qquad \frac{(w_a, w_p) \xrightarrow{r_i,\; p}_R (w'_a, w'_p)}{(w_a, w_p) \xrightarrow{[r_i],\; p}{}^{+}_R (w'_a, w'_p)}$$

$$\text{(multiple rules sequence)} \qquad \frac{(w_a, w_p) \xrightarrow{r_i,\; p_i}_R (w'_a, w'_p) \qquad (w'_a, w'_p) \xrightarrow{\overline{r},\; \overline{p}}{}^{+}_R (w''_a, w''_p)}{(w_a, w_p) \xrightarrow{\overline{r} @ [r_i],\; p_i \cdot \overline{p}}{}^{+}_R (w''_a, w''_p)}$$

$$\text{(step rule)} \qquad \frac{(w, \emptyset) \xrightarrow{\overline{r},\; \overline{p}}{}^{+}_{R(w)} (w_a, w_p) \qquad (w_a, w_p) \nrightarrow_{R(w)}}{w \xmapsto{\overline{r},\; \overline{p}}_R w_a + w'_p}$$

where $[r_i]$ denotes the sequence composed of the single element r_i, and @ denotes the concatenation of sequences.

Given a system state, w, the (step rule) describes the evolution in a new state by the $\xmapsto{\overline{r},\; \overline{p}}_R$ relation, where $\overline{p}$ is the probability of the transition, and $\overline{r}$ is the sequence of applied ground rules. (step rule) invokes $(w, \emptyset) \xrightarrow{\overline{r},\; \overline{p}}{}^{+}_{R(w)} (w_a, w_p)$ where $R(w)$ is the set of applicable ground rules in the state w, with their weights, namely:

$$R(w) = \left\{ u_{\mathrm{V}}\sigma \xrightarrow{f(w)} v_{\mathrm{V}}\sigma \;\middle|\; u_{\mathrm{V}} \xrightarrow{f} v_{\mathrm{V}} \;|_{pr_{\mathrm{V}}} \in R,\; \exists \sigma.\, u_{\mathrm{V}}\sigma \subseteq w \wedge pr_{\mathrm{V}}\sigma \subseteq w \right\}$$

where (i) $\sigma : V \to flat(D_{a_1}) \cup \ldots \cup flat(D_{a_n})$, with $flat(D_{a_i}) = I_1 \cup \ldots \cup I_{arity(a_i)}$, for all $a_i \in A$; (ii) $u_{\mathrm{V}}\sigma$ ($u_{\mathrm{V}} \in \Sigma^*_{\mathrm{V}}$) is the well-typed multiset obtained by substituting values for variables in u_{V} according to σ; and (iii) $v_{\mathrm{V}}\sigma$ ($v_{\mathrm{V}} \in \Sigma^*_{\mathrm{EV}}$) is the well-typed multiset obtained by evaluating the expressions in v_{V} under the substitution σ. Transition relation $\xrightarrow{\overline{r},\; \overline{p}}{}^{+}_R$ is the transitive closure of $\xrightarrow{r,\; p}_R$.

A transition $(w_a, w_p) \xrightarrow{r_i,\; p}_R (w_a - u, w_p + v)$ corresponds to the application of a single rule. When a rule is selected, its application consists in removing its reactants from w_a and adding its products to w_p. The w_p multiset will collect all products of all applied rules. Note that $R(w)$ takes into account that each rule is applied with respect to the weights of the rules computed in the initial state w. Moreover, $R(w)$ contains only the ground rules. Their promoters are present in the initial state w ($pr_{\mathrm{V}}\sigma \subseteq w$). Once objects in w_a are such that no further rule in $R(w)$ can be applied to them, by (step rule) the new system state is $w_a + w_p$ (where w_a are the unused objects and w_p are the new products).

Intuitively, the semantic definition states that all the rules to be applied are selected in a probabilistic way from the set of applicable rules, their reactants are removed for the

available reactants, w_a, and their products are added to a suspended multiset w_p. When no further rule can be applied to w_a the new state, which is composed by the unused objects in w_a plus the suspended products in w_p, is produced. Finally we give the probability of a transition between two states by means of the following rule:

$$\text{(state transition prob.)} \qquad \frac{PR = \{(\overline{r}, \overline{p}) |\ w \overset{\overline{r},\ \overline{p}}{\longmapsto}_R w'\} \qquad p \ = \sum_{(\overline{r},\overline{p}) \in PR} \overline{p}}{w \overset{p}{\Longrightarrow}_R w'}$$

5.4 Simple example of modelling: the protozoans

The purpose of this first example is to show some features of APP systems. We propose a simple model with a representation of the space, some example of rules affect attributes of reagents and the use of control objects who last more than one computational step. This example will help to get along some basic concepts before the chapter's real case study.
We consider a population of sexually reproducing protozoans in which two individuals are necessary to produce offspring.
Protozoans are free ranging on a laboratory Petri dish. The Petri dish is abstracted by a $n \times n$ grid and protozoans can move one step at a time on the grid. They can reproduce if they meet in the same grid entry. They have a finite lifespan, at the end of which they die.
Each individual is represented by an attributed object $p_{(a,x,y)}$ in which p stands for "protozoan", attribute a is an integer representing the age of the individual, and attributes x and y are the coordinates of the position of the individual on the Petri dish.
The evolution cycles among three phases: a *movement* phase, a *reproduction* phase, and an *aging* phase.
Each phase is represented by a different symbol, namely *move*, *repr*, and *aging*, respectively, which are used as promoters to enable different sets of rules for each phase. The movement phase has a duration of two maximally-parallel steps, hence the *move* symbol has an attribute taking values from the set $\{1, 2\}$ to allow modelling it.

For the sake of simplicity we assume a 4×4 grid and a lifespan of 3 age units. We consider an initial configuration in $move_{(1)}$ phase with two individuals, of age 1, in positions $(1, 2)$ and $(2, 0)$, respectively.
The APP system is shown in Figure 5.1.

Rules from r_1 to r_8 model the movement phase, with a different rule for each possible direction of movement: rule r_1 for eastward movement, rule r_2 for northward movement, and so on, also allowing diagonal movement as exemplified by rule r_8. Rule r_9 handles the reproduction phase, while rules r_{10}, r_{11} model the aging phase. Finally, rules r_{12}–r_{15} are used to switch phases. Note that all the weights associated with the rules are constant and equal to 1, thus for each phase all (and only) the rules that are specific to that phase can be applied, and such rules are equiprobable. An example of evolution of the system is shown in Figure 5.2 with (a)–(f) representing a possible sequence of states reached by the system. The caption of each figure contains the complete multiset of objects present in the system.(For compactness, only the age attributes are shown for the objects in the figures, that is, $p_{(a,x,y)}$ is depicted as p_a.

5.5 Discussion about APP main features

In the example and in APP systems formal definition we see how attributes help us to define elements and are used in rules to give multivalue definition, but still, in this section,

${}^aP_{\text{prot}} = (A, arity, D_p, D_{move}, D_{repr}, D_{aging}, w_0, R)$ where:

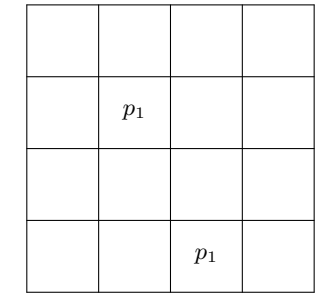
$A = \{p, move, repr, aging\}$ $\quad arity = \{p \mapsto 3, move \mapsto 1, repr \mapsto 0, aging \mapsto 0\}$

$D_p = \{0,1,2\} \times \{0,1,2,3\} \times \{0,1,2,3\}$;

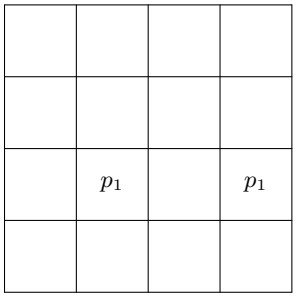
$D_{move} = \{1,2\}$; $D_{move2} = D_{repr} = D_{aging} = \emptyset$ $\quad w_0 = \{\!|move_{(1)},\ p_{(1,1,2)},\ p_{(1,2,0)}|\!\}$

$$R = \left\{ \begin{array}{lll}
r_1: & p_{(a,x,y)} \xrightarrow{1} p_{(a,x+1,y)} \,|_{move_{(n)}} & \text{if } x < 3 \\
r_2: & p_{(a,x,y)} \xrightarrow{1} p_{(a,x,y+1)} \,|_{move_{(n)}} & \text{if } y < 3 \\
r_3: & p_{(a,x,y)} \xrightarrow{1} p_{(a,x-1,y)} \,|_{move_{(n)}} & \text{if } x > 0 \\
r_4: & p_{(a,x,y)} \xrightarrow{1} p_{(a,x,y-1)} \,|_{move_{(n)}} & \text{if } y > 0 \\
r_5: & p_{(a,x,y)} \xrightarrow{1} p_{(a,x-1,y+1)} \,|_{move_{(n)}} & \text{if } x > 0 \ \wedge y < 3 \\
r_6: & p_{(a,x,y)} \xrightarrow{1} p_{(a,x+1,y-1)} \,|_{move_{(n)}} & \text{if } x < 3 \ \wedge y > 0 \\
r_7: & p_{(a,x,y)} \xrightarrow{1} p_{(a,x+1,y+1)} \,|_{move_{(n)}} & \text{if } x < 3 \ \wedge y < 3 \\
r_8: & p_{(a,x,y)} \xrightarrow{1} p_{(a,x-1,y-1)} \,|_{move_{(n)}} & \text{if } x > 0 \ \wedge y > 0 \\
r_9: & p_{(a_1,x,y)}, p_{(a_2,x,y)} \xrightarrow{1} p_{(a_1,x,y)}, p_{(a_2,x,y)}, p_{(0,x,y)} \,|_{repr} & \\
r_{10}: & p_{(a,x,y)} \xrightarrow{1} p_{(a+1,x,y)} \,|_{aging} & \text{if } a < 2 \\
r_{11}: & p_{(2,x,y)} \xrightarrow{1} |_{aging} & \\
r_{12}: & move_{(1)} \xrightarrow{1} move_{(2)} & r_{13}: move_{(2)} \xrightarrow{1} repr \\
r_{14}: & repr \xrightarrow{1} aging & r_{15}: aging \xrightarrow{1} move_{(1)}
\end{array} \right\}$$

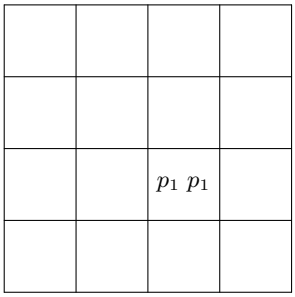
Figure 5.1: APP system modelling protozoans



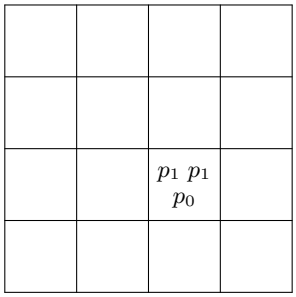
(a) $move_{(1)}, p_{(1,1,2)}, p_{(1,2,0)}$



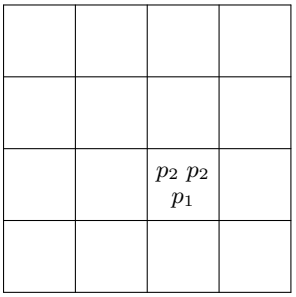
(b) $move_{(2)}, p_{(1,1,1)}, p_{(1,3,1)}$



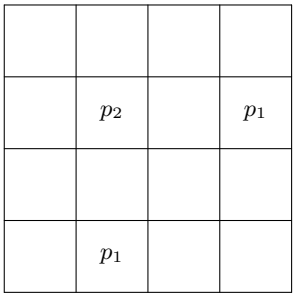
(c)$repr, p_{(1,2,1)}, p_{(1,2,1)}$



(d)$aging, p_{(1,2,1)}, p_{(1,2,1)}, p_{(0,2,1)}$



(e)$move_{(1)}, p_{(2,2,1)}, p_{(2,2,1)}, p_{(1,2,1)}$



(f)$move_{(2)}, p_{(1,1,0)}, p_{(1,3,2)}, p_{(2,1,2)}$

Figure 5.2: Example of the evolution of the protozoans model, with (a)-(f) representing a possible sequence of states reached by the system. The caption of each figure contains the complete multiset of objects present in the system. (For compactness, only the age attributes are shown for the objects in the figures, that is, $p_{(a,x,y)}$ is depicted as p_a).

we want to highlight two other specific uses of attributes in APP systems.

5.5.1 Attributes and control objects

Attributes have a special use associated to control object. Control objects are elements of the alphabet and so they can have attributes like any other elements of the model alphabet. One of the most intuitive attributes a control object can have is how many computational steps we want the control object to last.
For example we want the movement phase associated with the control object MOV to last three consecutive computational steps before shifting into another phase. To model that fact we can use the attribute "last" and set its value to "3" to define MOV_3.
Associated to the control object we will have two rules:
$MOV_{last>0} \longrightarrow MOV_{last-1}|MOV$
$MOV_0 \longrightarrow AGG_{Agg\_step}|MOV$

The first rule reduces the attribute "last" of the control object until it reaches zero, the second rule switch the control object MOV_0 in the control object AGG with the attribute $last = Agg\_step$, Agg_step is a parameter of the simulation and defines how many consecutive computational steps we want the "AGG" phase to last.

5.5.2 Attributes and probabilistic functions

We consider now the role of attributes in the probabilistic function associated to the rules. Now, in the probabilistic functions, we can use attributes too as input and this is a very powerful tool to model the non-determinism with a finer probabilistic structure.
For example we consider two rules which clearly describe the outcome of a fight between two protozoans of antagonist species (ProtozoanRed and ProtozoanBlue). Each protozoan has a different value for the attribute "size" and they try to eat each other

$$ProtozoanRed_{size1}, ProtozoanBlue_{size2} \overset{f(size1,size2)}{\rightarrow} ProtozoanRed_{size1+size2/2}|AGG_r$$
$$ProtozoanRed_{size1}, ProtozoanBlue_{size2} \overset{g(size1,size2)}{\rightarrow} ProtozoanBlue_{size2+size1/2}|AGG_r$$

If these two rules are the only two rules enabled during the AGG (aggression) phase, then it is easy to see that the chance to win for the Red Protozoan is:

$$P_{winRed}(size1, size2) = \frac{f(size1,size2)}{f(size1,size2)+g(size1,size2)}$$

and it depends on the sizes of the two protozoans.
In APP systems the probability of the outcome depends not only on the state of the membrane but also on the attributes of the reagents.
This kind of probabilistic function can take as input attributes of promoters as well. For example a model can have one element which models the acidity of the Petry dish. Taking into account the presence of these elements we will have a upgraded version of the rules above:

$$ProtozoanRed_{size1}, ProtozoanBlue_{size2} \overset{f(size1,size2,n)}{\rightarrow} ProtozoanRed_{size1+size2/2}|AGG_r, ACID_n$$
$$ProtozoanRed_{size1}, ProtozoanBlue_{size2} \overset{g(size1,size2,n)}{\rightarrow} ProtozoanBlue_{size2+size1/2}|AGG_r, ACID_n$$

Again the Red Protozoan chance to prevail is :

$P_{winRed}(size1, size2, n) = \frac{f(size1,size2,n)}{f(size1,size2,n)+g(size1,size2,n)}$

Probabilistic functions of this kind are used in APP systems as a natural consequence of the introduction of attributes associated to individuals.
If we analyze a simple "aging" rule over the element $Protozoan_{age}$ like :

$Protozoan_n \overset{f(n)}{\rightarrow} Protozoan_{n+1}$

This rule will be selected with rate $r = f(n)$. The reagent $Protozoan_n$ is removed from the state of the membrane and the product $Protozoan_m$ is added to the state of the membrane where $f(n) = m = n+1$ is the function which give a value to the attribute of the product. The effect of the rule is trivial, just add one to the age of the Protozoan, but the idea behind is not so simple. Something is consumed, something is created anew and the description of the rule contains not only the probabilistic rate but also all the functions associated to any attributes of the product(s). This happens because the formalism does not admit an "undefined" value for the attribute associate to one element in the state of the system. The value of attributes of any product must be specified in the rule which create them.

5.6 Modelling social interaction in primates with APPS

In this model, we describe the behaviour of male monkeys and how it changes when a female monkey enters the oestrum taking as reference species the *lemur catta*.
This is a perfect case study to show the potential of APP systems. We can exploit the parallelism inherited by original P systems, model the outcome of fights between male monkeys with the probabilistic structure over rules, use attributes to characterize the different specimens and model the space with attributes over a coordinate system. We also use two sets of control objects to manage different phases of computations asynchronously. The model is inspired by the social behaviors of species of *lemur catta* as described in [82, 81, 80]. The population taken into account is dispersed in an environment, which is modelled as a continuous 2D space. Each individual is associated with coordinates (x, y). Male and female monkeys are represented by symbols MMonkey and FMonkey, respectively, and both have attributes in the domain $\mathbb{R}^2 \times \mathbb{N}$. Beyond the actual position, we also keep track of the *dominance* level of each individual, which is used to derive the likeliness of the individual to win (or just engage in) a fight against another individual.

At the beginning, the population is composed only by male monkeys having a dominance level of 1500. All the male monkeys alternate between two phases: a *movement* phase, represented by the special symbol MOV, in which they wander around slowly and move towards other individuals in order to keep the population compact; and a *fight* phase, represented by FGT, in which they chase other individuals to fight, yielding to variations in their levels of dominance. Females alternate between a *normal* phase, denoted by the symbol NORMAL, and an *oestrum* phase, denoted by OEST. In this model, for simplicity, there is only one female monkey, which is explicitly represented only during the oestrum phase. In other words, the individual FMonkey appears only at the beginning of the oestrum phase, and is removed from the model at the end of the phase.

Formally, the model is composed of 6 symbols $A = \{\text{MMonkey}, \text{FMonkey}, \text{MOV}$ $, \text{FGT}, \text{NORMAL}, \text{OEST}\}$, having a corresponding set of attributes defined as $D = \{(\mathbb{R}^2 \times \mathbb{N}), (\mathbb{R}^2 \times \mathbb{N}), \mathbb{N}, \mathbb{N}, \mathbb{N}, \mathbb{N}\}$.

As regards symbols $\text{MOV}, \text{FGT}, \text{NORMAL}, \text{OEST}$, an attribute from $\mathbb{N}$ is associated with each of them, denoting the length of those phases in terms of the steps taken by the

Attributed P system. The initial state of the system is the following:

$$\begin{aligned} w_0 = \{&\text{MMonkey}_{(x_1,y_1,1500)}, \text{MMonkey}_{(x_2,y_2,1500)}, \text{MMonkey}_{(x_3,y_3,1500)}, \\ &\text{MMonkey}_{(x_4,y_4,1500)}, \text{MMonkey}_{(x_5,y_5,1500)}, \text{MMonkey}_{(x_6,y_6,1500)}, \\ &\text{MMonkey}_{(x_7,y_7,1500)}, \text{MMonkey}_{(x_8,y_8,1500)}, \text{MOV}_{(1)}, \text{NORMAL}_{(Snl)}\} \end{aligned}$$

where the positions of the individuals $(x_i, y_i) \in \mathbb{R}^2$ are randomly generated within a square area of $cage \in \mathbb{R}$ side length. Parameter Snl denotes the duration of the "normal" phase for females.

The evolution rules of the model are as follows.

$$\begin{aligned} r_1 &: \text{MOV}_{(n)} \xrightarrow{1} \text{MOV}_{(n-1)} && \forall n > 1 \\ r_2 &: \text{MOV}_{(1)} \xrightarrow{1} \text{FGT}_{(Nfl)}|_{\text{NORMAL}_{(n)}} \\ r_3 &: \text{MOV}_{(1)} \xrightarrow{1} \text{FGT}_{(Ofl)}|_{\text{OEST}_{(n)}} \\ r_4 &: \text{FGT}_{(n)} \xrightarrow{1} \text{FGT}_{(n-1)} && \forall n > 1 \\ r_5 &: \text{FGT}_{(1)} \xrightarrow{1} \text{MOV}_{(Msn)} \end{aligned}$$

Rules r_1–r_5 model the alternation between "movement" and "fight" phases for males. For both MOV and FGT, their attributes decrease to keep track of the number of steps passed, until they reach 1 and a phase switch occurs. In particular, while switching from MOV and FGT, the number of steps that the fight phase lasts depend on the phase of the female; namely it is either Nfl or Ofl, if the female is currently either in the normal phase (NORMAL) or the oestrum phase (OEST), respectively.

