Thesis Dissertation

MODELING AND ANALYSIS OF ELEONORA'S FALCON POPULATION USING THE UPPAAL STATISTICAL MODEL CHECKER

Constantinos Skitsas

UNIVERSITY OF CYPRUS

COMPUTER SCIENCE DEPARTMENT

May 2018

UNIVERSITY OF CYPRUS
COMPUTER SCIENCE DEPARTMENT

Modeling and Analysis of Eleonora's falcon population using the Uppaal Statistical Model Checker

Constantinos Skitsas

Supervisor
Dr. Anna Philippou

Thesis submitted in partial fulfilment of the requirements for the award of degree of Bachelor in Computer Science at University of Cyprus

May 2018

Acknowledgments

I would like to thank my Supervisor Dr. Anna Philippou for her guidance and support throughout this thesis.

I acknowledge and appreciate the support of my family during my four years at the University of Cyprus.

Abstract

A computer-based model is a computer program, designed to simulate what may happen in the future or what probably happened in a given situation. Basically, it consists of algorithms and equations, which are used to capture the behaviour of the system that is modelled. Simulation is the process of running a model. Computer models are increasingly being used in many fields, including sciences and several aspects of daily life.

As the use of complex systems and our dependence on them is growing, their smooth and reliable operation is of increasing importance. Therefore, when a system is designed, its verification is of utmost importance. This dissertation provides a brief description of the Uppaal SMC model-checker and how it works and also examines how it can be used to aid biologists in the study of the Population Ecology. In Population Ecology, a sub-field of Ecology, the structure and dynamics of populations and how the populations interact with their environment are studied.

In particular, computer modelling could help population ecology by creating a specific model, for the life cycle of one or more species, in this case the Eleonora's falcon, and predicting the future characteristics and survival-chances of the species through simulation.

This thesis concludes that UPPAAL SMC modelling can be a useful tool for biologists in studying population ecology, as long as there is statistical data available, in order to predict future behaviour or situations.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Contents

1.1 Motivation

In the field of population ecology, the structure and dynamics of populations are studied, and how the populations interact with their environment. A population is defined as all of the individuals of the same species within an ecological community.

This field is interested in the growth of a population, variations in population size, the spread of the population and any other interaction with or between it and other populations. The size of a population grows through births and immigration and declines through deaths and emigration [3]. This is how growth rate can be calculated. Other factors that may affect growth are density-dependent factors and density-independent factors. Density-dependent factors are factors that restrict growth based on the density of the population. For example, if the density increases and reaches the maximum consumption of resources an area can provide, the population will level and hit carrying capacity, which is the maximum number of individuals an area can support [4]. Density-independent factors have to do with chance, like a fire for example, which could limit the size of the population. Apart from size, other characteristics of a population are distribution, sex ratio, density and age structure. Distribution is the total area that a population covers. Density is the number of individuals within a certain space. Age structure is the number of individuals in different age classes. Sex ratio is the proportion of males to females.

Population ecology studies these factors and creates models that help to describe what is going on within the population [5]. In order to have a better understanding on how to preserve the endangered species and to be able to propose ways of biodiversity conservation in complicated ecological communities, ecologists need the assistance of useful tools, equipped with graphical user interfaces (GUI), which simplify the visual animation and visual presentation of the output. Ecologists emphasise on the usefulness of modelling demographic and environmental stochasticity in metapopulation dynamics in the investigation of changes affecting the densities of populations in communities as a result of environmental variability [6].

Computer Science can help population ecology by creating a specific model for one or more species' life-cycle. Through simulation, a species future characteristics can be predicted. Such a model can be used to predict population growth and the danger of extinction. Accordingly, the above mentioned factors can be varied in order to find ways to prevent the extinction of the species. This dissertation intends to create a model for the Eleonora's falcon species. The Eleonora's falcon is a medium sized migratory raptor, belonging to the family of Falconidae.

The species is listed as "Least Concern" (has been categorized by the International Union for Conservation of Nature), so it is not in danger of extinction [7]. However, it is interesting to analyse the Eleonora's falcon because of the species' distribution, as 80% of the population lives in Greece [8] and the survival of the population depends on living conditions on the Greek Islands. For example, climate change or the development of infrastructure for tourism in these areas can limit the population size of Eleonora's falcon and increase the danger of extinction.

1.2 Aims of the Dissertation

Some attempts to model Eleonora's falcon life-cycle were carried out previously by students of the University of Cyprus. The primary aim of the first dissertation, that attempted to model Eleonora's falcon life-cycle, was to compare process algebras Palps and S-Palps using the Prism Model Checker through the modeling of Eleonora's falcon life-cycle [9]. The next dissertation attempted to model Eleonora's falcon life-cycle using Uppaal SMC Model Checker [10]. A comparison with Prism was done but only regarding the usability of the tools. The results from both dissertations were that S-Palps is notably faster than Palps and that Uppaal SMC was way easier and simpler than Prism.

The goal of this dissertation is to create a model regarding the Eleonora's falcon population that is more