The movement phase lasts Msn steps.

$$\begin{aligned} r_6 &: \text{NORMAL}_{(n)} \xrightarrow{1} \text{NORMAL}_{(n-1)} && \forall n > 1 \\ r_7 &: \text{NORMAL}_{(1)} \xrightarrow{1} \\ &\qquad \text{OEST}_{(Sol)}, \text{FMonkey}_{(FemaleInitX,FemaleInitY,FemaleInitDom)} \\ r_8 &: \text{OEST}_{(n)} \xrightarrow{1} \text{OEST}_{(n-1)} && \forall n > 1 \\ r_9 &: \text{OEST}_{(1)}, \text{FMonkey}_{(x,y,dom)} \xrightarrow{1} \text{NORMAL}_{(Snl)} \end{aligned}$$

Rules r_6–r_9 model the alternation between "normal" and "oestrum" phases for the female monkey. The durations of the oestrum phase is modelled by the parameter Sol. The initial coordinates of the female monkey are denoted by the parameters $FemaleInitX$ and $FemaleInitY$, while $FemaleInitDom$ denotes its initial dominance level.

$$\begin{aligned} r_{10} &: \text{MMonkey}_{(x',y',dom')} \xrightarrow{f_{10}} \text{MMonkey}_{(\text{move}(x',x'',SMMA),\text{move}(y',y'',SMMA),dom')} \\ &\qquad |_{\text{MOV}_{(n)},\text{NORMAL}_{(m)},\text{MMonkey}_{(x'',y'',dom'')}} \\ r_{11} &: \text{MMonkey}_{(x',y',dom')} \xrightarrow{f_{11}} \text{MMonkey}_{(\text{move}(x',x'',SMMA),\text{move}(y',y'',SMMA),dom')} \\ &\qquad |_{\text{MOV}_{(n)},\text{OEST}_{(m)},\text{MMonkey}_{(x'',y'',dom'')}} \end{aligned}$$

Rules r_{10}–r_{11} handle the movement of males during either the "normal" or "oestrum" phase for the female. In both cases, a male (in a position x', y') is allowed to move towards

any other male (in position x'', y''). The resulting position of the male which moves is computed as $(\text{move}(x', x'', SMMA), \text{move}(y', y'', SMMA))$, where move is a function to move the coordinates (x', y') towards coordinates (x'', y'') with a given speed factor described by the parameter $SMMA$ (Speed Male-Male approach).
Formally, this function is defined as:

$$\text{move}(a, b, \gamma) = a + (b - a)/\gamma$$

The most important difference between rules r_{10} and r_{11} lies in the weight functions, which are defined as:

$$f_{10} = \begin{cases} 1 & \text{if } SD/NT < dist((x', y'), (x'', y'')) < SD; \\ 0 & \text{otherwise;} \end{cases}$$

$$f_{11} = \begin{cases} 1 & \text{if } SD/OT < dist((x', y'), (x'', y'')) < SD; \\ 0 & \text{otherwise.} \end{cases}$$

where $dist((x', y'), (x'', y''))$ is a function giving the euclidean distance between two points, parameter SD (Spot Distance) denotes the maximum visibility distance of a monkey, and parameters NT (Normal Tolerance) and OT (Oestrum Tolerance) are used to derive the minimum distance allowed between two monkeys to enable the relative movement of one towards the other.
In particular, such a minimum distance depends on the phase of the female, and it is either SD/NT during the NORMAL phase, and SD/OT during the OEST phase. In this manner, during the oestrum phase, males are allowed to come closer one another, hence increasing the possibility to engage in a fight.

$$r_{12} : \text{MMonkey}_{(x', y', dom')} \xrightarrow{f_{12}} \text{MMonkey}_{(\text{move}(x', x'', SMFA), \text{move}(y', y'', SMFA), dom')} \\ \big|_{\text{MOV}_{(n)}, \text{FMonkey}_{(x'', y'', dom'')}}$$

Rule r_{12} models the movement of a male monkey towards the female. In this case, the speed of the male is denoted by the parameter $SMFA$. The corresponding weight function is:

$$f_{12} = \begin{cases} PfF & \text{if } dist((x', y'), (x'', y'')) < SD \text{ and } dom' + FT > dom''; \\ 0 & \text{otherwise;} \end{cases}$$

which enables the movement only if (i) their relative distance is less than SD, and (ii) the dominance level of the male, plus a tolerance value FT (Female Tolerance), is greater than that of the female. The actual weight used is denoted by the parameter PfF (Preference for Female).

$$r_{13} : \text{MMonkey}_{(x', y', dom')}, \text{MMonkey}_{(x'', y'', dom'')} \xrightarrow{elo\_rating(dom', dom'')} \\ \text{MMonkey}_{(\text{chase}(x', x'', CSN), \text{chase}(y', y'', CSN), elow(dom', dom''))}, \\ \text{MMonkey}_{(\text{flee}(x', x'', FSN), \text{flee}(y', y'', FSN), elol(dom'', dom'))} \big|_{\text{FGT}_{(n)}, \text{NORMAL}_{(m)}}$$

with $dom' \geq dom''$, $dist((x', y'), (x'', y'')) \leq NAD$, and $dom' - dom'' \leq AN$, where NAD (Normal Aggression Distance) and AN (Avoidance Normal) and model parameters representing the minimum distance and the maximum difference in dominance that enables an aggression when the female is in normal condition. In this rule the monkey in position (x', y') has a dominance that is higher or equal to that of the other monkey. The probability

that the first monkey wins the fight is given by the standard Elo rating method, originally defined for rating strength in games, and then used for modelling social interactions. Such a method is based on a table that gives the probability of success in a fight depending on the difference of rating (or dominance) of the involved individuals. The Elo rating table we consider for this model is in [78]. The function $elo\_rating(\Delta dom)$ looks in the table and gives as result the probability of the victory of the stronger monkey over the weaker one.

Function chase gives the new position of the winner of the fight; function flee gives the new position of the loser of the fight; *elow* gives the new dominance of the winner of the fight following the Elo rating table and method; *elol* the new dominance of the loser of the fight. These functions are defined as follows:

$$\text{chase}(a,b,\rho) = a + \rho \cdot (b-a) \qquad \text{flee}(a,b,\rho) = b + \rho \cdot (b-a)$$

$$elow(d',d'') = d' + \begin{cases} (1 - elo\_rating(\Delta dom)) \cdot stepness & \text{if } d' > d'' \\ elo\_rating(\Delta dom) \cdot stepness & \text{if } d' < d'' \end{cases}$$

$$elol(d',d'') = d' - \begin{cases} elo\_rating(\Delta dom) \cdot stepness & \text{if } d' > d'' \\ 1 - (elo\_rating(\Delta dom)) \cdot stepness & \text{if } d' < d'' \end{cases}$$

where $\Delta dom = |d' - d''|$, and *stepness* is a parameter representing the maximum increase/decrease of dominance. The parameters CSN(Chase Speed Normal) and FSN (Flee Speed Normal) used in rule r_{13} describe how fast the monkeys move.

$$\begin{aligned} r_{14} : \text{MMonkey}_{(x',y',dom')}, \text{MMonkey}_{(x'',y'',dom'')} &\xrightarrow{1-elo\_rating(dom',dom'')} \\ &\text{MMonkey}_{(\text{flee}(x',x'',FSN),\text{flee}(y',y'',FSN),elol(dom',dom''))}, \\ &\text{MMonkey}_{(\text{chase}(x',x'',CSN),\text{chase}(y',y'',CSN),elow(dom'',dom'))} |_{\text{FGT}_{(n)}\,\text{NORMAL}_{(m)}} \end{aligned}$$

with $dom' > dom''$, $dist((x',y'),(x'',y'')) \leq NAD$, and $|dom' - dom''| \leq AN$.
Rule r_{14} is analogous to rule r_{13}, but describes the case in which the winner is the weaker monkey.

$$\begin{aligned} r_{15} : \text{MMonkey}_{(x',y',dom')}, \text{MMonkey}_{(x'',y'',dom'')} &\xrightarrow{elo\_rating(dom',dom'')} \\ &\text{MMonkey}_{(\text{chase}(x',x'',CSO),\text{chase}(y',y'',CSO),elow(dom',dom''))}, \\ &\text{MMonkey}_{(\text{flee}(x',x'',FSO),\text{flee}(y',y'',FSO),elol(dom'',dom'))} |_{\text{FGT}_{(n)},\text{OEST}_{(m)}} \end{aligned}$$

with $dom' \geq dom''$, $dist((x',y'),(x'',y'')) \leq OAD$, and $dom' - dom'' \leq AO$.

$$\begin{aligned} r_{16} : \text{MMonkey}_{(x',y',dom')}, \text{MMonkey}_{(x'',y'',dom'')} &\xrightarrow{1-elo\_rating(dom',dom'')} \\ &\text{MMonkey}_{(\text{flee}(x',x'',FSO),\text{flee}(y',y'',FSO),elol(dom',dom''))}, \\ &\text{MMonkey}_{(\text{chase}(x',x'',CSO),\text{chase}(y',y'',CSO),elow(dom'',dom'))} |_{\text{FGT}_{(n)},\text{OEST}_{(m)}} \end{aligned}$$

with $dom' > dom''$, $dist((x',y'),(x'',y'')) \leq OAD$, and $|dom' - dom''| \leq AO$.
Rules r_{15} and r_{16} are analogous to r_{13} and r_{14}, respectively, but describe the case in which the female is in oestrum state. Parameters OAD(Oestrum Aggression Distance), AO (Avoidance Oestrum), CSO (Chase Speed Oestrum) and FSO (Flee Speed Oestrum)

of these rules are analogous to the corresponding ones of rules r_{13} and r_{14}, but with values that depend on the fact that the female is in oestrum state.

$$r_{17} : \text{MMonkey}_{(x',y',dom')}, \text{MMonkey}_{(x'',y'',dom'')} \xrightarrow{1}$$
$$\text{MMonkey}_{(\text{chase}(x',x'',CSN),\text{chase}(y',y'',CSN),dom')},$$
$$\text{MMonkey}_{(\text{flee}(x',x'',FSN),\text{flee}(y',y'',FSN),dom'')} | \text{FGT}_{(n)}, \text{NORMAL}_{(m)}$$

with $dom' > dom''$, $dist((x', y'), (x'', y'')) \leq NAD$ and $|dom' - dom''| > AN$.

$$r_{18} : \text{MMonkey}_{(x',y',dom')}, \text{MMonkey}_{(x'',y'',dom'')} \xrightarrow{1}$$
$$\text{MMonkey}_{(\text{chase}(x',x'',CSO),\text{chase}(y',y'',CSO),dom')}$$
$$\text{MMonkey}_{(\text{flee}(x',x'',FSO),\text{flee}(y',y'',FSO),dom'')} | \text{FGT}_{(n)}, \text{OEST}_{(m)}$$

with $dom' > dom''$, $dist((x', y'), (x'', y'')) \leq OAD$ and $|dom' - dom''| > AO$.

Rules r_{17} and r_{18} describe the interaction between two individuals when the difference in dominance is too high to motivate a fight (greater than parameters AN and AO for the normal and oestrum cases, respectively). In these cases the monkeys move but do not fight, so there is no change in dominance levels.

5.6.1 Parameters of the simulation

The complete set of parameters used to perform the simulations is:

The **system parameters** are just relative to computations, and are not strongly related to the species we try to model. In the next section we will better explain the idea behind the value and the estimations of these parameters:

- *SOL* (starting Oestrum loop) number of consecutive computation steps under "oestrum" condition.
- *SNL* (Starting Normal Loop) number of consecutive computation steps under "normal" condition.
- *cage* dimension of the area Male monkeys are placed randomly in during the setup of the model.

This **density parameters** are related to the density of the troop of monkeys, model different population who live more or less dispersed in the habitat:

- move(x', x'') and move(y', y'') are two function both depending on the hidden parameter *SMMA* (Speed Male-Male approach).
- *SD* (Spot Distance) maximum distance within which two monkeys interact and move one toward the other.
- *NT* (Normal Tolerance) fraction of the spot distance within which monkeys don't want to move closer each other under "normal" condition.
- *OT* (Oestrum Tollerance) fraction of the spot distance within which monkeys don't want to move closer to each other under "oestrum" condition.
- *PfF* (Preference for Female) this parameter models the preference that a male has to move towards a female rather than to another male in "movement" loops.

The **social parameters** models the social behavior of the monkeys, if they are more or less aggressive or if they have a more or less strict hierarchy

- Nfl (Normal fight loop) number of fight loop for anyfull computation step under "normal"condition.
- Ofl (Oestrum fight loop) number of figth loop for any iterative step under "oestrum" condition, usually $Ofl > Nfl$ due to increased aggressiveness during Oestrum.
- FT (Female Tolerance) if male monkeys want move closer to female the gap between the dominance of the male and the female must be less than Female Tollerance or the female will not accept the proximity of the male.
- *chasex* and *chasey* are two functions both depending on the hidden parameters CSN(Chase Speed Normal), CSO (Chase Speed Oestrum), FSN (Flee Speed Normal) and FSO (Flee Speed Oestrum).
- *elow* (elo-rating winner) and *elol* (elo-rating looser) take as hidden parameter the classic elo-rating *stepness* and the elo-rating table with probability to win/lose depending on dominance of the contenders.
- NAD (Normal Aggression Distance) distance within which two male monkeys can engage in combat under "normal" condition.
- AN (Avoidance Normal) if gap in dominance between two male is bigger than AN the fight will not take place and the less dominant monkey will flee from the most dominant one without being engaged in combat, this under "normal" condition.
- OAD (Oestrum Aggression Distance) distance within which two male monkeys can engage in combat under "oestrum"condition.
- AO (Avoidance Oestrum) if gap in dominance between two males is bigger than AN the fight will not take place and the less dominant monkey will flee from the most dominant one without being engaged in combat, this under "oestrum" condition. Usually $AO > AN$ because Male Monkeys in "oestrum" condition of the system are more willing to fight even with very low chances of winning.

5.6.2 Use of parameters in the model

The estimation of parameters that are used in the rules is a crucial point in modelling. The parameters' settings depend on the model's goal: as we have shown these instruments can be used both to generate predictive models, and to validate theories or to estimate parameters.
In the case study that is being presented, parameters are chosen within a plausible and realistic range; we haven't used exact parameters to show how the model could develop predictions but, on the contrary, we chose parameters that could generate a model of possible species. The simulations' results were then compared with field observations to validate their correspondence with lemurs behaviour.
Even if it cannot directly give exact predictions, this kind of approach is very useful to show to scholars who study these kinds of animals correlations between parameters and the possible results; moreover they can provide realistic paremeters estimations, and observing the model's result can stimulate further observations and experiments, helping steering the direction of the research effort.
In conclusion: the parameters that have been used in these simulations have been chosen within ranges observed in the referred publications, and have been combined to model hypotetic prossimian populations with different social, density, population, and aggression characteristics.
Combinations of these parameters can easily produce characteristics that have never been

associated to any observed species, nevertheless these simulations' results have validity because they give researches hints on what could be expected, and suggest new research topics.

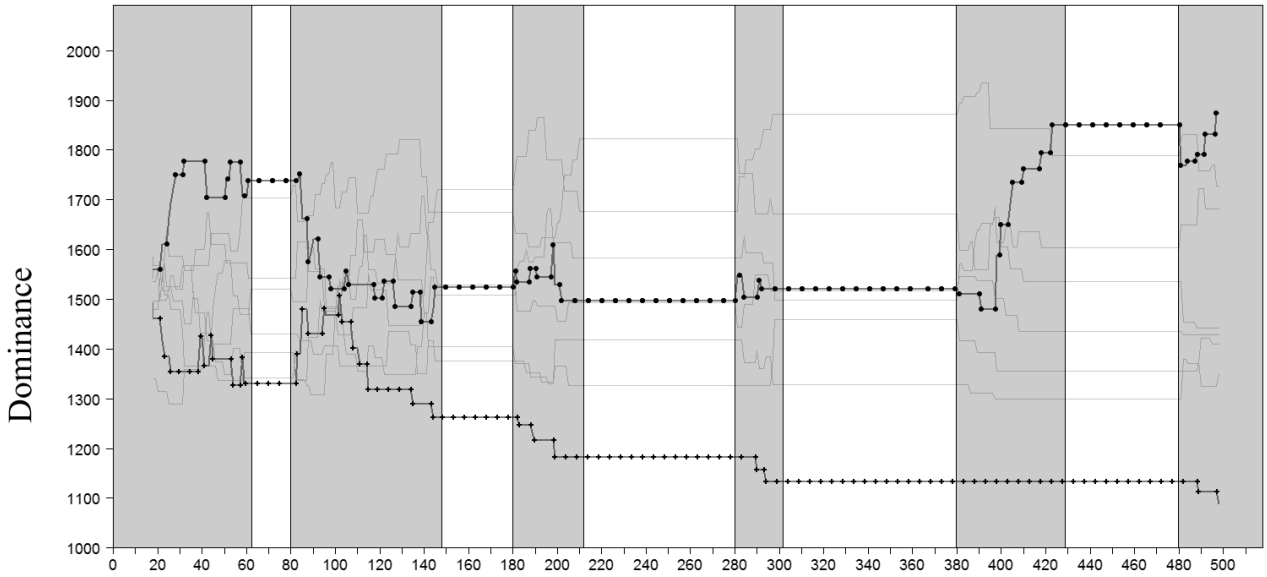
5.7 Experimental results

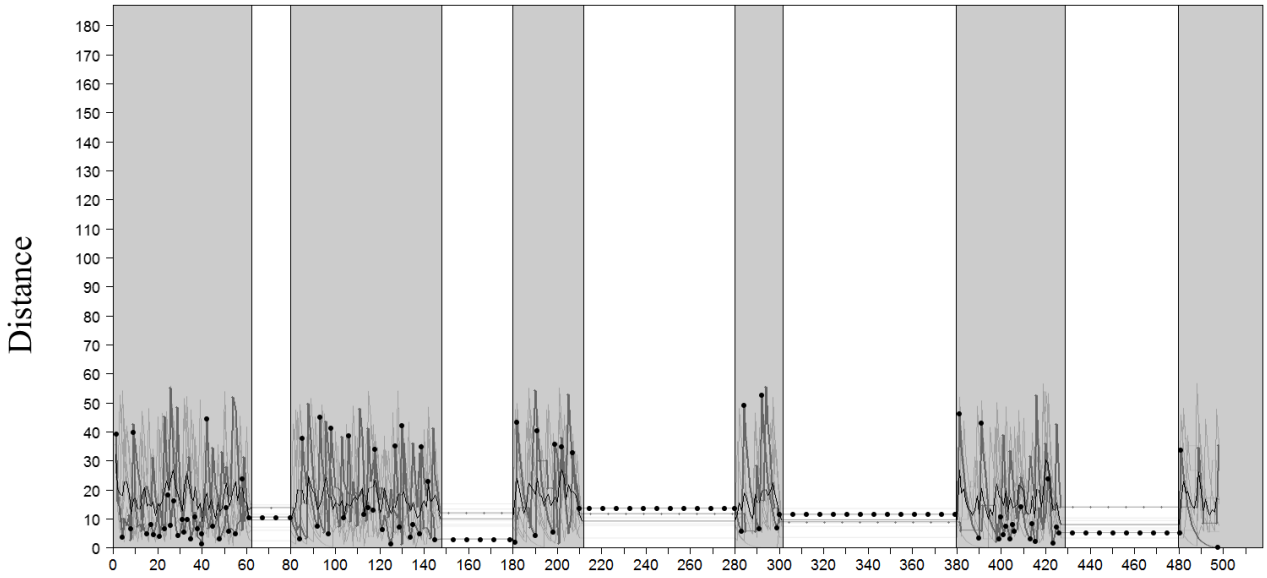
We studied the dynamics of the APP model described in the previous section by running simulations. In particular, we implemented an APP systems interpreter in C++ that allows attributed objects and evolution rules to be represented as instantiations of specific C++ classes.
Once an APP system model is specified, the interpreter simulates it by performing a number of iterations to be given as a parameter. In each iteration, a maximally parallel step is performed according to the APP systems semantics.
The result of a simulation is the sequence of configurations reached by the interpreter at each iteration. In order for the interesting measurements to be easily readable, we processed the simulation results and produced graphical representations by using the statistical framework R.

In Figure 5.3 and in Figure 5.4 we show the dynamics of two groups of lemurs. In particular, Figure 5.3 refers to a group with a low level of aggression(egalitarian), while Figure 5.4 describes a group with a higher level of aggression (despotic).
The upper part of both figures shows the dominance level of each male in the group during the simulation, while the lower part shows the distance of each male either from the center of the group or from the female (when present). In the figures we put in evidence the lines corresponding to both the monkey with highest dominance level at the end of the simulation (line marked with $\bullet$) and the one with the lowest one (marked with $+$).
The main model parameters (that are different in the two cases) are reported in the figures. The other parameters have, in both cases, the following values: iteration= 498, Sol= 20, Snl= 80, SMMA= 0.25, SD= 100, NT = 3, OT = 2, SMFA = 0.25, PfF = 8, Msn = 1 and Stepness = 100.
Note that we ran the simulations for 498 iterations.
The time corresponding to an iteration is, in the real world, about a few hours. Actually, such a time could vary among the different phases described by the model. However, this is not a problem since we are not interested in a precise description of timing aspects.
The other model parameters instead have been estimated from the descriptions in [82, 81, 80].
The results we obtain are compatible with the behaviour of lemurs, as described in the above studies.

In regard to Figure 5.3, initially all the males have the same dominance level. From the beginning and up to about 210 iterations, the group of monkeys struggles to define a clear dominance among individuals. Between iteration 210 and 390, the dotted line is not the dominant of the group, thus its position is not the closest neither to the group center nor to the female. At the end of the simulation, when the monkey with dotted line becomes the most dominant (alpha male), we can observe that it gains a central position in the group.

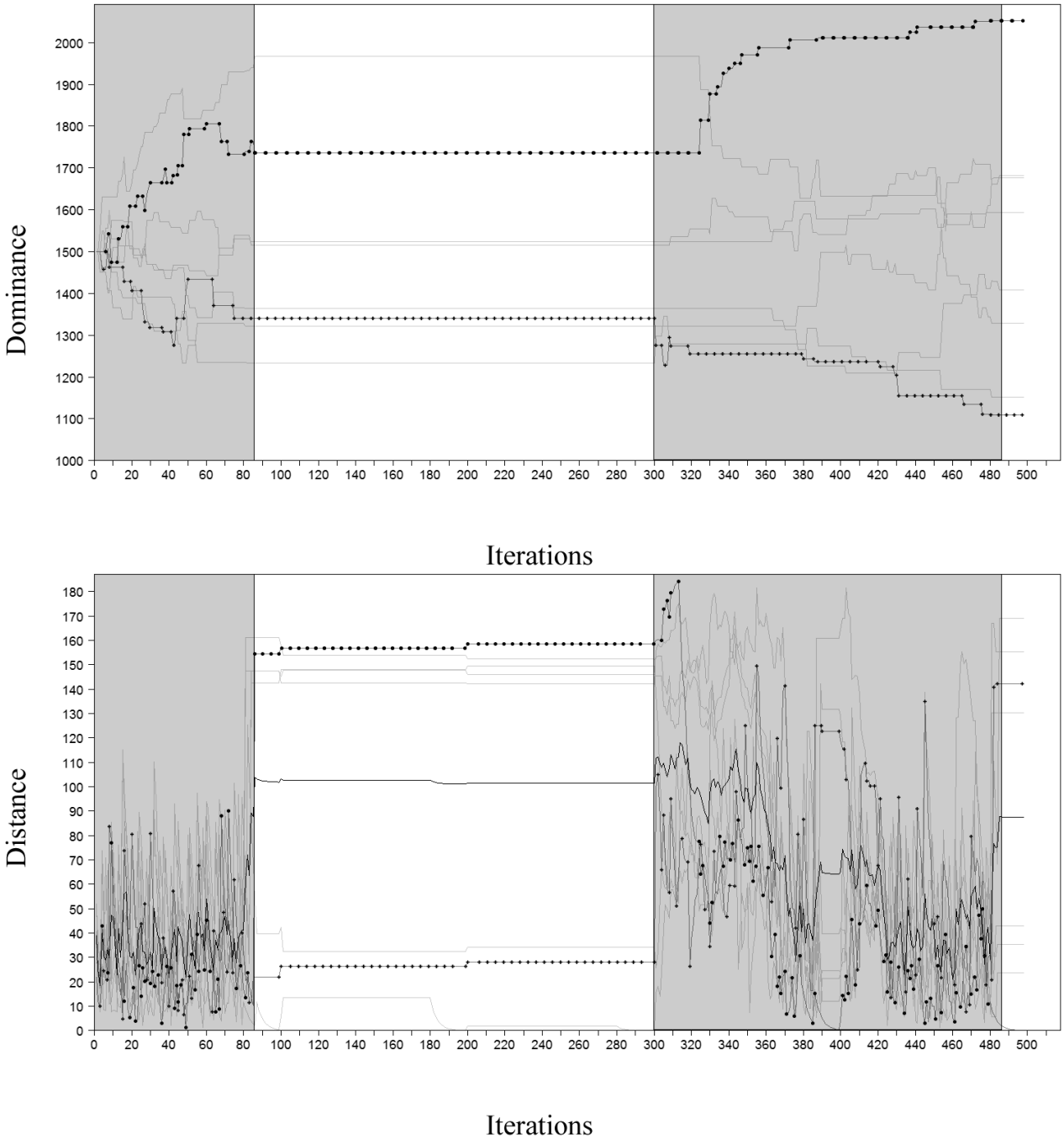
Figure 5.4 shows the dynamics of a group with a more rigid hierarchy, due to the higher level of aggressiveness. The first phase, up to iteration 90, in which the males establish a first hierarchy, is followed by a phase (up to iteration 320) in which the hierarchy becomes





$NAD = 5$	$OAD = 5$	$FT = 600$
$AN = 200$	$AO = 400$	$FSO = 12$
$CSN = 0$	$CSO = 0$	$Nfl = 1$
$FSN = 8$	$Ofl = 2$	

Figure 5.3: Simulation of an egalitarian group.



$NAD = 10$	$OAD = 10$	$FT = 400$
$AN = 350$	$AO = 700$	$FSO = 15$
$CSN = 1$	$CSO = 0$	$Nfl = 2$
$FSN = 10$	$Ofl = 4$	

Figure 5.4: Simulation of a despotic group.

very stable. From iteration 320, where the monkey with the dotted line becomes the alpha male, it gains the position that is closer to the center of the group and to the female, and it does not allow any other monkey to come close.

Normal and oestrum periods last respectively 80 and 20 iterations. In normal phase monkeys are less willing to fight and they keep a distance from each other in order to avoid unnecessary conflicts; this corresponds to the more linear parts of the graphs. In oestrum periods males have the female as the pole of attraction and they are more willing to fight. Fights can change the dominance levels of males, thus oestrum periods correspond to more struggling parts of the graphs, where, often, the ranking of dominance changes. When the dominance levels change, the topology of the group changes accordingly.

Simulations' results are compatible with the behavior of different species of prosimians as described in our reference texts [82, 81, 80]. The behavior presented by the simulated troop of prosimians is quite realistic and topological property are well emulated by the output off APP.
It is theoretically possible to use the parameters to attempt modelling other species more different than the ones we study. Difference between species often is not only in numerical values but also in the complexity of rules which describes social interactions. With sufficient data and enough facts it is theoretically possible to describe very complex social structures even like the first human ones.

5.8 Conclusion

The Attributed Probabilistic P systems (APP systems) are an extension of MPP systems in which objects are annotated with attributes.
In conclusion APP systems are intended to be used to model the dynamics of populations and ecosystems. In this context, attributes can be used to represent characteristics of the population individuals such as age, position, and so on. Apart from attributes, the feature that mainly makes a difference between APP systems and other proposals is the use of maximal parallelism for the application of rules. This feature is particularly suitable for the modelling of populations that evolve by stages (e.g. reproductive stages or stages related with seasons).

We used APP systems to model social behaviors of some species of primates as case study. In particular, as an application, we developed a model to compare despotic and egalitarian behaviors on different species of primates.
Social systems of his kind are usually approached by means of agent-based models that are often poorly documented and ambiguous. On the contrary, since both the syntax and the semantics of APP systems are formally defined, the model based on APP systems is unambiguous.
We plan to adapt our general model to model the behavior of lemurs catta by changing the values of the parameters, thus showing the flexibility of the formalism. The next step will include the possibility to explore not only the social behavior of one group of social animals but the interactions between different groups, or the multilevel social association in something more complex than a single tribe structure.

Chapter 6

Multilevel Attributed Probabilistic P systems (MAPP systems)

6.1 From APP to MAPP systems

The decision to further develop this formalism is taken when we start to study species with a social structure more complex than a simple linear hierarchy.
MPP and APP work as flat P System, with MAPPS we bring back the membrane hierarchical system of the original P System. We use membranes to model various levels of aggregation of individuals, as well as spatial or logical separation.
Two different membranes can model two different packs, two different hunting grounds, two different clans inside a community.
We must also redefine the rules semantics to ensure that they can operate not only on the usual elements, but also on the membranes, adding capabilities to create them, destroy them and change them.

Consequently the computational step cannot be parallel any longer. A membrane has to wait until all the states of contained membranes have calculated their internal states and are ready to be used, to be able to calculate its own new internal state.
We define a recursive computation which describes the environment of membranes' system starting from the inner and continues to the outer ones.
We can now see in detail that this formalism incorporates features from both MPP and APP systems, such probabilistic function attributes for elements, usage of promoters and attributes, and all the capabilities we formally defined in Chapters 4 and 5.

6.2 Introduction to MAPPS and informal definition

We give first an informal definition of MAPP systems (Multilevel Attributed Probabilistic P systems). The system is composed by one ordered **set of element of type membrane** and a **set of terminal elements** that represent reagents of the standard P systems.

To better illustrate how the model works, we will gradually introduce an example of what we have just described. When we will be done with the formalism's description and the inference rules that show its behavior, we will have a small and complete example that can show the main functionalities of MPP Systems.
The set of membranes is ordered and finite, the membranes are sorted depending on which one can incorporate the others. The first one, named *Env* can contain all the others but itself. Every following membrane can contain all those coming after it. The last in order cannot contain membrane elements but only terminal elements.

Each membrane is associated with:

- a set of domains that define the types of the attributes associated with that membrane.
- a set of rules that are applied within the membrane.
- a set of update functions in one to one correspondence with the attributes of the membranes.

Each terminal element has an associated set of domains that define the types of attributes associated with it.

Γ is a set of symbols representing the types of membranes. Terminal elements are collected in the set Σ, sets of attributes of membranes are collected in the **set** D_Γ, sets of attributes of terminal elements are collected in the **set** D_Σ. The sets of update functions of all membranes are collected together in the **set** U. Sets of rules associated with the membranes are collected in the set R.
Of course Γ, R, D_Γ and U have the same cardinality, as in one-to-one correspondence between them. The same holds for Σ and D_Σ, as they have the same cardinality.

To show how membranes and terminal elements are used, we will create an artificial example for demonstration purposes.
Let's suppose we want to model a social species competing for territory. Territory is represented by our Env membrane, and we assume that along their life our elements will never get out of the territory they live in. The territory is divided into zones, and to model them we will use a membrane-type element contained within Env. Finally our individuals gather into tribes, and to model the tribes too we will use membranes that contain the individuals.
The hierarchy of membranes is thus Env, zones, tribes. To each zone we assign one attribute, "resources", that defines how much food is available to a tribe's gatherers.
To each tribe we assign two attributes: "harvester", which stands for the number of workers/harvester within a tribe, and "warriors", which stands for the number of warriors within a tribe, and the attribute "dominant" that can assume values 0,1 and defines if a given tribe is the dominant one within the zone where it resides, or if it's not.
Individuals are modeled by the terminal element beast, and are defined by the attributes age and role; role can assume values of juvenile, harvester, warrior. We will now give a meaning to these elements' inclusions to explain how these mechanisms work.
Env membrane can contain zones, this models that each territory is divided into zones.
Env membrane can contain tribes, this models a tribe that is moving from one zone to another.
Env membrane can contain a terminal element beast, this models a single individual, cast out from a tribe which is moving from one zone to another.
A zones membrane can contain some tribes, this models the fact that the tribe is within that specific zone.
A zones membrane can contain a beast terminal, this this models a single individual, cast out from a tribe which is within that specific zone.
A tribe membrane can only contain beast terminals, this models individuals that are part of that tribe.
All other possible inclusions are prohibited by the membranes' rules.

The function ***arity()*** takes as input an element of the set Σ or the set Γ and returns

the number of its attributes (and consequently, for an element of type membrane, the number of update functions).
Elements of the system are of three types:

- a **membrane**: the membrane is described by a tuple $\langle \gamma_i, d, s \rangle$ where γ_i is an element of Γ alphabet, d is an instantiation of D_{γ_i} attributes of γ_i, s is the internal state of the membrane and describes everything that is contained in the membrane.

- a **terminal element**: the terminal element is described by a tuple $\langle \sigma_i, d \rangle$ where σ_i is a element of Σ alphabet and d is an instantiation of D_{σ_i} attributes of σ_i.

- the special element ϵ used to describe one empty internal state.

The internal state of a membrane γ_i is a multiset of these elements with only one restriction: the membranes inside the internal state of γ_i can only be composed of γ_j elements with $j > i$.
A **state of the system** ω is given by the special membrane object Env and its internal state s. In the definition of the system the **initial state of the system** ω_0 from which the computation starts is given.

Following the example in Fig.6.1 we show one possible state of the system.

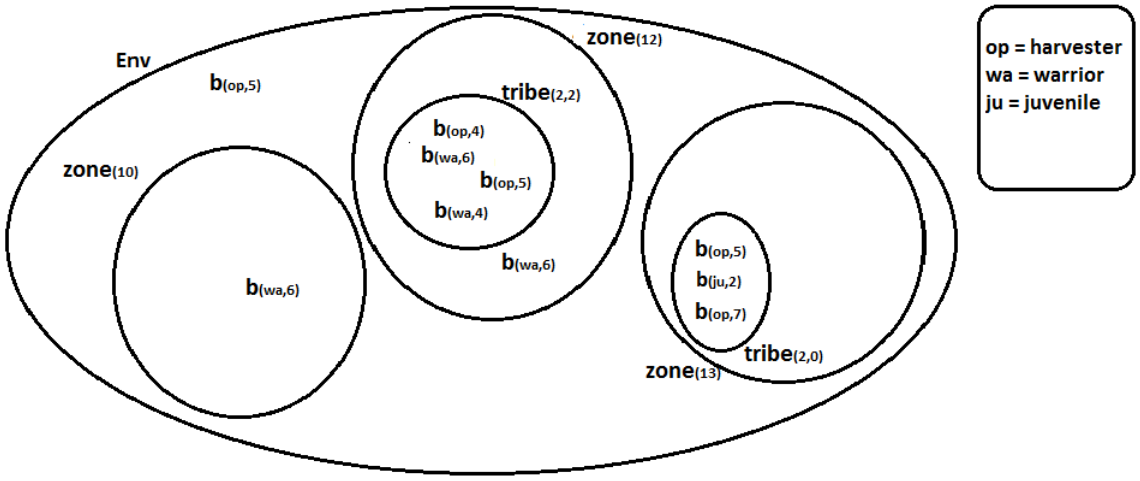


Figure 6.1: representation of the state of the system. Inside the membrane Env the system has three zone membranes and one beast terminal element. One zone membrane contains only one beast terminal element, another zone membrane contains one tribe membrane and one beast terminal element, and the last zone membrane contains one tribe membrane, and so on. Attributes' values are associated to any element, be it membrane or terminal element

A set of rules is associated to each membrane,**rules** are composed by a multiset of reagents, one of products, one of promoters, and a rating function:

- Reagents are multisets of both membranes and terminal elements. Each element is associated with an instantiation of its attributes, even a partial one, which can contain both properly typed values and variables. The multiset of reagents cannot be empty.

- Products are multisets, which can also be empty, of both membranes and terminal elements. Each attribute of elements must be associated to an expression that calculates its value. Also each element is associated to a flag which describes where the product will be created.

- The promoters are a multiset of elements possibly empty of membranes and terminals. Each element is associated with an instantiation, possibly a partial one, of its attributes which can contain both properly typed values and variables. Promoters cannot be modified by rules and are not consumed by them.

- A **rating function** takes as inputs the attributes of both promoters and reagents, and returns as an output a positive number, used as weight for the rule.

In our example, a possible environment rule, could be:
$Tribe(x, y), Zone(z) \stackrel{f(z)}{\rightarrow} Tribe(x, y) in Zone(z)$

this represents the idea that a tribe "moving along" the environemnt chooses a new location to "get into", in proportion to its resources availability.

To complete the MAPPS description for each attribute of each membrane there is an associated update function. Each function takes as input all membrane's attributes, the inner state of the membrane and returns as an output a new value for its associated attribute.

.

6.3 Multilevel Attributed Probabilistic P systems: formal definition and semantic

Definition: MAPPS is a tuple: $\langle \Gamma, \Sigma, arity(), D_\Sigma, D_\Gamma, R, U, \omega_0 \rangle$
which consists of the following elements:

- $\Gamma = \{\gamma_1, .., \gamma_n\} \cup \{Env\}$ is an ordered finite alphabet of symbols, representing the membrane elements of our system. We order each element of the set by its own position. The order we give to this set represents the property that a membrane has of being able to incorporate in its internal state only membranes with a larger index. The Env (environment) element is called skin membrane, it cannot be contained by elements of Γ while it can contain each element of Γ

- $\Sigma = \{\sigma_1, .., \sigma_m\} \cup \{\epsilon\}$ is an ordered finite alphabet of symbols, where the elements of the alphabet represent common elements that we can find into an instance of a standard P System.
  The special symbol ϵ is used to define the inner state of an empty membrane.

- ***arity()***: $\Sigma \cup \Gamma \rightarrow \mathbb{N}$ is a function which for each $a_i \in \Sigma \cup \Gamma$ gives the length of the sequence D_{a_i} that describes the attributes of a_i.

- $D_\Gamma = \{D_{\gamma_1} \dots D_{\gamma_n}\}$ Ordered set of domains, in correspondence with elements of set Γ.
  Each D_{γ_i} is a set of tuples, $D_{\gamma_i} = I_{\gamma_i,1} \times \cdots \times I_{\gamma_i,arity(\gamma_i)}$, where each $I_{\gamma_i,j}$ is a (possibly infinite) set of unstructured values. The sequence D_{γ_i} is called the set of attributes of γ_i.

- $D_\Sigma = \{D_{\sigma_1} \dots D_{\sigma_m}\}$ Ordered set of domains, in correspondence with elements of set Σ.
  Each D_{σ_i} is a set of tuples, $D_{\sigma_i} = I_{\sigma_i,1} \times \cdots \times I_{\sigma_i,arity(\sigma_i)}$, where each $I_{\sigma_i,j}$ is a (possibly infinite) set of unstructured values. The sequence D_{σ_i} is called the set of attributes of σ_i.

The generic internal state of a membrane γ_i is a multiset over the set:

$$S_i=\{\langle\gamma_j,\overline{d},\overline{s}\rangle|j>i,\gamma_j\in\Gamma,\overline{d}\in D_{\Gamma_j},\overline{s}\in S_j^*\}\cup\{\langle\sigma_j,\overline{d}\rangle|\sigma_j\in\Sigma,\overline{d}\in D_{\Sigma_j}\}$$

The generic element of the internal state of membrane γ_i is part of the resulting union of three sets.
The first set is a tuple $\langle\gamma_j,\overline{d},\overline{s}\rangle$ and describes a membrane with its attributes and its internal state, with the restriction $j > i$.
The second set is formed by the terminal symbols belonging to Σ with an instance of its attributes.
The last set consists of the singlet ϵ that is used to represent the absence of elements within a membrane.

With this definition of internal state, we define the generic state of the system $\omega = (Env, s)$ composed by a couple consisting of the symbol Env, representing the skin membrane, and its internal state s. Following the definition above we consider the element Env as γ_0.

- The **initial state** $\omega_0 = (Env, s)$ is the state of the system at the beginning of global computation. It' s represented by terminal and not-terminal items contained in the internal state of Env at iteration 0.
- The set R is a set of sets of rules, associated to the set Γ. For each element $\gamma_i \in \Gamma$ there is an associated set of rules $R_{\Gamma_i} \in R$.

Given a set of variables V, R_{Γ_i} is a finite set of evolution rules having the form:

$$u_V \xrightarrow{f} v_V|_{pr_V}$$

where $u_V \in (\{(\gamma_j, d_j, s)|\ j > i, \gamma_i \in \Gamma, d_j \in D^V_{\Gamma_j}, s \in S^{*V}_j\} \cup \{(\sigma_l, d_l)|\sigma_l \in \Sigma, d_l \in D^V_{\Sigma_l}\})^+$ is a non empty multiset of objects each one with its own attributes made explicit by values or by variables.
$pr_V \in (\{(\gamma_j, d_j, s)|\gamma_i \in \Gamma, s \in S^{*V}_j, d_j \in D^V_{\Gamma_j}, j > i\} \cup \{(\sigma_l, d_l)|\sigma_l \in \Sigma, d_l \in D^V_{\Sigma_l}\})^*$ is a possibly empty multiset of objects with its own attributes expressed by values or by variables.
Where:

$S^{*V}_j = V \cup \{\varnothing\}$
$D^V_{\Gamma_j} = (V \cup I_{\gamma_j,1}) \times \cdots \times (V \cup I_{\gamma_j,arity(\gamma_j)})$

$D^V_{\Sigma_l} = (V \cup I_{\sigma_l,1}) \times \cdots \times (V \cup I_{\sigma_l,arity(\sigma_l)})$

$v_V \in (((\{(\gamma_j, d_j, s) | \gamma_i \in \Gamma, s \in ES^{*V}_j, d_j \in E^V_{\Gamma_j}, j > i\} \cup \{(\sigma_l, d_l) | \sigma_l \in \Sigma, d_l \in E^V_{\Sigma_l}\}, \{out, \_\}) \cup$
$(\{(\gamma_k, d_k, s) | \gamma_i \in \Gamma, s \in ES^{*V}_k, d_k \in E^V_{\Gamma_k}, k > j\} \cup \{(\sigma_l, d_l) | \sigma_l \in \Sigma, d_l \in E^V_{\Sigma_l}\}, \{in(\langle \gamma_j, d, s \rangle\})))^*$
where $j > i$
is a possibly empty multiset of pairs consisting of one object with its own attributes expressed by values or functions which can take as input attributes both from u_V and pr_V, and a flag from the set $\{in(\langle \gamma_i, d, s \rangle), out, \_\}$ where $\langle \gamma_i, d, s \rangle \in u_V$. For the pairs flagged by "*out*" or "_" the membranes of the multiset v_V have an index that is greater than that of γ_i, where γ_i is the membrane that owns the rule. Otherwise for the pair flagged by $in(\langle \gamma_{j>i}, d, s \rangle)$ the membranes have an index that is greater than γ_i and γ_j.

We define:

$Vars(v_V) \subseteq Vars(u_V) \cup Vars(pr_V)$
$ES^{*V}_j = Exp(V \cup S_j)$
$E^V_{\Gamma_j} = Exp(V \cup I_{\Gamma_j(1)}) \times \cdots \times Exp(V \cup I_{\Gamma_j(arity(\gamma_j))})$
$E^V_{\Sigma_l} = Exp(V \cup I_{\sigma_l,1}) \times \cdots \times Exp(V \cup I_{\sigma_l,arity(\sigma_l)})$

where $Exp(V \cup I_j)$ denotes the set of well-typed expressions built from operators, variables V, and values of I_j. $Exp(V \cup S_j)$ denotes a well typed expression built on strings operator "+", variables V and elements and strings of set S^V_j.
To complete the definition, each rule is also associated with a rate function f typed as:
$f : S^+_j \times S^*_j \to \mathbb{R}^{\geq 0}$
the domain is composed by reagent (S^+_j) and the promoters (S^*_j), the output is a positive real number (the weight of the rules).

- The set of set of functions **Update** $U = \{U_{\gamma_1}, \ldots, U_{\gamma_n}\}$ has the same number of elements of the set Γ and its elements are in one to one correspondence with elements of Γ. Each set of functions U_{γ_i} is composed by $arity(\gamma_i)$ functions. $U_{\gamma_i} = (U_{\gamma_i,1}, \ldots, U_{\gamma_i,arity(\gamma_i)})$

Each individual set of functions U_{γ_i} consists of a number of functions equivalent to the number of attributes of the membrane associated with it, then $|U_{\gamma_i}| = arity(\gamma_i)$. The generic function $U_{\gamma_i,j}$ belonging to U_{γ_i} is typed as:

$U_{\gamma_i,j} : S^*_i \times D_{\Gamma_i} \to \quad D_{\Gamma_{i,j}}$

So, if from the set of update functions of the i-th membrane we consider the j-th update function, then this will take as input one internal state of level i, one.
The promoters are a multiset instantiation of the attribute domain of the membrane Γ_i and will give as output one value of type $D_{\Gamma_i,j}$, i.e. from the j-th set of unstructured values $I_{\gamma_i,j}$ from $D_{\Gamma_i} = \{I_{\gamma_i,1} \times \cdots \times I_{\gamma_i,j} \times \cdots \times I_{\gamma_i,arity(\gamma_i)}\}$

Given that formal definition we show the semantics of MAPPS using inference rules. We have seen that the system elements can be membranes or terminals.

We represent the structure of the internal state of Env as a tree where each node is a

membrane, each node has the membranes contained in its internal state as children. Computing the internal state of each membrane does not happen in parallel as in regular P System, but sequentially in the order induced by a postorder traversal of the tree. One by one the states of the membranes are computed back to the root of the tree represented by the Env element that concludes the complete computation of the new state of the MAPPS.

The computation of a membrane new state is managed by the single internal step rule, and it's is composed by three steps:

- Extract the elements in the set "out" of all membrane elements present in the internal state of the one we are processing

- Choose a maximal set of rules considering internal membranes as reactants ready to be used. Then apply in parallel the chosen set to the current state of the membrane.

- Apply the update functions on the inner state and attributes of the membrane to recalculate the attributes

After these three steps, the membrane has a new internal state and set of attributes. Here below we describe, one by one, the inference rules that describe the computational process of the entire system.

In the first six inference rules we describe the internal computational step of a single membrane. In order to achieve our purpose, we define three multisets: w_a which indicates the multiset of reagents ready to be used by rules, w_p the multiset of products created by the rules application, w_{out} the multiset of products created by rules with "out" flag.

The evolution of the internal state of a generic membrane γ_j is a sequence of probabilistic maximally parallel steps. We formally define the semantics of that computation as a transition relation in the style of [79]. In each step a maximal multiset of evolution rule instances is selected and applied as described by the following semantic rules:

(*single rule application*)

$$\frac{r_i{=}u\xrightarrow{k}v{\in}\overline{R}\;\; u{\subseteq}w_a\;\; K{=}\{\!|\; k'|u'\xrightarrow{k'}v'{\in}\overline{R},u'{\subseteq}w_a\;|\!\}\;\; p{=}k/\sum_{k'\in K}k'}{(w_a,w_p,w_{out})\xrightarrow{r_i,p}_{\overline{R}}(w_a{-}u,w_p{+}v,w_{out})}$$

single rule application - the rule application describes the application of a single rule that is associated with a probability p. The rule is applied starting from an internal state (w_a, w_p, w_{out}) subtracts reagents from w_a and adds products to w_p. The probability p is evaluated by normalizing the weight of the rule r_i compared to the weights of all the applicable rules of the set R_{γ_j} to reagents in w_a.

(*single rule application with the "in" flag*)

$$\frac{r_i{=}(\gamma_j,d,s){+}u\xrightarrow{k}v_{in(\gamma_j,d,s)}{\in}\overline{R}\;\; (\gamma_j,d,s){+}u{\subseteq}w_a\;\; K{=}\{|k'|(\gamma,d,s){+}u'\xrightarrow{k'}v'_{in(\gamma)}{\in}\overline{R},(\gamma,d,s){+}u'{\subseteq}w_a|\}\;\; p{=}k/\sum_{k'\in K}k'}{(w_a,w_p,w_{out})\xrightarrow{r_i,p}_{\overline{R}}(w_a{-}((\gamma_j,d,s){+}u),w_p{+}(\gamma,d,s{+}v),w_{out})}$$

single rule application with in flag - describes rules where products are sent into another membrane. The target membrane must be among the reagents. The rule definition ensures that it is not possible to send it into a membrane of higher or equal level

(*single rule application with the "out" flag*)

$$\frac{r_i{=}u\xrightarrow{k}v_{out}{\in}\overline{R}\ \ u{\subseteq}w_a\ \ K{=}\{|k'|u'^{k'}\ \ u{\subseteq}w'\ \ K{=}\{|k'|u\xrightarrow{k}v_{out}{\in}\overline{R},u'{\subseteq}w_a|\}\ \ p{=}k/\sum_{k'\in K}k'}{(w_a,w_p,w_{out})\xrightarrow{r_i,p}_{\overline{R}}(w_a{-}u,w_p,w_{out}{+}v)}$$

single rule application with'out' flag - is very similar to *single rule application*, the rule is applied starting from an internal state (w_a, w_p, w_{out}) subtracts reagents from w_a but adds products to w_{out} to be sent outside the membrane. The probability p is evaluated by normalizing the weight of the rule r_i compared to the weights of all applicable rules of the set R_{Γ_j} to reagents in w_a.
These three single rule applications are substantially similar to those of APP systems, with the difference that these sets of rules are limited to the only visible part inside a membrane j, then R_{Γ_j} is the set of rules related to the j-th element of the set Γ. K is the set of weights associated to all applicable rules. The probability is given by the weight k of a single rule divided by the sum of all the applicable weights in K.
In these rules the state of the membrane is represented by three multisets:
w_a, w_p and w_{out}.
w_a is the multiset of reagents consumed by rules, w_p is the multiset of products created by rules, w_{out} is the multiset of products created by rules and flagged by the "out" flag.
Any rules subtracts reagents from w_a and add the reagent to w_p or w_{out} depending by the flag of the rule.
(*single rule sequence rule*)

$$\frac{(w_a,w_p,w_{out})\xrightarrow{r_i,p}_{\overline{R}}(w'_a,w'_p,w'_{out})}{(w_a,w_p,w_{out})\xrightarrow{[r_i],p^+}_{\overline{R}}(w'_a,w'_p,w'_{out})}$$

single rule sequence rule - describes the application of a sequence (of rules) composed by a single rule. The sequence is associated with the probability of the single rule. The set of rules of our interest is limited to those visible within a membrane which has a type corresponding to the j-th element of Γ. The rule, conditioned by the respective probability, makes a transition from (w_a, w_p, w_{out}) to (w'_a, w'_p, w'_{out}).

(*multiple rules sequence*)

$$\frac{(w_a,w_p,w_o)\xrightarrow{r_i,p_i^+}_{\overline{R}}(w'_a,w'_p,w'_o)\qquad(w'_a,w'_p,w'_o)\xrightarrow{\overline{r},\overline{p}^+}_{\overline{R}}(w''_a,w''_p,w''_o)}{(w_a,w_p,w_o)\xrightarrow{\overline{r}@[r_i],p_i\cdot\overline{p}^+}_{\overline{R}}(w''_a,w''_p,w''_o)}$$

multiple rules sequence - describes the application of a sequence of rules using as basic case "the single rule sequence", the sequence probability is the product of rules' individual probabilities within the sequence. Our multiple rules sequence is an update of an APP systems rule. Moreover, here the set of rules is limited to R_{Γ_j}, we can see as a sequence of rules makes the system transition from (w_a, w_p, w'_{out}) to (w''_a,w''_p w''_o). In our definition $[r_i]$ denotes the sequence composed of the single element r_i, and @ denotes the concatenation of sequences.

(*single membrane internal step rule*)

$$\frac{(s,\varnothing,\varnothing)\xrightarrow{\overline{r},\overline{p}^+}_{R_{\Gamma_j}(s)_d}(w_a,w_p,w_{out})\qquad(w_a,w_p,w_{out})\nrightarrow_{R_{\Gamma_j}(s)_d-}}{(\gamma_j,d,s,o)\xrightarrow{\overline{r},\overline{p}}_{R_{\Gamma_j}}(\gamma_j,d,w_a{+}w_p,w_{out})}$$

Given a membrane internal state, s, the (single membrane internalstep rule) describes the evolution in a new membrane internal state by the $\xrightarrow{\overline{r},\ \overline{p}}_R$ relation, where $\overline{p}$ is the probability of the transition, and $\overline{r}$ is the sequence of applied rules. (single membrane internal step rule) invokes $(s,\varnothing,\varnothing)\xrightarrow{\overline{r},\ \overline{p}\ +}_{R(s)}(w_a,w_p,w_{out})$ where $R(s)_d$ is the set of applicable rules

in the state s and with membrane attribute d, with their weights, namely:

$$R(s)_{d_V} = \left\{ u_V\lambda \xrightarrow{f(s)} v_V\lambda \;\middle|\; u_V \xrightarrow{f} v_V \mid_{pr_V} \in R_{\gamma_j},\ \exists\lambda.\, u_V\lambda \subseteq s \wedge pr_V\lambda \subseteq s \right\}$$

where (i) $\lambda : V \to flat(D_{a_1}) \cup \ldots \cup flat(D_{a_n})$, with $flat(D_{a_i}) = I_{a_1} \cup \ldots \cup I_{a_{arity(a_i)}}$, for all $a_i \in \Gamma \cup \Sigma$; (ii) $u_V\lambda$, $pr_V\lambda$ and $d_V\lambda$ are well-typed multiset obtained by substituting values for variables in u_V, pr_V and d_V according to λ; and (iii) $v_V\lambda$ is the well-typed multiset obtained by evaluating the expressions in v_V under the substitution λ. Transition relation $\xrightarrow{\overline{r},\,\overline{p}}{}^{+}_{R}$ is the transitive closure of $\xrightarrow{r,\,p}_R$.

A transition $(w_a, w_p, w_{out}) \xrightarrow{r_i,\,p}_R (w_a - u, w_p + v, w_{out} + v_{out})$ corresponds to the application of a single rule. When a rule is selected, its application consists in removing its reactants from w_a and adding its products to w_p or w_{out} according to the flags of the products. The two multiset w_p and w_{out} will collect all products of all applied rules. Note that $R(s)$ takes into account that each rule is applied with respect to the weights of the rules computed in the initial state s. Moreover, $R(s)$ contains only the rules which promoters are present in the initial state s ($pr_V\lambda \subseteq s$). Once objects in w_a are such that no further rule in $R(s)$ can be applied to them, by (single membrane step rule) the new membrane is $(\gamma_j, d, w_a + w_p, w_{out})$ (where w_a are the unused objects and w_p are the new products).
(*update rule*)

$$\frac{(\gamma_j,d,s) \quad U_{\gamma_j,1}(d,s) \longrightarrow d'_1, \ldots, U_{\gamma_j,arity(\gamma_j)}(d,s) \longrightarrow d'_{arity(\gamma_j)}}{(\gamma_j,d,s,o) \xrightarrow{update(\gamma_j)} (\gamma_j,(d'_1,\ldots,d'_{arity(\gamma_j)}),s,o)}$$

$d = (d_1, \ldots d_{arity(\gamma_j)})$
The update rule is a feature not present in APPS. Here we obtain a new domain of attributes d' from previous attributes and the state of γ_j, thanks to a set of functions that change the value of each attribute.

(*getting on rule*)

$$\frac{s' = s - \uplus_{(\overline{\gamma},\overline{d},\overline{s},\overline{o}) \in s} (\overline{\gamma},\overline{d},\overline{s},\overline{o}) + \uplus_{(\overline{\gamma},\overline{d},\overline{s},\overline{o}) \in s} (\overline{\gamma},\overline{d},\overline{s},\varnothing) + \uplus_{(\overline{\gamma},\overline{d},\overline{s},\overline{o}) \in s} \overline{o}}{(\gamma_j,d,s,o) \xrightarrow{gettingon(\gamma)} (\gamma_j,d,s',o)}$$

The (getting on rule) applied on a target membrane searches all membranes in its internal state, and then, for each membrane found, keeps the elements present in o and adds them to the internal state (s) of the target membrane. Moreover, this rule makes o empty for each internal membrane. The new state of the target membrane s' is obtained subtracting the union of all internal membranes $[- \uplus_{(\overline{\gamma},\overline{d},\overline{s},\overline{o}) \in s} (\overline{\gamma},\overline{d},\overline{s},\overline{o})]$, adding the union of all membranes where o has been emptied $[+ \uplus_{(\overline{\gamma},\overline{d},\overline{s},\overline{o}) \in s} (\overline{\gamma},\overline{d},\overline{s},\varnothing)]$and adding the union of all elements taken from the o multiset of internal membranes $[+ \uplus_{(\overline{\gamma},\overline{d},\overline{s},\overline{o}) \in s} \overline{o}]$.

(*single membrane step rule*)

$$\frac{\begin{array}{c}(\gamma_j,d,s,\varnothing) \xrightarrow{gettingon(\gamma_j)} (\gamma_j,d,s',\varnothing) \\ (\gamma_j,d,s',\varnothing) \xrightarrow{\overline{r},\overline{p}}_{R_{\Gamma_j}} (\gamma_j,d,s'',o) \\ (\gamma_j,d,s'',o) \xrightarrow{update(d,s'')_j} (\gamma_j,d',s'',o)\end{array}}{(\gamma_j,d,s,\varnothing) \xrightarrow{\overline{r},\overline{p},update(d,s')_j} (\gamma_j,d',s'',o)}$$

The single membrane step rule describes a composition of the getting on rules, an internal computational step and the subsequent application of update functions.

These three steps are performed in the correct order on any membrane element.
For the membrane (γ_j, d, s) first we search in all membranes elements inside the internal state s, for any of this we take the elements in $w_o ut$ and add them to s as described in the getting on rules. This give us a new internal state s', then form (γ_j, d, s') the full single membrane internal step takes place. Similar to a computation of one APP systems (enriched by rules over membrane elements) the internal step affect the internal state of the membrane leading from (γ_j, d, s') to (γ_j, d, s''), s'' is the new internal state computed by the single membrane internal step rule (and all rules before that one).
After the new internal step is computed, the calculation of the attributess new values take place. Any attributes $(d_1, \ldots, d_{arity(\gamma_j)})$ will be used togheter with the internal state s'' as argument for the $arity(\gamma_j)$ functions in U_j, the new attributes values will be $d' = (U_1(d, s), \ldots, U_{arity(\gamma_j)}(d, s))$ as described by update rule.

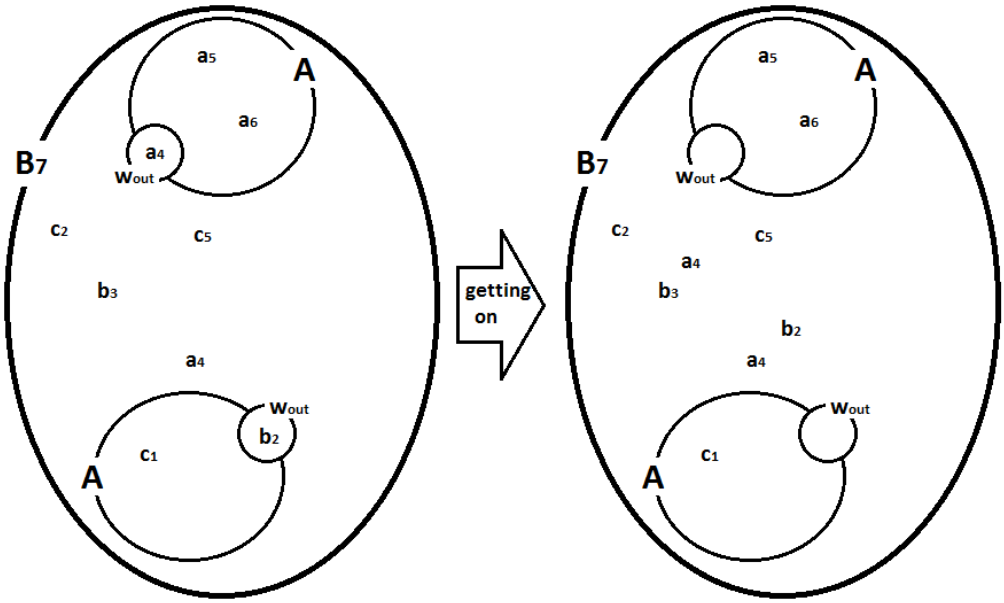


Figure 6.2: In this example we follow the computation of membrane B, first all elements in w_{out} multiset of membranes A inside membrane B are collected and added to the internal state of B (getting on rule)

At the end of this three steps the transition is complete and the membrane object will have a new internal state, updated attribute values and a new multiset of elements o. The membrane is now ready to be eventually used in upper computational step.
Figure 6.2, 6.3 and 6.4 shows one easy example of this part of the process.
After defined how a single membrane compute a step the next three rule manage the computation of the entire system by some recursive rules.

(*recursive membrane computational step rule*)

(1)

$$\frac{\sigma \in \Sigma \; d \in D_\Sigma}{(\sigma, d) \Rrightarrow (\sigma, d)}$$

(2)

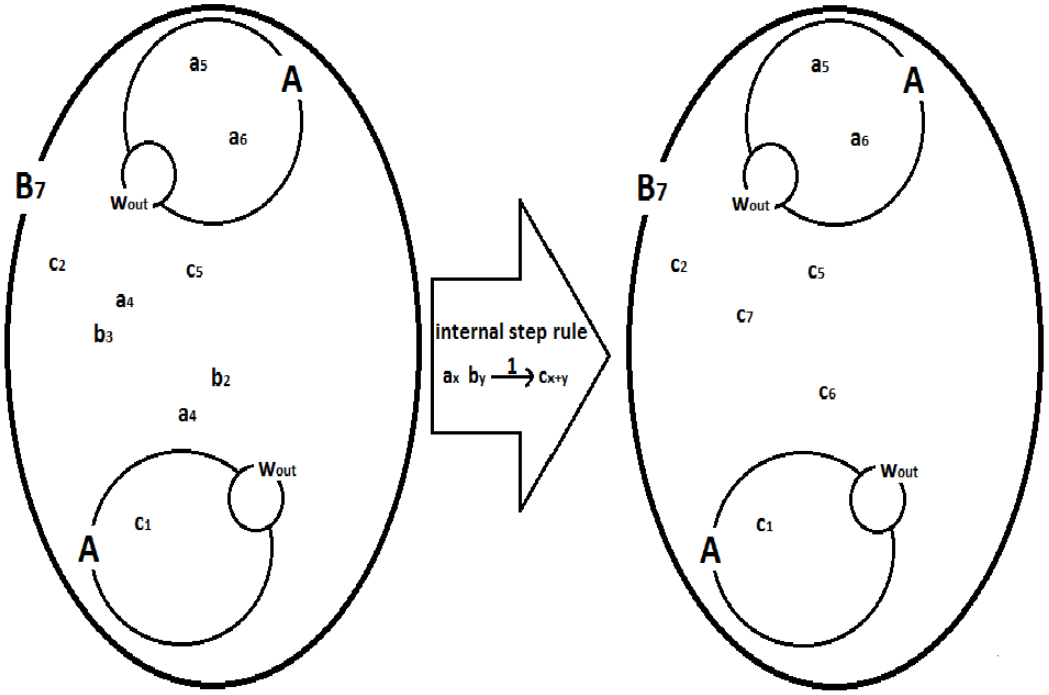


Figure 6.3: Then rules are selected and applied inside the membrane B, in this example we only have one rule, $R_B = \{a_x, b_y \overset{1}{\rightarrow} c_{x+y}\ \}$ (single membrane internal step rule)

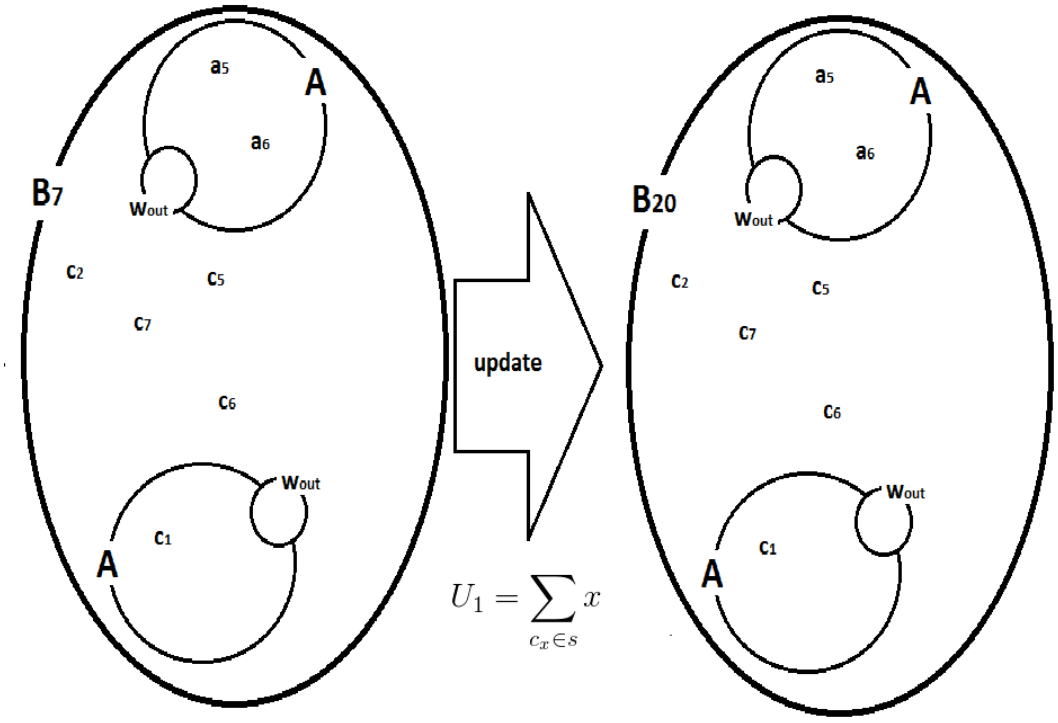


Figure 6.4: Finally, to complete the step the attribute of B is updated. B has only one attribute so the set U_B has only one function $B_1 = \sum_{c_x \in s} x$ thus the value 7 is changed in 20 (update rule)

$$\frac{s = x_1.x_2.\ldots.x_l \quad \forall i, 1 < i < l, x_i \Rrightarrow \overline{x}_i \qquad \overline{s} = \overline{x}_1.\overline{x}_2.\ldots.\overline{x}_m \quad (\gamma_j, d, \overline{s}, o) \overset{\overline{r},\overline{p},update(d,s')_j}{\Longrightarrow} (\gamma_j, d^*, s^*, o^*)}{(\gamma_j,d,s,o)\Rrightarrow(\gamma_j,d*,s*,o*)}$$

(3)

$$\frac{s = x_1.x_2.\ldots.x_l \quad \forall i, 1 < i < l, x_i \Rrightarrow \overline{x}_i \quad \overline{s} = \overline{x}_1.\overline{x}_2.\ldots.\overline{x}_m \quad (\gamma_j, \varnothing, \overline{s}, \varnothing) \overset{gettingon(\gamma_j)}{\longrightarrow} (\gamma_j, \varnothing, \overline{s}', \varnothing) \quad (\gamma_j, \varnothing, \overline{s}', \varnothing) \overset{\overline{r},\overline{p}}{\longrightarrow}_{R_{\Gamma_j}} (\gamma_j, \varnothing, s^*, o)}{(Env,s)\Rrightarrow(Env,s^*)}$$

where

$$x_i = \{\langle \gamma_k, \overline{d}, \overline{s}, o\rangle | k > i, \gamma_k \in \Gamma, \overline{d} \in D_{\Gamma_k}, \overline{s} \in S_k^*, o \in S_k^*\} \cup \{\langle \sigma_k, \overline{d}\rangle | \sigma_k \in \Sigma, \overline{d} \in D_{\Sigma_k}\}$$

These three rules are related to the recursive computation of all membranes in the system.
The rule (3) says how the whole system completes a full computational step. The Enviroment passes from state s to new state s^* with computation $(Env, s) \Rrightarrow (Env, s^*)$. To do that recursivly the operator $\Rrightarrow$ is applied to all objects in s, if the object is a terminal element $\sigma \in \Sigma$ the rule (1) tells that the terminal elements are unaltered by the operator $\Rrightarrow$, if the object is a membrane element the rule (2) recursively calls both rules (1) and (2) on all the elements of the internal state of the membrane, and so on.

In other words the three recursive membrane computational step rules are applied to all elements of the system: membranes represented by (γ, d, s, o), terminal elements by (σ, d), Env represented by (Env, s). The (1) tells us that the terminal elements are unaltered by this transformation. The (1) represents the ground rule of the recursion. The (2) is applied to membranes. The rule (2) is recursively called to each element (x_i) of the internal state (s) of the membrane to obtain a new state $\overline{s}$ and then the single membrane step rule is applied over the tuple $(\gamma_j, d, \overline{s}, o)$ to obtain $(\gamma_j, d^*, \overline{s}^*, o^*)$.
The (3) describes the applyng of the rule to Env to obtain a new step of the entire system. Instead of applying a full single membrane step rule to Env, the (3) applies to Env only the getting on and the single membrane internal step rule. The update rule is not required because Env has no attributes. In both rule (2) and (3) the states s and $\overline{s}$ are represented by sequence of x_i elements. Each x_i can be a membrane represented by a tuple $\langle \gamma_k, \overline{d}, \overline{s}\rangle$ or a terminal element represented by $\langle \sigma_k, \overline{d}\rangle$.

In the description of these rules we use the following sets, operator and conventions:
$R_{\gamma_n}(s) = \{u_V\phi \overset{f(s)}{\rightarrow} v_V\phi | u_V \overset{f}{\rightarrow} v_V|_{pr_V} \in R_{\gamma_n}(\phi) \exists \phi. u_V\phi \subseteq s \wedge pr_V\phi \subseteq s\}$

where $\phi : V \rightarrow flat(D_{a_1} \cup \cdots \cup flat(D_{a_n})$, with $flat(D_{a_i}) = I_{a1} \cup \cdots \cup I_{arity(a_i)}$, for all $a_i \in A$; $u_V\phi (u_V \in \Sigma_V^*)$ is the well-typed multiset obtained by substituting values for variables in u_V according to ϕ; and $(v_V \in \Sigma_{EV}^*)$

6.4 A simple example

Before going deeper into the main case study for the MAPP System we want to show a simple example. With this example we want to give an idea about both the formalization of a model and how computational steps take place one after another.

We consider a basic model about wolves. We will model a few facts:
- male and female wolves live in packs.
- packs keep a hunting territory.
- packs hunt as one and share the food
- inside a pack some wolves fight to assess dominance
- as result of a defeat a wolf can walk away from the pack and become a lone wolf
- new pups are born, wolves die.

Now we define the formal elements of the MPP systems to model this facts:

- Γ is the set of terminal elements. We consider MaleWolf, FemaleWolf and Prey as main elements. Later we will add some control objects to manage the different phases of the simulation.
$\Gamma = \{MaleWolf, FemaleWolf, Prey\} \cup ControlObjects$

- Σ is the set of membrane elements ordered by inclusion. We have one generic Environment which contains HuntingGround(s) which can contain Pack(s). The order of the set Σ allows Packs to be directly inside both Environment and HuntingGrounds. Elements such a lone MaleWolf can exist outside the pack and be alone inside one HuntingGround. What is forbidden by the order is, for example, that a HuntingGround is inside a Pack or a Pack is inside another Pack.
$\Sigma = \{Enviroment, HuntinGround, Pack\}$

- D_Γ is the set of set of attributes of elements of Γ. We take as attributes for MaleWolf "age" and "dominance", for FemaleWolf we take "age", for Prey we take the attribute "size".
$D_\Gamma = \{D_{MaleWolf}, D_{FemaleWolf}, D_{Prey}\} \cup D_{ControlObjects}$

- D_Σ is the set of attributes associated to membrane elements. To Packs membranes we associated attributes "health" and "strength" and "size", to "HuntingGround" we associate the attribute "wealth" referred to how many preys are in the membrane and a "carrying capacity" as a maximum level of wealth you can have in that territory. The Environment membrane doesn't really need attributes in first instance.
$D_\Sigma = \{D_{HuntingGround}, D_{Pack}\}$

- R is the set of sets of rules associated to each membrane, so we will have a set of rules associated to the Environment membrane, one set associated to the HuntingGround membrane and a set of rules associated to the Pack membrane.
$R = \{R_{Env}, R_{HuntingGround}, R_{Pack}\}$
Later in this example we will show rules inside each set and how they apply to the system.

- U is the set of update rules associated to attributes of membrane elements. $U = \{U_{HuntingGround}, U_{Pack}\}$. The set $U_{HuntingGround}$ contains two functions, one to update the attribute "wealth" and one to update the attribute "carrying capacity", the set U_{Pack} contains three functions, one to update the attribute "health", one for "strength" and one for "size".

- ω_0 describes one generic initial state of the models.The state of the System is described by the internal state of each membranes of the system. For each element of these internal states attributes value must be defined.

After this first formalization we still have to define the model core: the rules set. To manage a functional subset of sets of rules we will add control objects to the Γ alphabet, the control objects will implicitly define the simulation phases.

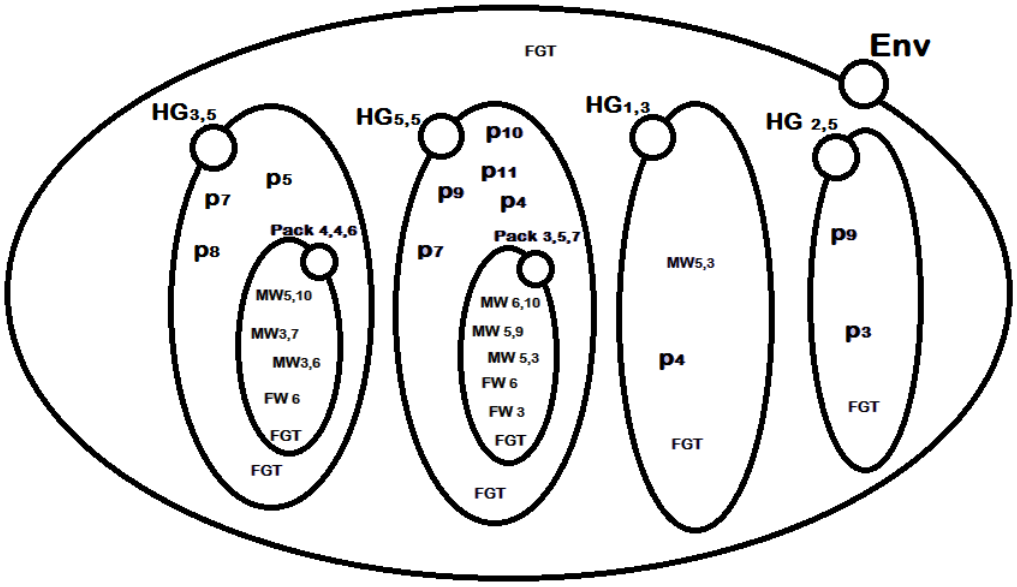


Figure 6.5: Example of wolf model with four Hunting Grounds and two Packs, also a Lone Wolf in the third Hunting ground. The system is in FGT phase for the presence of FGT control object in all membranes.

Now we have a skeleton of our model. We know what the model is about and what is the level of aggregation we are going to suggest. What set of rules we need and the attributes of both terminal and membranes elements. The next modeling steps we are taking consists in:
- deciding which control objects is necessary to introduce so we can shape different computation phases. It is important to introduce enough control objects to keep the phases between membranes synchronized.
- deciding what kind of facts (events) we will model with one or more rules. Usually we have a rule for any possible outcome of an event.
- deciding to which set of rules each rule belongs, depending on the reagents involved in the rule.
- deciding which facts (rules) are in competition which each other and fall in the same computation phase.

We start with the rules for Pack membrane. Pack membrane is the last of the ordered set Σ so inside this membranes we can only have terminals from set Γ and control objects. Rules in R_{Pack} can have as reagents and products only elements of set Γ and control objects, as promoter any rule can have the membrane Pack too with the associated attributes. We decide to have three computation phases. In the first one (which we call "reproduction") one single FemaleWolf give birth to new offspring. In the phase "fight" MaleWolf(s) fight to asses dominance over the pack. In the phase "survival" each wolf survives according to the food hunted by the pack and the age of the individual.
For each of this phases we add to the alphabet Γ one associated control object $\{REPR, FGT, SURV\}$.

Rules in R_{Pack} will have one of this control objects as promoters to implements the division in subsets associated to the different phases.
Also we will add rules to R_{Pack} to manage control objects and define the succession of different phases.
Now we decide, phase by phase, which rules model the facts that belong to that phase:
REPR phase - in this phase we add one rule for any possible outcome of the fact "only one female (the mate of the packleader) gives birth to new offspring". We describe the rules which have as outcome one, two, three or more puppies of both sexes. We also model the fact that the mother dies or not, depending on the age and the availability of food. All these rules have as promoter the element REPR.
For example the rule which gives as outcome two male puppies has the following form:

$$FemaleWolf_{agef>3} \overset{f(agef,h)}{\rightarrow} FemaleWolf_{agef}, MaleWolf_{0,0}, MaleWolf_{0,0} | REPR, Pack_{h,\_,\_}$$

if we analyze this rule we read:

- we include the Pack membrane as promoter only to access to the attribute h of the Pack where the rule happen
- with a rate $f(agef, h)$ depending on the age of a FemaleWolf and the health of the pack (the parameter h)...
- we apply the rule first, consuming one element FemaleWolf with age attribute at least 4 from the membrane. Then we create and put back the FemaleWolf with the same age and add two elements MaleWolf with age and dominance 0 to model the two offspring.

If we use the reproduction rule in this form each female in the pack will have a reproduction phase because the control objects REPR is used as promoter and enables the reproduction rule for each FemaleWolf elements. But we want only one to reproduce during any REPR season. To model this we use REPR object not as promoter but as reagent, in this way the REPR control object will enable the reproduction rule just one time. The rule now is:

$$FemaleWolf_{agef>3}, REPR \overset{f(agef,h)}{\rightarrow} FemaleWolf_{agef}, MaleWolf_{0,0}, MaleWolf_{0,0}, FGT | Pack_{h,\_,\_}$$

Now the control object REPR is consumed by the rule allowing only one female to reproduce. We add the control object FGT to model the switch from reproduction phase to the fight phase.

FGT phase - in this phase wolves fight for dominance with rules similar to the case study for APP systems. Any of the followings outcomes is modelled by one rule with associated one rate function in accordance with the data collected by empiric observations. The outcomes can be:
- Two males face-off, the less dominant walks away, no fight happens
- Two males face-off, the most dominant wins, his dominance is raised by little, the dominance of the loser goes down a little
- Two males face-off, the less dominant wins, his dominance raise a lot, the dominance of the loser goes down a lot
- Two males face-off, the less dominant wins, his dominance is raised a lot, the loser walks away from the pack to become a lone wolf

Each rule has a rate function depending on age and dominance of both the males involved in the face-off $f(age1, age2, dom1, dom2)$, all these rules have as promoter the control object FGT, in the end we add the rule to manage the control object

$FGT \rightarrow SURV|FGT$

SURV phase - in this phase each wolf is tested to model if it survives (aging) or dies. The two outcomes are in competition over each element of the Pack with two functions $f(age, dom, h, s)$ and $g(age, dom, h, s)$ where age is the age of the wolf, dominance is the dominance (for females this is a constant), h and s are the health and the size attributes of the pack.
For example the rules for the male wolf are:
$MaleWolf_{age,dom} \overset{f(age,dom,h,s)}{\rightarrow} MaleWolf_{age+1,dom}|SURV, Pack_{h,\_,ss}$
$MaleWolf_{age,dom} \overset{g(age,dom,h,s)}{\rightarrow} \_|SURV, Pack_{h,\_,s}$

The first rule "consumes" the wolf tested and brings it back in the pack, aged by one. The second rule "consumes" the wolf and nothing comes back, the wolf has not survived the test. We already have seen this kind of rules in the case study of MMP systems with frogs. Again to complete the set of rules of the phase we add one rule to manage the control object
$SURV \rightarrow REPR|SURV$

These three phases model what happens inside the pack. The female of the alpha gives birth to new offspring, males fight for supremacy and each year some wolves age to live another year and some others die.
To complete the definition of the Pack membrane we describe the set U_{Pack} which contains three functions, one to update the attribute "health" (constant), one for the attribute "strength" (function of the number of the members of the pack considering their ages) and one for the attribute "size" (the number of the members of the pack).

We now model what happens inside the "HuntingGround" membranes. Inside this membrane we can find elements of set Γ and elements "Pack" of set Σ. According to what happens inside the Pack we divide the computation in three phases associated to the control objects REPR, FGT and SURV.
In the Hunting Ground membrane we want to model packs which hunt together, packs which fight with other packs for the territory, reproduction of prey, packs which drive away a lone wolf.
As we have done for the Pack membrane, we subdivide the facts we want to model into the differents phases associated to control objects.

REPR phase in this phase any prey present in the membrane duplicates (or more). It is a very simple way to model the reproduction of the prey but for our basic model it is enough. The limit about how much prey is contained in a HuntingGround membrane is given by the attribute "carrying capacity" of that membrane, the current number of preys is given by the attribute "wealth". We have a rule like:
$prey_{size} \overset{f(w,cc)}{\rightarrow} prey_{size}, prey_1, prey_1|REPR, HuntingGround_{w,cc}$

The function $f(w, cc)$ when w is near the carrying capacity give, to the reproduction rule, a very low rate near to zero, otherwise it gives a high chance of reproduction to any prey in the hunting ground zone.
Again to complete the subset of rules of the phase we add :
$REPR \rightarrow FGT|REPR$
to manage the shift from REPR phase to FGT phase.

FGT phase - in this phase we want to model the following facts outcome:
- two packs face off for the territory, the strongest wins and the weakest is cast out of the Hunting Ground, both sides lose some health
- two packs face off for the territory, the weakest win and the strongest is cast out the Hunting Ground, both sides lose some health
- two packs face off for the territory, the weakest walks away without a fight
- a pack finds a lone wolf in the territory and drives him away
- a pack tolerates the presence of one lone wolf in the territory

For any fact we want to model, we associate each possible outcome to a rule. Like for the rule which models the fight between wolves, we create rules for the fight between packs, but here the losers always leave the ground and is not dominance but health which is lost in the process.

$FGT \rightarrow SURV|FGT$
completes the subset relative to the phase FGT.

SURV phase in this phase packs and lone wolves hunt over prey defining if they survive or not. Rules are structured like:

$$prey_s, Pack_{h,st,si} \overset{f(s,st,si)}{\rightarrow} Pack_{h+s,st,si}|SURV$$

preys are hunted by packs and the health of the pack is increased. The bigger is the prey and the bigger is the increase of the health of the pack which hunt it. We add the usual rule about control object
$SURV \rightarrow REPR|SURV$

To complete the description of the HuntingGround we define the update function for the attribute "wealth" of the membrane. The function simply counts the number of preys in the membrane, then updates the value of the attribute. The attribute "carrying capacity" is constant.
In conclusion, inside the Hunting Grounds packs fight each other to defend the territory, hunt to share the prey with the whole pack, and the prey reproduces taking in account the land resources (the carrying capacity).

To complete the model we define the R_{Env} set, the set of rules about the single membrane Environment which can contain elements from Γ as well as "Pack" and "HuntingGround" elements from the set Σ.
If a pack (or a lone wolf) is in the Environment membrane this means the pack was just kicked out from some Hunting Ground. The pack wanders looking for a new Hunting Ground to claim. The fact we want to model is that the pack moves to another Hunting Ground, and we do that with a rule like this:

$$Pack, HuntinGround_{w,cc} \overset{f(w)}{\rightarrow} Pack_{w_0,\_} \; in \; HuntinGround_{w,cc}$$

the function $f(w)$ makes the Hunting Ground with more prey the prefered choice for a pack looking for food. Also if there is a lot of prey it is more liklely that there is not another pack hunting in the location.
At the Environment membrane level we just model packs and lone wolves kicked out and looking for a new territory. This completes the description of the model.
Now the model is complete. Running it we can discover that we need a rule to describe

some additional facts, for example:
- if there is no more prey left in one hunting ground the pack who lives there will move away looking for another hunting ground
- a lone wolf can join another pack
- if one Pack is too big it is possible it will split in two packs and so on...

It is not hard to add facts and their associated rules to the model, in order to make it more accurate. The correct analysis of the phenomena will lead the decision about . We can decide we want the FGT phase last more than one computational step or add more Hunting Grounds after a first instance of the model. With our formalism all this little changes are easy to put in the model.

6.5 Case Study: Modelling social life of Serengeti Lions

Lions are the only felines that are truly social, living in prides and coalitions, the size and dynamics of which are determined by an intricate balance of evolutionary costs and benefits. Why has social behaviour, lacking in other cats, become so important to them? Is it a necessary adaptation for hunting large prey such as wildebeest? Does it facilitate the defence of young cubs? Has it arisen from the imperatives of competing for territory? As details of leonine sociality have emerged, mostly over the past 40 years, many of the key revelations have come from a continuous study of lions within a single ecosystem: the Serengeti.

As we can read in [68], [67], [66] and [65], Serengeti National Park encompasses 5,700 square miles of grassy plains and woodlands near the northern border of Tanzania. The park had its origin as a smaller game reserve under the British colonial government in the 1920s and was established formally in 1951. The greater ecosystem, within which vast herds of wildebeest, zebra, and gazelle migrate seasonally, following the rains to fresh grass, includes several game reserves (designated for hunting) along the park's western edge, other lands under mixed management regimes (including the Ngorongoro Conservation Area) along the east, and a trans-boundary extension (the Masai Mara National Reserve) in Kenya. In addition to the migratory herds, there are populations of mice, reedbuck, waterbuck, eland, impalas, buffalo, warthogs, and other herbivores living less peripatetic lives. Nowhere else in Africa supports quite such a concentrated abundance of hoofed meat, amid such an open landscape, and therefore the Serengeti is an important place for lions and an ideal site for lion researchers.

For this animals, life is hard and precarious, and casualties are numerous. For them as well as for their prey, life spans tend to be short, more often terminating abruptly than in graceful decline. An adult male lion, if he's lucky and durable, might attain the advanced age of 12 in the wild. Adult females can live longer, even up to 19. A typical case of mortality in Serengeti is caused by Masai farmers and shepherds who identify the lions as source of danger to their herds and their crops, as we can see in [73]. Life expectancy at birth is much lower, for a lion, if you consider the high mortality among cubs, half of which die before age two. But surviving to adulthood is no guarantee of a peaceful demise. Continual risk of death, even more than the ability to cause it, is what shapes the social behaviour of this ferocious but ever jeopardized animal.

Male lions, not strictly belonging to any pride, instead form coalitions with other males and exert controlling interest over a pride, fathering the cubs and becoming resident, loosely

associated with the pride as we can see in [77]. They also play an important role in helping kill prey, especially with larger and more dangerous animals, such as cape buffalo or hippos, thereby contributing something besides sperm and protection to the life of the pride as we can see in [74].

Usually lion coalitions cast a challenge for the controlling rights of a pride. In this situation the roars play an important role, as they serve to indicate to their opponents their numerical strength. In some cases, conflicts between lions finish even before beginning, when a group of lions realizes it is inferior in size to the other group [70]. If a coalition of males takes over, it will kill the young of their rivals to bring the females quickly back into heat. Mostly lions die because they kill each other: the number one cause of death for lions, in an undisturbed environment, is other lions. At least 25 percent of cub loss is owed to infanticide by incoming males. Females too, given the chance, will sometimes kill cubs from neighbouring prides. They will even kill another adult female, if she unwisely wanders into their ambit. Resources are limited, prides are territorial: a lot of bite wounds visible on lions reflect the competitive struggle for food, territory, reproductive success and sheer survival.

As we can see in [69], it is not just the need for a joint effort in making and defending kills, that drives lionesses to live in prides. It's also the need to protect offspring and retain those premium territories. Although pride size varies widely, from just one adult female to as many as 18, prides in the middle range succeed best at protecting their cubs and maintaining their territorial tenure. Prides that are too small tend to lose cubs. Periods of oestrus for the adult females often are synchronized especially if an episode of male infanticide has killed off all their youngs and resetted their clocks so that cubs of different mothers are born at about the same time. This allows the formation of créches, lion nursing groups in which females suckle and protect not just their own cubs but others too. Such cooperative mothering, efficient in itself, is further encouraged by the fact that the females of a pride are related as mothers and daughters and sisters and aunts, sharing a genetic interest in one another's reproductive success. But prides that are too large do poorly also, because of excessive intra-pride competition. A pride of two to six adult females seems to be optimal on the plains as we can see in [71].

Male coalition size is governed by similar logic. Coalitions are formed, typically, among young males who have outgrown the natal pride and gone off together to cope with adulthood. One pair of brothers may team with another pair, their half-siblings or cousins, or even with unrelated individuals that turn up, solitary, nomadic, and needing partnership. Put too many such males together as a roving posse, each hungry for food and for chances to mate, and you have craziness. But a lone male, or a coalition that's too small will also suffer disadvantages.

As we can see in this little preamble, life is very hard for a lion, but more interesting for a realization of a model based on MAPPS. Because of the many challenges, life for pride proves to be really complex. We tried to put the total amount of problems faced by lions during their lifetime into this model, and we formalized MAPP systems as the main tool to help us in this purpose. Here below we present a formal definition of our model for Serengeti lions, taken as example because the most studied and known pride of lions live in Serengeti.

6.6 Modelling the Lion Model with MAPPS

The model of the Serengeti Lions has as membrane elements:

1. Environment: the place of the simulation, where the hunting territories controlled by pride are contained
2. Hunting ground: Locations where there is a pride, and may be subjected to invasion by coalitions of stranger males
3. Pride: the largest social organization of lions consists of several females with their cubs, young members and a small number of males
4. Court: group of females kindred with each other by matrilineality, and their young or cubs
5. Coalition: group of males kindred with each other, which are part of a pride or wander in search of a pride to take over
6. Family: a female with cubs under 18 months
7. Subcourt: a family of cubs between 18 months and 4 years, differs from the family, because, in the case of a takeover of pride, cubs are not killed by the new males
8. Subcoaltion: a group of male cubs between 18 months and 4 years within the pride, differs from the family, because, in the case of a pride's takeover, cubs are not killed by new males

Terminal elements of simulation are:

1. male: age, health
2. female: age, health
3. cube: resources: health, casualty

In addition to those above, we provided a collection of control objects, to manage the events and seasonal cycles of lions' life.

The simulation of a year of life of the lions proceedes as follows:

1. a finite number of hunting iterations under favourable conditions:
   in these the pride seeks to acquire prey, individual survival is tested as a function of the preys captured, during the phases of hunting a shortage of prey can cause divisions
2. take over
   at this stage a coalition of nomadic males tries to chase away the previous residents and replace them, if they succeed all the little lions are killed to make the females fertile and available again.
3. A mating season
   where the lions of the coalition within the pride mate with all adult females who don't have cubs under the age of 18 months
4. a finite number of hunting iterations under adverse conditions:
   even in this phase, the pride seeks to acquire prey, individual survival is tested as a function of the captured prey, so in this case the shortage of food is greater, and this favours internal divisions

5. Births and ageing At the end of the season of adversities and with the beginning of the abundance season, new cubs are born and new families are formed within cohorts and each lion goes through the ageing rule

In our simulation we also manage events like:

1. creation of coalitions formed by unrelated peripatetic males
2. Expulsion of members from pride
3. Expulsion of members from location
4. Casualties during hunting phase

A first draft of this model is interpolated by the study of the following paper and some videos from national geographic too. A second version was developed under counselling of our supervisors and interviewing friendly biologists. After that, we saw we could add complexity to the system implementing many other features, but we have tried to capture only the most important events, leaving out what would occur on too specific conditions, or as consequence of sporadic events, or what would be of little interest to the simulation.

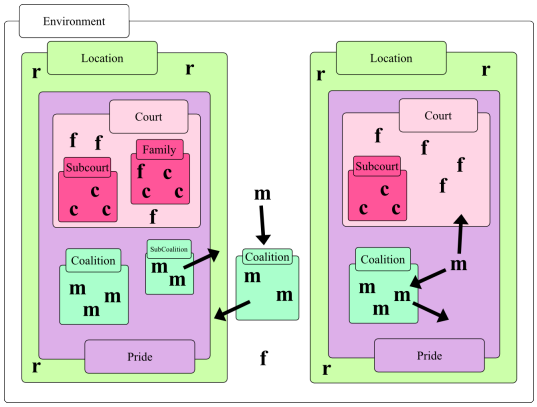


Figure 6.6: initial state ω_2

As we see in the picture above, our model is composed by eight membranes and five elements. As shown by arrows, our items can pass from a membrane to another. A male lion, for example, can leave its coalition, go to pride membrane, and, from there, enter the court in order to mate with the lionesses of that court. A male in the environment can also join a coalition in order to find allies, with the aim of trying a take over in the future. Also a membrane can move from a parent membrane to another. For instance, a coalition of males can enter into a pride, fight with the resident coalition and stay there, driving out the loser coalition. A membrane could also change into another one, copying its state. For example, a Family when it grows enough turns into court.
We give now the formal definition of the model with all the rules set:

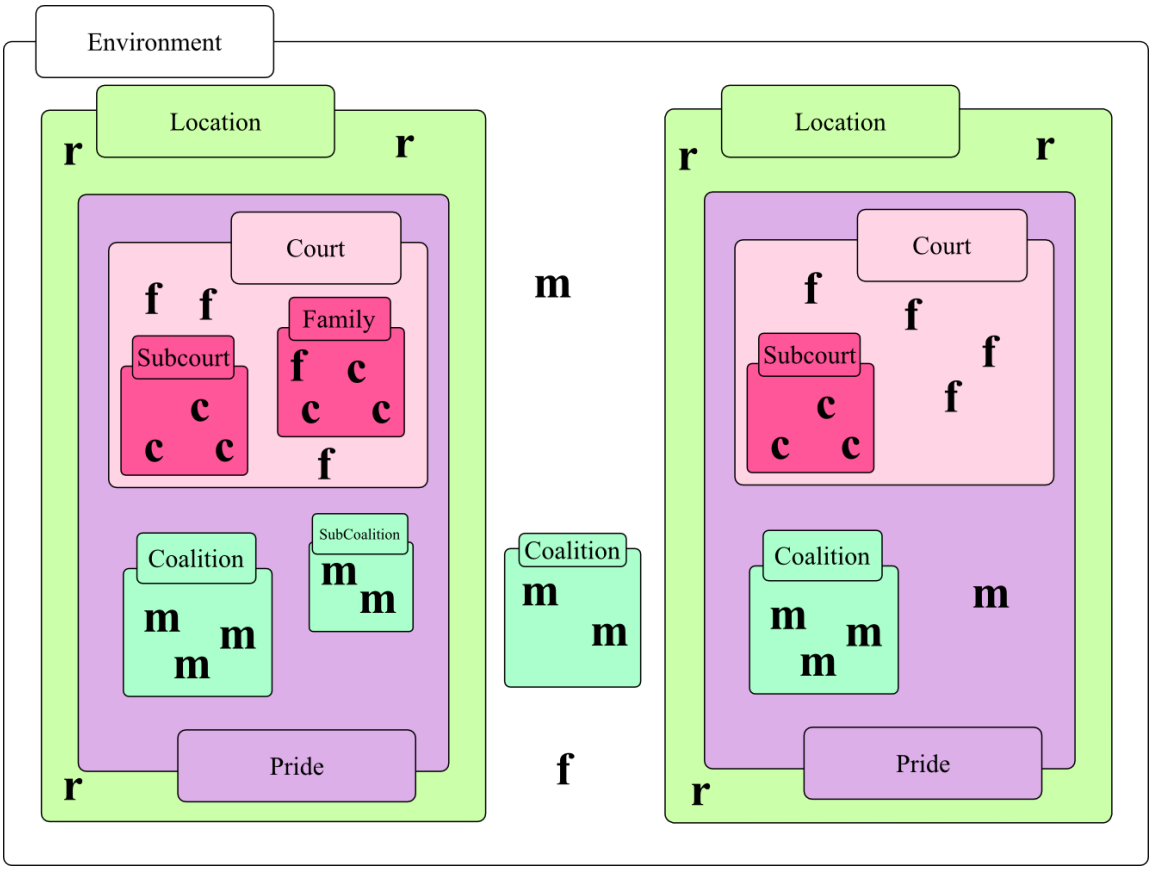
P is a tuple so composed: $\{\Theta, \Gamma, \Sigma, \omega_0, R\}$

1. Σ is a finite alphabet of symbols { m,f,c,r,s,winter,spring,autumn,summer } representing elements present into membranes, they respectively stand for male, female, cub, resources, sources and seasons, season are used as control elements
2. Γ is a finite alphabet of symbols { E,L,P,Cr,Cl,Scr,Scl,F } representing possible type of membranes:

(a) E : Environment
(b) L : Location
(c) P : Pride
(d) Cr: Court
(e) Cl: Coalition
(f) Scr: SubCourt
(g) Scl: SubCoalition
(h) F : Family

3. Θ represents membranes' order:

(a) $L \subset E$
(b) $P \subset L$
(c) $Cr \subset P$
(d) $Cl \subset Cr$
(e) $Scr \subset Cl$
(f) $Scl \subset Scr$
(g) $F \subset Scl$



4. ω_0 is a tuple of values in $\Sigma, \Gamma = \{\dots\}$ who describes initial state of system, where Σ is the set of elements that we can find in a membrane of P and Γ is the set of membrane of P. The elements belonging both sets associated to some array of attributes.

5. $D_\Gamma = \{D_{\Gamma_E}, D_{\Gamma_L}, D_{\Gamma_P}, D_{\Gamma_{Cr}}, D_{\Gamma_{Cl}}, D_{\Gamma_{Scr}}, D_{\Gamma_{Scl}}, D_{\Gamma_F}\}$ Ordered set of domains, in a one-to-one correspondence with elements of set Γ.

(a) $D_{\Gamma_E} = \{geneticcode, privation\}$
Genetic code indicates the most suitable lion who can survive as a peripatetic, Privation indicates a percentage of weakness that can affect lion's health.

(b) $D_{\Gamma_L} = \{geneticcode, \}$
Genetic code indicates the most suitable lion who can survive in a specific location.

(c) $D_{\Gamma_P} = \{Health, Strength, Requirements, Coalitions\}$
$Health$ indicates the total amount of *health* property taken from every members of a Pride, $Strength$ indicates the total amount of $Strength$ property taken from every members of a Pride, $Requirements$ indicates total food requirements for a pride, $Coalitions$ indicates the number of coalitions present in a Pride.

(d) $D_{\Gamma_{Cr}} = \{Health, Strength, Requirements\}$
Health indicates the total amount of Health property taken from every members of a Court, Strength indicates the total amount of Strength property taken from every members of a Court, Requirements indicates total food requirements for a pride.

(e) $D_{\Gamma_{Cl}} = \{Health, Strength, Requirements, Resident\}$
Health indicates the total amount of Health property taken from every members of a Coalition, Strength indicates the total amount of Strength property taken from every members of a Coalition, Requirements indicates total food requirements for a pride. Resident indicates if a Coalition is resident or not.

(f) $D_{\Gamma_{Scr}} = \{Health, Strength, Requirements\}$
Health indicates the total amount of Health property taken from every members of a SubCourt, Strength indicates the total amount of Strength property taken from every members of a SubCourt, Requirements indicates total food requirements for a pride.

(g) $D_{\Gamma_{Scl}} = \{Health, Strength, Requirements\}$
Health indicates the total amount of Health property taken from every members of a SubCoalition, Strength indicates the total amount of Strength property taken from every members of a SubCoalition, Requirements indicates total food requirements for a pride.

(h) $D_{\Gamma_F} = \{Health, Cubs, Age\}$
Health indicates the total amount of Health property taken from every members of a Family, Cubs indicates the total number of Cubs within the Family, Age indicates the age of the cubs, all cubs have the same age.

6. $D_\Sigma = \{D_{\Sigma_m}, D_{\Sigma_f}, D_{\Sigma_c}, D_{\Sigma_r}\}$ Ordered set of domains, in a one-to-one correspondence with elements of set Σ.

(a) $D_{\Sigma_m} = \{Age, Strength, Geneticcode, resident\}$
Age indicates how old a lion is, Strength indicates the percentage of success for a lion to defeat other lions, according to strength of opponents, Genetic code indicates if a lion is suitable for the location in which he lives, resident is a flag, representing if the male is resident into a pride from more than one year or if it is just arrived after a take over. Instead, if he have just take over the pride, he will kill the cubs of females.

(b) $D_{\Sigma_f} = \{Age, Strength, Geneticcode, Secondarygeneticcode, \}$
Age indicates how old a lioness is, Strength indicates the percentage of success for a lioness to defeat other lions, according to the strength of opponents, Genetic code indicates if a lioness is suitable for the location in where she lives

while secondary genetic code indicates the genetic code of the male, if she is in the family way, it's empty otherwise.

(c) $D_{\Sigma_c} = \{Geneticcode, Sex\}$
Cubs don't need an age, it is recorded in the Family attributes, Genetic Code will be useful once they will become adults, sex is necessary to produce a male or a female from a cub.

(d) $D_{\Sigma_r} = \{Rating, Amount\}$
Rating indicates how fast a resource grow up, Amount indicates how much food requirements can satisfy.

(e) $D_{\Sigma_s} = \{\epsilon\}$
this item has no attributes it is necessary only to produce new resources.

(f) $D_{\Sigma_{spring}} = \{duration\}$
duration indicates how many iterations this control item waits before it expires.

(g) $D_{\Sigma_{summer}} = \{duration\}$
duration indicates how many iterations this control item waits before it expires.

(h) $D_{\Sigma_{autumn}} = \{duration\}$
duration indicates how many iterations this control item waits before it expires.

(i) $D_{\Sigma_{winter}} = \{duration\}$
duration indicates how many iterations this control item waits before it expires.

7. The system evolves within a cycle of four seasons:

	F	Scl	Scr	Cl	Cr	P	L	E
Season 1					Birth		Coalition enter in pride Food collecting	
Season 2					Family became Subcourt	Internal fight Expulsion	Expulsion	Coalition enter in location
Season 3				Expulsion	mating Expulsion	Male enter in Court		
Season 4	Selection Aging	Selection Aging	Selection Aging	Family become Court Selection Aging	Selection Aging	Subcoalition become Coalition Selection Aging	Selection Aging	Selection Aging

In each season we can see a different set of rules for each membrane:

(a) Season 1:
During season 1, new lions are born; Families are created; a coalition, stationing in the same location of a Pride where there is only a coalition, can enter into it; in the same season lions collect food. During the hunt some lion could die.

(b) Season 2
During season 2, there could be a take over; a Subcoalition too old to be considered a cub could be expelled from pride, Coalition or Subcoalition present in a location could be expelled.

(c) Season 3
This is the mating season, males leave their membranes and, from Pride membrane enter in court membrane, where they mate with females; this season takes eight iterations, to allow this shifting of lions and their return to their pride

(d) Season 4
Winter has come, this is the harder season for lions, in this season, those who survives get older otherwise they dies. We use this season to simulate ageing, families old enough become subcourts, other families become subcoalitions, subcoalitions get stronger and become coalition, ready to try an internal take over in the next season.

8. The set of rules R is divided in subsets by membranes: (E,L,P,Cr,Cl,Scr,Scl,F) to make reading easier we decided to divide them even further according to the seasons.

   (a) Season 1

      i. $R_E = \{$
      $spring \rightarrow summer$
      $Cl, L_{coalitions<2} \rightarrow Cl_{in(L)}, L$
      $\}$
      During Season1, coalitions move from environment to locations where the number of coalitions is less than 2. In each membrane we have an element usually used as promoter, that changes from spring to summer.

      ii. $R_L = \{$
      $spring \rightarrow summer$
      $P, r \rightarrow P, r$
      $P_{health}, r_{amount} \rightarrow P_{health+=amount}$
      $P_{health}, r_{amount} \rightarrow P_{health+=amount}, w_{in(P)}$
      $P, r \rightarrow P, r, w_{in(P)}$
      $Cl, P_{coalitions<2} \rightarrow Cl_{in(P)}$
      $\}$
      During Season1, a Pride hunts a resource, there are four rules that managed this event and which represent the possibility of capturing the food, or not, and the possibility that during the hunt there would be casualties, or injuries. In the same season a Coalitions moves into a pride where the number of coalitions is less than 2.

      iii. $R_P = \{$
      $spring \rightarrow summer$
      $w, Cl \rightarrow w_{in(Cl)}, cl$
      $w, Cr \rightarrow w_{in(Cr)}, cl$
      $w, Scl \rightarrow w_{in(Scl)}, cl$
      $w, Scr \rightarrow w_{in(Scr)}, cl$
      $\}$
      During Season1, a wound (represented by one elements generated from fight rules) received by Pride goes into a coalition, a subcoalition, a subcourt or into a court.

      iv. $R_{Cr} = \{$
      $spring \rightarrow summer$
      $f_{code,code^2,strength,age} \rightarrow F_{health,age=0,litter=2}(\{f, c_{code=mix(code_f, code_f^2)}, c_{code=mix(code_f, code_f^2)}\}$
      $f_{code,code^2,strength,age} \rightarrow F_{health,age=0,litter=3}(\{f, c_{code=mix(code_f, code_f^2)}, c_{code=mix(code_f, code_f^2)},$
      $c_{code=mix(code_f, code_f^2)}\})$
      $f_{code,code^2,strength,age} \rightarrow F_{health,age=0,litter=4}(\{f, c_{code=mix(code_f, code_f^2)}, c_{code=mix(code_f, code_f^2)},$

$c_{code=mix(code_f, code_f^2)}$, $c_{code=mix(code_f, code_f^2)}\})$
$f_{code,code^2,strength,age} \rightarrow F_{health,age=0,litter=6}(\{f, c_{code=mix(code_f, code_f^2)}, c_{code=mix(code_f, code_f^2)},$
$c_{code=mix(code_f, code_f^2)}, c_{code=mix(code_f, code_f^2)}, c_{code=mix(code_f, code_f^2)} c_{code=mix(code_f, code_f^2)}\})$
$Scl \rightarrow Scl_{out}$;
}
During Season1, a pregnant female, who copied in $code^2$ the genetic code of a male, generates a Family, four different rules concur to decide whether the number of cubs in the family will be 2, 3, 4 or 6, in this season a sub-coalition, generated from a Family, goes out from the court to the Pride.

v. $R_{Cl} = \{$
$spring \rightarrow summer$
$w, m \rightarrow m$
$w, m \rightarrow \_$
}
During Season1, in each membrane we have an element usually used as promoter, that changes from spring to summer.

vi. $R_{Scr} = \{$
$spring \rightarrow summer$
$w, f \rightarrow f$
$w, f \rightarrow \_$
$c \rightarrow f$
}
During Season1, in each membrane we have an element usually used as promoter, that changes from spring to summer.

vii. $R_{Scl} = \{$
$spring \rightarrow summer$
$w, m \rightarrow m$
$w, m \rightarrow \_$
$c \rightarrow m$
}
During Season1, in each membrane we have an element usually used as promoter, that changes from spring to summer.

viii. $R_F = \{$
$spring \rightarrow summer$
During Season1, in each membrane we have an element usually used as promoter, that changes from spring to summer.

(b) Season 2

i. $R_E = \{$
$summer \rightarrow autumn$
$m \rightarrow m_{in(Cl)}|Cl$
}
During Season 2, in each membrane we have an element usually used as promoter, that changes from summer to autumn.

ii. $R_L = \{$
$summer \rightarrow autumn$
$Cl \rightarrow Cl_{out}$
}
During Season 2, a Coalition exits from location to environment.

iii. $R_P = \{$
$summer \rightarrow autumn$

$Cl_{strenght}, Cl_{strenght'} \overset{strenght/(strenght+strenght')}{\rightarrow} Cl_{resident=true}, Cl_{out}$
$Cl_{strenght}, Cl_{strenght'} \overset{strenght'/(strenght+strenght')}{\rightarrow} Cl_{resident=true}, Cl_{out}$
}
During Season 2, two coalitions fight and one is expelled from pride, the coalition who won becomes the resident one.

iv. $R_{Cr} = \{$
$summer \rightarrow autumn$
$w, f \rightarrow f$
$w, f \rightarrow \_$
}
During Season 2, a wound kills a female or is consumed.

v. $R_{Cl} = \{$
$w, m \rightarrow m$
$w, m \rightarrow \_$
$summer \rightarrow autumn$
}
During Season 2, a wound kills a male or is consumed.

vi. $R_{Scr} = \{$
$summer \rightarrow autumn$
$w, f \rightarrow \_$
}
During Season 2, a wound kills a female

vii. $R_{Scl} = \{$
$summer \rightarrow autumn$
$w, c \rightarrow \_$
}
During Season 2, a wound kills a male

viii. $R_F = \{$
$summer \rightarrow autumn$

(c) Season 3

i. $R_E = \{$
$autumn_{duration>8} \rightarrow winter$
$autumn_{duration\leq 8} \rightarrow autumn_{duration+1}$
$L, Cl \rightarrow Cl_L, L$
}
During Season 3, in each membrane we have an element usually used as promoter, that changes from autumn to winter, if duration > 8 else duration is increased by one.

ii. $R_L = \{$
$autumn_{duration>8} \rightarrow winter$
$autumn_{duration\leq 8} \rightarrow autumn_{duration+1}$
}
During Season 3, in each membrane we have an element usually used as promoter, that changes from autumn to winter, if duration > 8 else duration is increased by one.

iii. $R_P = \{$
$autumn_{duration>8} \rightarrow winter$
$autumn_{duration\leq 8} \rightarrow autumn_{duration+1}$

$m, Cr \overset{duration \pmod 2=0}{\rightarrow} m_{in(Cr)}, Cr$
$m, Cl \overset{duration \pmod 2=1}{\rightarrow} m_{in(Cl)}, Cl$
}
During Season 3, males exit from coalition and enter into court to mate with females.

iv. $R_{Cr} = \{$
$autumn_{duration>8} \rightarrow winter$
$autumn_{duration\leq 8} \rightarrow autumn_{duration+1}$
$m \overset{duration \pmod 2=1}{\rightarrow} m_{out}$
$f \overset{strength_m}{\rightarrow} f_{code^2=code_m}|m$
$f \rightarrow f|m$
$F \rightarrow f|m_{resident=false}$
}
During Season 3, males exit from court to pride, after that they mated with females, if males are not resident (The coalition was set as resident but not males inside) a Family produces a female.

v. $R_{Cl} = \{$
$autumn_{duration>8} \rightarrow winter$
$autumn_{duration\leq 8} \rightarrow autumn_{duration+1}$
$m \overset{duration \pmod 2=0}{\rightarrow} m_{out}$
}

vi. $R_{Scr} = \{$
$autumn_{duration>8} \rightarrow winter$
$autumn_{duration\leq 8} \rightarrow autumn_{duration+1}$
}
During Season 3, in each membrane we have an element usually used as promoter, that changes from autumn to winter, if duration $>$ 8 else duration is increased by one.

vii. $R_{Scl} = \{$
$autumn_{duration>8} \rightarrow winter$
$autumn_{duration\leq 8} \rightarrow autumn_{duration+1}$
$c \rightarrow m;$
}
During Season 3, a cub become a male.

viii. $R_F = \{$
$autumn_{duration>8} \rightarrow winter$
$autumn_{duration\leq 8} \rightarrow autumn_{duration+1}$

(d) Season 4

i. $R_E = \{$
$winter \rightarrow spring$
}
During Season 4, in each membrane we have an element usually used as promoter, that changes from winter to spring.

ii. $R_L = \{$
$winter \rightarrow spring$
}

During Season 4, in each membrane we have an element usually used as promoter, that changes from winter to spring.

iii. $R_P = \{$
$winter \rightarrow spring$
$Scl \rightarrow Scl_{age+1}$
$m_{resident} \overset{\phi(age)}{\rightarrow} m_{resident=resident',age+1} | Cl_{resident'}$
$m \overset{\phi(age)}{\rightarrow} \_$
$\}$
During Season 4, a male ages or dies according to the rules and a subcoalition ages. Value of resident attribute of males is set with the value of resident of Coalition, so males that were not resident in Season 3 become resident.

iv. $R_{Cr} = \{$
$winter \rightarrow spring$
$Scr \rightarrow Scr_{age+1}$
$F \rightarrow F_{age+1}$
$F_{age>3} \rightarrow Cl$
$F_{age>3} \rightarrow Cr$
$f \overset{\phi(age)}{\rightarrow} f_{age+1}$
$f \overset{\phi(age)}{\rightarrow} \_$
$\}$
During Season 4, a female ages or dies according to the rules and a subcourt ages.

v. $R_{Cl} = \{$
$winter \rightarrow spring$
$m \overset{\phi(age)}{\rightarrow} m_{age+1}$
$m \overset{\phi(age)}{\rightarrow} \_$
$\}$
During Season 4, a male ages or dies according to the rules.

vi. $R_{Scr} = \{$
$winter \rightarrow spring$
$f \overset{\phi(age}{\rightarrow} f_{age+1}$
$f \overset{\phi(age}{\rightarrow} \_$
$\}$
During Season 4, a female ages or dies according to the rules.

vii. $R_{Scl} = \{$
$winter \rightarrow spring$
$m_{age} \overset{\phi(age)}{\rightarrow} m_{age+1}$
$m_{age} \overset{\phi(age)}{\rightarrow} \_$
$\}$
During Season 4, a male ages or dies according to the rules.

viii. $R_F = \{$
$winter \rightarrow spring$
$f_{age} \overset{\phi(age}{\rightarrow} f_{age+1}$
$f_{age} \overset{\phi(age}{\rightarrow} \_$
$c \overset{\phi(age}{\rightarrow} \_ | F_{age}$

$c \overset{\phi(age}{\rightarrow} c|F_{age}$
}
During Season 4, a female ages or dies according to the rules, a cubs dies or survives according to the rules.

where $\phi(age)$: is a function that return 1 if $age < 10$, 0 if $age = 16$, 0,5 if $age \geq 10$, 0,5 if $age \leq 3$

9. The set of update functions U is divided in subsets by membranes:
(E,L,P,Cr,Cl,Scr,Scl,F)

   (a) $U_E = \{$
   Empty
   }

   (b) $U_L = \{$
   $\phi_{Coalitions} : arity(Cl) \in L$
   }
   this function calculates the number of Coalitions into a location.

   (c) $U_P = \{$
   $\phi_{Coalitions} : arity(Cl) \in P$
   $\phi_{Strength} : Strength_{Cr} + Strength_{Cl} + \Sigma_{\forall Scl \in w_P} Strength_{scl}$
   }
   this functions calculate the number of Coalitions into a Pride and the strength of a Pride.

   (d) $U_{Cr} = \{$
   $\phi_{Strength} : \Sigma_{\forall f \in w_{Crl}} Strength_f + \Sigma_{\forall Scr \in w_{Cr}} Strength_{Scr}$
   }
   this function calculates the strength of a Court.

   (e) $U_{Cl} = \{$
   $\phi_{Strength} : \Sigma_{\forall m \in w_{Cl}} Strength_m$
   }
   this function calculates the strength of a Coalition.

   (f) $U_{Scr} = \{$
   $\phi_{Strength} : \Sigma_{\forall f \in w_{Scr}} Strength_f$
   }
   this function calculates the strength of a subCourt.

   (g) $U_{Scl} = \{$
   $\phi_{Strength} : \Sigma_{\forall m \in w_{Scl}} Strength_m$
   }
   this function calculates the strength of a subCoalition.

   (h) $U_F = \{$
   $\phi_{Cubs} : arity(Cubs) \in w_F$
   } this function calculates the number of cubs into a Family.

6.7 Experimental results

The simulations we run, with different sets of parameters and starting conditions, lead to realistic outcomes in accord with the observations of the last 20 years in Serengeti Park. Obviously to have a perfect model we have to add and model a lot of other facts about the life and death of Lions in the park. Anyway to show the features of MAPP systems this level of detail is more than enough.

6.8 Data and results

From our model we can observe that, once we reach a certain complexity level, which includes a great number of elements and iterations, the system reaches a certain stability. Where the number of elements of a pride in one location does not grow and the coalition within it is strong enough to not suffer a take over from foreign coalitions.
In figure 6.7 we can see a Pride that starts with only one male and one female and a subcourt of three young females. At time t_{12} young females grow and become adult, so the number of young females decreases to 0 and the number of adult females increases to 3, while an adult female dies. The pride grows as females reproduce and, between time t_{25} and time t_{58}, the number of cubs increases. At time t_{83} we can see 5 females, 2 young females 1 male and 4 cubs. At time t_{84} three old females die because of their old age, three young females become adult, and a family changes in one subCoalition. At time t_{85} 10 new cubs are born and at time t_{96} a subCoalition evolves in coalition, ready for an inner take over. Graph shows the evolution of pride within a number of 120 iterations, that means 10 years, because every 12 iterations our system passes through all seasons. It is important to note that most of the significant changes take place along seven years. During this time, females become pregnant, families grow and take their places into cohorts or coalitions, lions begin to die. This example shows a period of only ten years, to describe what happens from the birth of a pride, and it gives a comprehensive idea of pride's dynamics. Longer simulations show that a situation where deaths reduce the population growth and lead to a substantial stability of the system is finally reached.

In figure 6.8 we can see a Pride starting with only one family composed by an adult

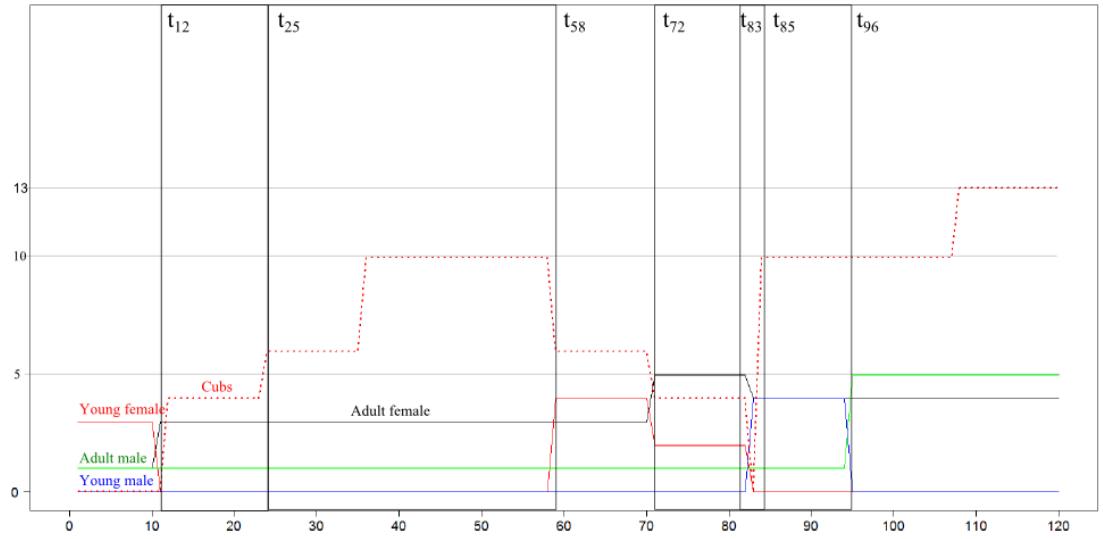


Figure 6.7: Pride starts with only one male and one female and a subcourt of three young females

female and three cubs, six young females and two coalitions each one consisting of one male. At time t_{12} we see the effect of a take over, a family dies with its cubs. The young females become adult and one new family with four cubs is born. At time t_{25} we see that other two families are born and we have twelve cubs. The oldest four grow up and become young females, forming a subcourt at time t_{59} while at time t_{72} four young females become adults, four cubs become young males, forming a subpride and four cubs become young females. At t_{85} the lone adult male dies, young females become adult, and young males become adult. At time t_{88} four females die. At time t_{95} six new cubs are born.

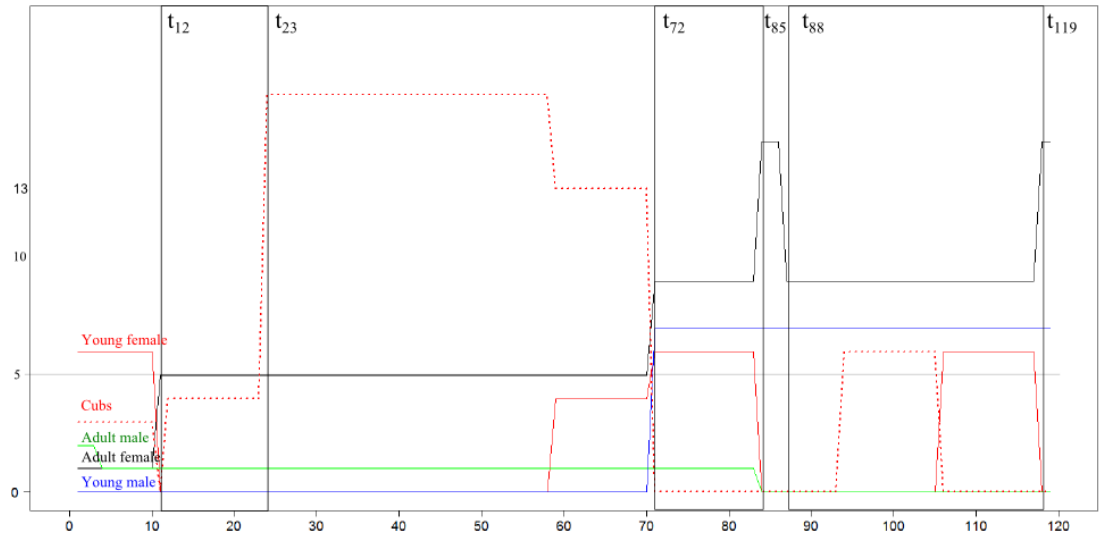


Figure 6.8: Pride starting with only one family composed by a single adult female and three cubs, six young females and two adult males

These are just two examples of what we can produce as output, using our model and our software. We wanted to propose a comprehensive and at the same time easy to read overview of the social dynamics of Serengeti lions. Changing certain settings may offer different solutions, such as the difficulty in finding food, or the rate of death or birth of lions, the different life expectancy between males and females, all of these are parameters that can affect results.

This model can be used to predict what happens in a pride when an outside coalition is introduced in the same environment. Sufficiently strong residents adult Lions can stand up to any foreign coalitions and bring their own cubs to reach the age of maturity, while a weak coalition is unable to maintain control of the pride and makes it liable to a take over.

6.9 Conclusion

In conclusion we say MAPP systems lose some features about maximum parallelism but gain the possibility to have membranes and aggregations of elements/animals interact directly under special rules who can take them as reagents. Our results on the case study are a good match with those of the chosen reference texts [66, 65, 72]

We want to highlight again how case studies choosen for those chapters are instrumental in showing the instruments made available by formalisms and how their evolution brings us to MAPP system.

Our focus, more than utilizing correct parameters and accurately predicting outcomes, is on showing how from credible parameters estimations we can obtain scenarios useful to extract clues about the examined model.

Those cluese can be used to suggest new experiments suitable to better evaluate paraments, describe an iterative modeling process which can lead us to reconsider the model including new facts and attributes, validate theories or propose new ones.

Of course a purely predictive model is possible granted you have the right amount of data gathered on the field, in this case it's possible to obrain a model useful for the decision making or the biologist community.

Speaking of which one of the next developments we are moving towards is the creation of an interface which could make this modeling style more appealing outside the computer scientists community.

What we would like to propose is mainly a more standardized formalization to define the set of terminal elements, the set of membranes and rules. Through this we should be able to obtain simpler models, as useful as more widespread agent based models.

Chapter 7

Conclusions and future developments

What drives us to define MPP systems starting from P systems, APP systems starting from MPP systems and MAPPS starting from APP systems are new families/groupings/domains of problems which require new tools to be described and simulated successfully.

It is possible the next challenge is about problems with an entire ecosystem with different species living togheter. One complex grid describes relations between different species, some species living in sympatry, others in prey-predator relationship, others in symbiosis.

It is possible, to add complexity to the system, for one of the involved species to be humans. What kind of real problems can be modelled with MAPP systems? Is this tool for theoretical models only or can it be used by biologist too?
We already produced some general purpose engines to run simulations expressed in MAPP formalism. This code take as input one XML-like description of the model and some initial state (list of elements presents into the starting internal state of any membrane of the system). After generating the model this engine can run simulations with a fixed number of computational steps to give us as output some final internal state of the system to study.
Anyway, since there is still a lot to work to do with this kind of instruments, here we propose two of many possible research avenues: in the first one we show how is possible to emulate agent-based systems with MAPP with a minor change in the way we model the system entities. In the second one we shortly describe the advantage we can obtain using again PRISM to translate APP and MAPP systems, like we do for the MPP.

7.1 The use of MPP, APP and MPP systems and other considerations

In Fig. 7.1 we can see the main features of the systems presented in this thesis, which were introduced, from system to sytem, in order to offer modelling tools for the various phenomena encountered. Surely agent based systems have a big appeal for the ecologist community and the rule based approach often results too strictly formal to be really used outside the Computer Science academia.
Moreover, focus on rules rather than on agents gives us another perspective to the study of phenomena. We think this change of perspective can be very useful in some specific cases, for example when, in some situation or within some specific experiments, is more easy observe interactions whenever they happen rather than the full behaviour of individuals.

P systems	MPP systems	APP systems	MAPP systems
parallel computation non deterministic choice of rules rewriting rules over objects catalyst multilevel membrane structure	parallel computation probabilistic choice of rules rewriting rules over objects rating function over internal state flat membrane structure	parallel computation probabilistic choice of rules rewriting rules over objects control objects attributes over objects rating function over internal state and attributes flat membrane structure	cascade computation probabilistic choice of rules rewriting rules over objects and membrane control objects attributes over objects and membranes rating function over internal state and attributes multilevel membrane structure

Figure 7.1: here we summarize the main features of the formalism proposed in this thesis

Is hard to give an accurate comparison between MAPP systems and agent-based systems, bu I will do some very simple considerations:
- is possible to emulate simple agent-based system with an APP system, for any "action" one agent in the agent-based system can do, you can shape a computation stage using control objects in APP system.
In that stage you put in place two rules (or more, if the action can have more than one outcome), one rule doesn't change the element which acts as reagent. This rule models the fact that the agent doesn't take that action, while the other rule changes the element which acts as reagent, to model the fact that the agent takes that action. Stage by stage you describe the behaviour of the agent-based system, defining a succession of actions any element in the system takes or not in his life.
Probabilistic function associated to rules will take as input only attributes of the object used as reagent, to rate the possibility of the action happening (or not).

- is possible to apply the ODD protocol as a reference to analyze the APP (and MAPP) system too. If we consider the main points only they can be applied to our system:
State variables and scales - in APP systems is possible to find variables and parameters in two main features. Variables are associated to any element as attributes, any attributes are well defined and their use is defined by the system rules. Variables are also present in the rate functions formalization associated to the system rules. Like the case study we present in this thesis, the documentation of the model will always clearly state any parameter used in it.
Process overview and scheduling - often it is not specified in what sequence model entities are processed and when state variables are updated, but in this kind of system the sequence is defined by control objects and by the rules which compute when one control object (and computational stage) shifts into another. In any different stage all the elements who can be a reagent of one of the rules enabled in that stage by control objects are processed. The updates of the variables are all parallel, after the maximal set of rules to be applied has been computed.
emphInizialization and Input Data - the initial state of the system, as any other state of the system is always clearly defined by the multiset of elements in the Env membrane with the value of attributes associated to each element. Input data of the system are completed by parameters of the rules and are also clearly stated in the formulation of the model.
Submodels - the idea of submodel are always present in ours systems, in MAPP system any membrane define a submodel and the cascade computation of the whole system put in evidence the skeleton of the model. Is easy divide the model in membranes or stages submodels and add to a model new modules adding elements to the definition.
In conclusion some of the point od ODD protocol can be applyed to check some strong point of the three systems presented in the thesis as well, some features anyway can be better described by formal definitions and inference rules.

- Global sensitivity analysis (SA) aims to quantify the relative importance of input variables or factors in determining the output value of the whole system. We can identify two different schools of thought: the local SA school, and the global one. In the first one, the local response of the system output, obtained by varyng input factors one at a time, is investigated while holding the others fixed to a central (nominal) value. The analysis is a run at a given central point in the input factors space, and the volume of the region explored is nil. The second SA school is more ambitious in two respects: first, the space of the input factors is explored within a finite (or even infinite) region and, second, the output variation induced by a factor is taken globally - that is, averaged over the variation of all factors.
This kind of investigations on MPP, APP and MAPP system is hard because the system's output depends on both the initial multiset of elements (and the associated attributes values) and on the parameters associated to rules and rate functions. Translation in PRISM language can help in this kind of analysis of model behaviour, but any single model needs a specific dissertation and study.

7.2 Use of MAPP systems to emulate agent based models

This section will not develop further MAPP systems but will try to change perspective by using them with another goal. We plan to extend the paradigms the formalism can deal with. We consider a MAPPS with this definition:
- Γ is the set of terminal elements. The set Γ is composed by the union of two sets. The first one is composed by terminal elements called "motivation", for example $\{hunger, fear, procreation, aggression\}$. The second set contains common elements which are important for the model but don't need a fine modellation like $\{prey, predator\}$
- Σ is composed by the Enviroment membrane and only another level of membranes called "Individuals". $\Sigma = \{Enviroment, Individual\}$
- D_Γ is empty, for now we consider all the "motivation" elements without attributes.
- D_Σ is composed only by the set of attributes of Individuals $D_{Individual}$, the set $D_{Individual}$ has only the attribute "behaviour". The domain of the attribute "behaviour" is a special set of possible actions the "Individual" is willing to do, for example $behaviour \in \{mate, eat, hide, attack\}$.
- R is the set of sets of rules associated to membranes $\in \Sigma$, so $R = \{R_{Enviroment}, R_{Individual}\}$. The set $R_{Enviroment}$ is composed by rules that manage the interaction between membranes modelingl individuals. All the probabilistic functions associated to the rules in $R_{Enviroment}$ take into account the attribute "behaviour" of any "Individual".
For example, if we suppose our Individuals are Lions, we can have a rule in the Enviroment to model one hungry Lion who eats some prey:

$$Lion_{(eat,S)}, prey \rightarrow Individual_{(eat,S\cup\{prey)\}}$$

The meaning of this rule is "a lion which wants to eat something meets a prey, with some probability p the outcome of the encounter will be the lion with the prey added inside his internal state S to model the lion eating the prey".
In the next computational step of that lion's internal state, the prey will satisfy the hunger of the lion, andit will be more likely he wants to do something else than hunting.
The set $R_{Individual}$ is composed by rules that manage the internal decision of the individuals, the "motivation" elements inside the individual interact with each other to evolve the internal state of individuals.
For example:

$hunger, prey \rightarrow \_$,
$prey, prey \rightarrow fat$,

The main feature of this paradigm is the work of the single function inside the set $U_{Individual}$. This update function takes into account the internal state of each Individual membrane and gives the new value of the attribute "behaviour" to that Individual.
In a very simple example of $U_{(Individual,behaviour)}$ the function will set the "behaviour" value according with the prevalent "motivation" element present in its internal state. If the number of "hunger" elements inside Individual internal state surpasses all the other elements, then the "behaviour" will be "eat", if "fear" is the most present "motivation" element then the "behaviour" will be "hide" and so on.

Without others details the idea behind this particular application of the MAPP systems is to use the last membrane of Σ set to model individuals of the model. If in the MAPP model we have terminal elements "MaleLion" and "FemaleLion", in this application we will have two membrane called "MaleLion" and "FemaleLion". Any membrane which models one individual has one attribute "behaviour". The value assumed by the attribute "behaviour" for any elements selects the subset of rules of the Enviroment which can take the Individual as reagent.
The attribute "behaviour" works in a very similar way to control objects. The difference is that control objects are used to divide rules in subsets associated to different computational stage, while the attribute "behaviour" is used to divide rules applicable to the individual into subsets associated with what the Individual is willing to do.
Summing up:

- Any membrane models one Individual. The internal state of any membrane which models an individual can be composed only by elements in Γ set, both "motivation" elements and "common" elements.
- The internal computation step takes the internal state as input and has the new internal state as output.
- After any "individual" membrane calculates its internal state, the update function of each membrane calculates the new value of the "behaviour" attribute.

- When all membranes which model one Individual have a new "behaviour" attribute, the computation step of upper membranes takes place. The rule sets of the membrane takes the Individual's membranes as reagents taking into account the "behaviour" attribute.

- All the internal states of all membranes are calculated like MAPP systems till the computational full step is completed, and so on.

We still use a formalism derived from P systems, but the individual membranes work in a similar way to agents in a classic agent-based model. The membranes/Individuals independently decide on their internal state what actions are possible, and run in parallel, interacting with each other to make the system evolve.
In [134] something very similar is proposed, but at cellular and tissue level. Cells are described as membranes with their internal composition (and computations). They are inside an enviroment membrane which contains all the system. Cells can be of different kinds (like we model different species), also cells can be born or die and interact with each others with some communication rules. "Population P systems" will provide a good guide to what kind of property is necessary to enforce when we will be working on this kind of

formalisms. Anyway this use of MAPP systems still needs lot of work but we think this can be a good direction to improve the use of the formalism and his way to model agents-like in general.

7.3 PRISM translations

MPP system models can be easily simulated and translated into the PRISM input language to enable statistical model checking of properties. In chapter four case study we have demonstrated that the use of such two analysis techniques allows complementary aspects of the systems to be studied, and hypotheses to be verified. As a future research direction we hope to provide translations in PRISM for the others two formalisms (APP and MAPP systems). Attributes and membranes raise some problems to overcome for a correct translation. For sureit will be useful to have this kind of analysis tool for our models, in order to give better previsions and to define more rigorously the models' probabilistic properties.

Bibliography

[1] Cardelli Luca "Brane Calculi, Interactions of biological membranes." Proceedings of Computational Methods in systems Biology. 2004.

[2] Simao, E., et al. "Qualitative modelling of regulated metabolic pathways: application to the tryptophan biosynthesis in E. coli." Bioinformatics 21.suppl 2 (2005): ii190-ii196.

[3] Kuffner, Robert, Ralf Zimmer, and Thomas Lengauer. "Pathway analysis in metabolic databases via differential metabolic display (DMD)." Bioinformatics 16.9 (2000): 825-836.

[4] Nagasaki, Masao, et al. "Bio-calculus: its concept and molecular interaction." Genome Informatics 10 (1999): 133-143.

[5] John, Mathias, Roland Ewald, and Adelinde M. Uhrmacher. "A Spatial Extension to the p Calculus." Electronic notes in theoretical computer science 194.3 (2008): 133-148.

[6] Breitling, Rainer, et al. "A structured approach for the engineering of biochemical network models, illustrated for signalling pathways." Briefings in bioinformatics 9.5 (2008): 404-421.

[7] Gupta, Simone, et al. "Boolean network analysis of a neurotransmitter signaling pathway." Journal of theoretical biology 244.3 (2007): 463-469.

[8] Li, Fangting, et al. "The yeast cell-cycle network is robustly designed." Proceedings of the National Academy of Sciences of the United States of America 101.14 (2004): 4781-4786.

[9] Shmulevich, Ilya, et al. "Probabilistic Boolean networks: a rule-based uncertainty model for gene regulatory networks." Bioinformatics 18.2 (2002): 261-274.

[10] Akutsu, Tatsuya, Satoru Miyano, and Satoru Kuhara. "Inferring qualitative relations in genetic networks and metabolic pathways." Bioinformatics 16.8 (2000): 727-734.

[11] D'haeseleer, Patrik, Shoudan Liang, and Roland Somogyi. "Genetic network inference: from co-expression clustering to reverse engineering." Bioinformatics 16.8 (2000): 707-726.

[12] Akutsu, Tatsuya, Satoru Miyano, and Satoru Kuhara. "Identification of genetic networks from a small number of gene expression patterns under the Boolean network model." Pacific symposium on biocomputing. Vol. 4. 1999.

[13] Ermentrout, G. Bard, and Leah Edelstein-Keshet. "Cellular automata approaches to biological modeling." Journal of theoretical Biology 160.1 (1993): 97-133.

[14] Kauffman, Stuart A. "Metabolic stability and epigenesis in randomly constructed genetic nets." Journal of theoretical biology 22.3 (1969): 437-467.

[15] Prusinkiewicz, Przemyslaw, and Aristid Lindenmayer. The algorithmic beauty of plants. Springer Science & Business Media, 2012.

[16] Hardy, Simon, and Pierre N. Robillard. "Petri net-based method for the analysis of the dynamics of signal propagation in signaling pathways." Bioinformatics 24.2 (2008): 209-217.

[17] Sackmann, Andrea, Monika Heiner, and Ina Koch. "Application of Petri net based analysis techniques to signal transduction pathways." BMC bioinformatics 7.1 (2006): 1.

[18] Koch, Ina, Bjorn H. Junker, and Monika Heiner. "Application of Petri net theory for modelling and validation of the sucrose breakdown pathway in the potato tuber." Bioinformatics 21.7 (2005): 1219-1226.

[19] Zevedei-Oancea, Ionela, and Stefan Schuster. "Topological analysis of metabolic networks based on Petri net theory." In silico biology 3.3 (2003): 323-345.

[20] Reddy, Venkatramana N., Michael L. Mavrovouniotis, and Michael N. Liebman. "Petri net representations in metabolic pathways." ISMB. Vol. 93. 1993.

[21] Regev, Aviv, and Ehud Shapiro. "Cellular abstractions: Cells as computation." Nature 419.6905 (2002): 343-343.

[22] Regev, Aviv, William Silverman, and Ehud Shapiro. "Representation and simulation of biochemical processes using the-calculus process algebra." Pacific symposium on biocomputing. Vol. 6. 2001.

[23] Chen, Li, et al. "Modelling and simulation of signal transductions in an apoptosis pathway by using timed Petri nets." Journal of biosciences 32.1 (2007): 113-127.

[24] Danos, Vincent, and Sylvain Pradalier. "Projective brane calculus." Computational Methods in systems Biology. Springer Berlin Heidelberg, 2004.

[25] Cardelli L. Computational Methods in systems Biology. Springer, Berlin Heidelberg; 2005

[26] Priami, Corrado, et al. "Application of a stochastic name-passing calculus to representation and simulation of molecular processes." Information processing letters 80.1 (2001): 25-31.

[27] Priami, Corrado. "Stochastic π-calculus." The Computer Journal 38.7 (1995): 578-589.

[28] Penna, Pierluigi, et al. "DISPAS: an agent-based tool for the management of fishing effort." Software Engineering and Formal Methods. Springer International Publishing, 2013. 362-367.

[29] De Jong, Hidde, et al. "Qualitative simulation of genetic regulatory networks using piecewise-linear models." Bulletin of mathematical biology 66.2 (2004): 301-340.

[30] Turner, Thomas E., Santiago Schnell, and Kevin Burrage. "Stochastic approaches for modelling in vivo reactions." Computational biology and chemistry 28.3 (2004): 165-178.

[31] Theobald, Uwe, et al. "In vivo analysis of metabolic dynamics in Saccharomyces cerevisiae: I. Experimental observations." Biotechnology and bioengineering 55.2 (1997): 305-316.

[32] Batt, Grégory, et al. "Validation of qualitative models of genetic regulatory networks by model checking: analysis of the nutritional stress response in Escherichia coli." Bioinformatics 21.suppl 1 (2005): i19-i28.

[33] Chen, Ting, Hongyu L. He, and George M. Church. "Modeling gene expression with differential equations." Pacific symposium on biocomputing. Vol. 4. No. 29. 1999.

[34] Tyson, John J., Katherine C. Chen, and Bela Novak. "Sniffers, buzzers, toggles and blinkers: dynamics of regulatory and signaling pathways in the cell." Current opinion in cell biology 15.2 (2003): 221-231.

[35] Chassagnole, Christophe, et al. "Dynamic modeling of the central carbon metabolism of Escherichia coli." Biotechnology and bioengineering 79.1 (2002): 53-73.

[36] Gavrilets, Sergey, et al. "Case studies and mathematical models of ecological speciation. 1. Cichlids in a crater lake." Molecular Ecology 16.14 (2007): 2893-2909.

[37] Barbuti, Roberto, et al. "A methodology for the stochastic modeling and simulation of sympatric speciation by sexual selection." Journal of Biological systems 17.03 (2009): 349-376.

[38] Manca, Vincenzo, and Luca Bianco. "Biological networks in metabolic P systems." Biosystems 91.3 (2008): 489-498.

[39] Gillespie, Daniel T. "Exact stochastic simulation of coupled chemical reactions." The journal of physical chemistry 81.25 (1977): 2340-2361.

[40] Cardona, Mónica, et al. "A computational modeling for real ecosystems based on P systems." Natural Computing 10.1 (2011): 39-53.

[41] Ciocchetta, Federica, and Jane Hillston. "Bio-PEPA: A framework for the modelling and analysis of biological systems." Theoretical Computer Science 410.33 (2009): 3065-3084.

[42] Ciocchetta, Federica, and Jane Hillston. "Bio-PEPA: an extension of the process algebra PEPA for biochemical networks." Electronic Notes in Theoretical Computer Science 194.3 (2008): 103-117.

[43] Grimm, Volker, and Ilse Storch. "Minimum viable population size of capercaillie Tetrao urogallus: results from a stochastic model." Wildlife Biology 6.4 (2000): 219-225.

[44] Lindenmayer, D. B., et al. "Predictions of the impacts of changes in population size and environmental variablitity on Leadbeater's possum, Gymnobelideus leadbeateri McCoy (Marsupialia: Petauridae) using population viability analysis: an application of the computer program VORTEX." Wildlife Research 20.1 (1993): 67-85.

[45] Bustamante, Javier. "Use of simulation models to plan species reintroductions: the case of the bearded vulture in southern Spain." Animal Conservation 1.4 (1998): 229-238.

[46] Lacy, Robert C. "Structure of the VORTEX simulation model for population viability analysis." ecological Bulletins (2000): 191-203.

[47] Efthymiou, Xenia, and Anna Philippou. "A process calculus for spatially-explicit ecological models." Application of Membrane Computing, Concurrency and Agent-based Modelling in Population Biology (AMCA-POP 2010) (2010): 84-78.

[48] Philippou, Anna, Mauricio Toro, and Margarita Antonaki. "Simulation and Verification in a Process Calculus for Spatially-Explicit Ecological Models." Sci. Ann. Comp. Sci. 23.1 (2013): 119-167.

[49] Cardona, Mónica, et al. "P System based model of an ecosystem of the scavenger birds." (2009).

[50] Cardona, Mónica, et al. "Modeling ecosystems using P systems: the bearded vulture, a case study." Membrane Computing. Springer Berlin Heidelberg, 2008. 137-156.

[51] Coria, Cesar Augusto Nieto, et al. "Sea-scale agent-based simulator of solea solea in the Adriatic sea." Software Engineering and Formal Methods. Springer International Publishing, 2014. 259-275.

[52] Vorburger, Christoph, and Heinz-Ulrich Reyer. "A genetic mechanism of species replacement in European waterfrogs?." Conservation Genetics 4.2 (2003): 141-155.

[53] Giavitto, Jean-Louis, Grant Malcolm, and Olivier Michel. "Rewriting systems and the modelling of biological systems." Comparative and Functional Genomics 5.1 (2004): 95-99.

[54] Nagl, Manfred. "Graph rewriting systems and their application in biology." Lecture Notes in Biomathematics 11 (1976): 135-156.

[55] Holmes, Elizabeth E., et al. "Partial differential equations in ecology: spatial interactions and population dynamics." Ecology (1994): 17-29.

[56] Brauer, Fred, Carlos Castillo-Chavez, and Carlos Castillo-Chavez. Mathematical models in population biology and epidemiology. Vol. 40. New York: Springer, 2001.

[57] May, Robert M. "Simple mathematical models with very complicated dynamics." Nature 261.5560 (1976): 459-467.

[58] Grimm, Volker, et al. "The ODD protocol: a review and first update." Ecological modelling 221.23 (2010): 2760-2768.

[59] Grimm, Volker, et al. "A standard protocol for describing individual-based and agent-based models." Ecological modelling 198.1 (2006): 115-126.

[60] Evans, Matthew R., et al. "Do simple models lead to generality in ecology?." Trends in ecology & evolution 28.10 (2013): 578-583.

[61] Grimm, Volker, and Steven F. Railsback. "Individual-based Modeling and Ecology:(Princeton Series in Theoretical and Computational Biology)." (2005).

[62] Hamilton, S., and H. Moller. "Can PVA models using computer packages offer useful conservation advice? Sooty shearwaters Puffinus griseus in New Zealand as a case study." Biological conservation 73.2 (1995): 107-117.

[63] Brook, Barry W., et al. "Predictive accuracy of population viability analysis in conservation biology." Nature 404.6776 (2000): 385-387.

[64] Pérez-Jiménez, Mario Jesús, and Francisco José Romero-Campero. "P systems, a new computational modelling tool for systems biology." Transactions on Computational systems Biology VI. Springer Berlin Heidelberg, 2006. 176-197.

[65] Sinclair, A. R. E. "Serengeti past and present." Serengeti II: Dynamics, management, and conservation of an ecosystem 2 (1995): 3.

[66] Sinclair, Anthony Ronald Entrican, and Peter Arcese. Serengeti II: dynamics, management, and conservation of an ecosystem. Vol. 2. University of Chicago Press, 1995.

[67] Saundry, Peter. "Protected areas."

[68] Brendel, Jason. "Geography and Climate."

[69] Bygott, J. David, Brian CR Bertram, and Jeannette P. Hanby. "Male lions in large coalitions gain reproductive advantages." (1979): 839-841.

[70] Grinnell, Jon, and Karen McComb. "Roaring and social communication in African lions: the limitations imposed by listeners." Animal Behaviour 62.1 (2001): 93-98.

[71] VanderWaal, Kimberly L., Anna Mosser, and Craig Packer. "Optimal group size, dispersal decisions and postdispersal relationships in female African lions." Animal Behaviour 77.4 (2009): 949-954.

[72] Bauer, H., H. H. De Iongh, and I. Di Silvestre. "Lion (Panthera leo) social behaviour in the West and Central African savannah belt." Mammalian Biology-Zeitschrift fur Saugetierkunde 68.4 (2003): 239-243.

[73] Woodroffe, Rosie, and Laurence G. Frank. "Lethal control of African lions (Panthera leo): local and regional population impacts." Animal Conservation 8.1 (2005): 91-98.

[74] Funston, P. J., et al. "Hunting by male lions: ecological influences and socioecological implications." Animal Behaviour 56.6 (1998): 1333-1345.

[75] Heinsohn, Robert. "Group territoriality in two populations of African lions." Animal behaviour 53.6 (1997): 1143-1147.

[76] Mosser, Anna, and Craig Packer. "Group territoriality and the benefits of sociality in the African lion, Panthera leo." Animal Behaviour 78.2 (2009): 359-370.

[77] Grinnell, Jon, Craig Packer, and Anne E. Pusey. "Cooperation in male lions: kinship, reciprocity or mutualism?." Animal Behaviour 49.1 (1995): 95-105.

[78] Albers, Paul CH, and Han de Vries. "Elo-rating as a tool in the sequential estimation of dominance strengths." Animal Behaviour 61.2 (2001): 489-495.

[79] Barbuti, Roberto, et al. "An overview on operational semantics in membrane computing." International Journal of Foundations of Computer Science 22.01 (2011): 119-131.

[80] Palagi, Elisabetta, Tommaso Paoli, and Silvana Borgognini Tarli. "Aggression and reconciliation in two captive groups of Lemur catta." International Journal of Primatology 26.2 (2005): 279-294.

[81] Cavigelli, Sonia A., and Michael E. Pereira. "Mating season aggression and fecal testosterone levels in male ring-tailed lemurs (Lemur catta)." Hormones and Behavior 37.3 (2000): 246-255.

[82] Nakamichi, Masayuki, and Naoki Koyama. "Social relationships among ring-tailed lemurs (Lemur catta) in two free-ranging troops at Berenty Reserve, Madagascar." International Journal of Primatology 18.1 (1997): 73-93.

[83] Grimm, Volker, et al. "The ODD protocol: a review and first update." Ecological modelling 221.23 (2010): 2760-2768.

[84] Grimm, Volker, et al. "A standard protocol for describing individual-based and agent-based models." Ecological modelling 198.1 (2006): 115-126.

[85] Hemelrijk, Charlotte K., and Ivan Puga-Gonzalez. "An individual-oriented model on the emergence of support in fights, its reciprocation and exchange." PLoS One 7.5 (2012): e37271.

[86] McLane, Adam J., et al. "The role of agent-based models in wildlife ecology and management." Ecological Modelling 222.8 (2011): 1544-1556.

[87] Macal, Charles M., and Michael J. North. "Tutorial on agent-based modelling and simulation." Journal of simulation 4.3 (2010): 151-162.

[88] Puga-Gonzalez, Ivan, Hanno Hildenbrandt, and Charlotte K. Hemelrijk. "Emergent patterns of social affiliation in primates, a model." PLoS Comput Biol 5.12 (2009): e1000630.

[89] Hemelrijk, Charlotte K. "Self-organization and natural selection in the evolution of complex despotic societies." The Biological Bulletin 202.3 (2002): 283-288.

[90] Hemelrijk, Charlotte K. "An individual-orientated model of the emergence of despotic and egalitarian societies." Proceedings of the Royal Society of London B: Biological Sciences 266.1417 (1999): 361-369.

[91] Hemelrijk, Charlotte K. "Spatial centrality of dominants without positional preference." Artificial Life VI. Vol. 6. 1998.

[92] Gautrais, Jacques, et al. "Deciphering interactions in moving animal groups." PLoS Comput Biol 8.9 (2012): e1002678.

[93] Hemelrijk, Charlotte K., and Hanno Hildenbrandt. "Schools of fish and flocks of birds: their shape and internal structure by self-organization." Interface focus (2012): rsfs20120025.

[94] Som, Christian, Bradley R. Anholt, and Heinz-Ulrich Reyer. "The effect of assortative mating on the coexistence of a hybridogenetic waterfrog and its sexual host." The American Naturalist 156.1 (2000): 34-46.

[95] Hellriegel, B., and H-U. Reyer. "Factors influencing the composition of mixed populations of a hemiclonal hybrid and its sexual host." Journal of Evolutionary Biology 13.6 (2000): 906-918.

[96] Bove, Pasquale, Paolo Milazzo, and Roberto Barbuti. "The role of deleterious mutations in the stability of hybridogenetic water frog complexes." BMC evolutionary biology 14.1 (2014): 107.

[97] Roesli, Marzia, and Heinz-Ulrich Reyer. "Male vocalization and female choice in the hybridogenetic Rana lessonae/Rana esculenta complex." Animal behaviour 60.6 (2000): 745-755.

[98] Reyer, H., Gerhard Frei, and Christian Som. "Cryptic female choice: frogs reduce clutch size when amplexed by undesired males." Proceedings of the Royal Society of London B: Biological Sciences 266.1433 (1999): 2101-2107.

[99] Engeler, Beat, and Heinz-Ulrich Reyer. "Choosy females and indiscriminate males: mate choice in mixed populations of sexual and hybridogenetic water frogs (Rana lessonae, Rana esculenta)." Behavioral Ecology 12.5 (2001): 600-606.

[100] Bergen, Kathrin, Raymond D. Semlitsch, and Heinz-Ulrich Reyer. "Hybrid female matings are directly related to the availability of Rana lessonae and Rana esculenta males in experimental populations." Copeia (1997): 275-283.

[101] Abt, Gaby, and Heinz-Ulrich Reyer. "Mate choice and fitness in a hybrid frog: Rana esculenta females prefer Rana lessonae males over their own." Behavioral Ecology and Sociobiology 32.4 (1993): 221-228.

[102] Tejedo, Miguel, Raymond D. Semlitsch, and Hansjurg Hotz. "Differential morphology and jumping performance of newly metamorphosed frogs of the hybridogenetic Rana esculenta complex." Journal of Herpetology (2000): 201-210.

[103] Hotz, Hansjurg, et al. "Spontaneous heterosis in larval life-history traits of hemiclonal frog hybrids." Proceedings of the National Academy of Sciences 96.5 (1999): 2171-2176.

[104] Anholt, Bradley R., et al. "Overwinter survival of Rana lessonae and its hemiclonal associate Rana esculenta." Ecology 84.2 (2003): 391-397.

[105] Guex, Gaston-Denis, Hansjurg Hotz, and Raymond D. Semlitsch. "Deleterious alleles and differential viability in progeny of natural hemiclonal frogs." Evolution 56.5 (2002): 1036-1044.

[106] Vorburger, Christoph. "Fixation of deleterious mutations in clonal lineages: evidence from hybridogenetic frogs." Evolution 55.11 (2001): 2319-2332.

[107] Semlitsch, Raymond D., et al. "Genetic compatibility between sexual and clonal genomes in local populations of the hybridogeneticRana esculenta complex." Evolutionary Ecology 10.5 (1996): 531-543.

[108] Berger, Leszek. "Systematics and hybridization in European green frogs of Rana esculenta complex." Journal of Herpetology (1973): 1-10.

[109] Kwiatkowska, Marta, Gethin Norman, and David Parker. "PRISM 4.0: Verification of probabilistic real-time systems." Computer aided verification. Springer Berlin Heidelberg, 2011.

[110] Sen, Koushik, Mahesh Viswanathan, and Gul Agha. "Statistical model checking of black-box probabilistic systems." Computer Aided Verification. Springer Berlin Heidelberg, 2004.

[111] Hansson, Hans, and Bengt Jonsson. "A logic for reasoning about time and reliability." Formal aspects of computing 6.5 (1994): 512-535.

[112] Clarke, Edmund M., E. Allen Emerson, and A. Prasad Sistla. "Automatic verification of finite-state concurrent systems using temporal logic specifications." ACM Transactions on Programming Languages and systems (TOPLAS) 8.2 (1986): 244-263.

[113] Clarke, Edmund M., Orna Grumberg, and Doron Peled. Model checking. MIT press, 1999.

[114] Barbuti, Roberto, et al. "AP systems flat form preserving step-by-step behaviour." Fundamenta Informaticae 87.1 (2008): 1.

[115] Cardona, Mónica, et al. "A computational modeling for real ecosystems based on P systems." Natural Computing 10.1 (2011): 39-53.

[116] Barbuti, Roberto, et al. "Simulation of spatial P system models." Theoretical Computer Science 529 (2014): 11-45.

[117] Barbuti, Roberto, et al. "Spatial P systems." Natural Computing 10.1 (2011): 3-16.

[118] Barbuti, Roberto, et al. "Spatial calculus of looping sequences." Electronic Notes in Theoretical Computer Science 229.1 (2009): 21-39.

[119] Barbuti, Roberto, et al. "AP systems flat form preserving step-by-step behaviour." Fundamenta Informaticae 87.1 (2008): 1.

[120] Busi, Nadia. "Using well-structured transition systems to decide divergence for catalytic P systems." Theoretical Computer Science 372.2 (2007): 125-135.

[121] Barbuti, Roberto, et al. "Maximally parallel probabilistic semantics for multiset rewriting." Fundamenta Informaticae 112.1 (2011): 1-17.

[122] Barbuti, Roberto, et al. "The calculus of looping sequences." Formal Methods for Computational systems Biology. Springer Berlin Heidelberg, 2008. 387-423.

[123] Spicher, Antoine, et al. "Stochastic P systems and the simulation of biochemical processes with dynamic compartments." Biosystems 91.3 (2008): 458-472.

[124] Ciobanu, Gabriel, and Laura Cornacel. "Probabilistic transitions for P systems." Progress in Natural Science 17.4 (2007): 432-441.

[125] Pescini, Dario, et al. "Dynamical probabilistic P systems." International Journal of Foundations of Computer Science 17.01 (2006): 183-204.

[126] Barbuti, Roberto, et al. "A probabilistic model for molecular systems." Fundamenta Informaticae 67.1-3 (2005): 13-27.

[127] Madhu, Mutyam. "Probabilistic rewriting P systems." International Journal of Foundations of Computer Science 14.01 (2003): 157-166.

[128] Milner, Robin. Communicating and mobile systems: the π calculus. Cambridge university press, 1999.

[129] Bottoni, Paolo, et al. "Membrane systems with promoters/inhibitors." Acta Informatica 38.10 (2002): 695-720.

[130] Paun, Gheorghe, and Grzegorz Rozenberg. "A guide to membrane computing." Theoretical Computer Science 287.1 (2002): 73-100.

[131] Paun, Gheorghe. Membrane computing: an introduction. Springer Science & Business Media, 2012.

[132] Norris, James R. Markov chains. No. 2008. Cambridge university press, 1998.

[133] Cerone, Antonio, and Marco Scotti."Research challenges in modelling ecosystems." Software Engineering and Formal Methods. Springer International Publishing, 2014. 276-293.

[134] Bernardini, Francesco, and Marian Gheorghe. "Population P systems." J. UCS 10.5 (2004): 509-539.

[135] Bernardini, Francesco, Marian Gheorghe, and Natalio Krasnogor. "Quorum sensing P systems." Theoretical Computer Science 371.1 (2007): 20-33.

[136] Calzone, Laurence, FranÃ§ois Fages, and Sylvain Soliman. "BIOCHAM: an environment for modeling biological systems and formalizing experimental knowledge." Bioinformatics 22.14 (2006): 1805-1807.

[137] Zevedei-Oancea, Ionela, and Stefan Schuster. "Topological analysis of metabolic networks based on Petri net theory." In silico biology 3.3 (2003): 323-345.

[138] Hardy, Simon, and Pierre N. Robillard. "Petri net-based method for the analysis of the dynamics of signal propagation in signaling pathways." Bioinformatics 24.2 (2008): 209-217.

[139] Hayes-Roth, Frederick. "Rule-based systems."Communications of the ACM 28.9 (1985): 921-932.

[140] Pierreval, Henri. "Rule-based simulation metamodels." European Journal of Operational Research 61.1 (1992): 6-17.

[141] shibuchi, Hisao, and Takashi Yamamoto. "Rule weight specification in fuzzy rule-based classification systems." IEEE Transactions on Fuzzy Systems 13.4 (2005): 428-435.

[142] Danos, Vincent, and Cosimo Laneve. "Formal molecular biology" Theoretical Computer Science 325.1 (2004): 69-110.

[143] Danos, Vincent, and Cosimo Laneve. "Core formal molecular biology." European Symposium on Programming. Springer Berlin Heidelberg, 2003.

[144] Wilson-Kanamori, John, et al. "Kappa Rule-Based Modeling in Synthetic Biology." Computational Methods in Synthetic Biology (2015): 105-135.

[145] Pescini, Dario, et al. "Dynamical probabilistic P systems" International Journal of Foundations of Computer Science 17.01 (2006): 183-204.

[146] Andrei, Oana, Gabriel Ciobanu, and Dorel Lucanu. "Executable specifications of P systems" International Workshop on Membrane Computing. Springer Berlin Heidelberg, 2004.

[147] Obtulowicz, Adam. "Probabilistic P systems." Workshop on Membrane Computing. Springer Berlin Heidelberg, 2002.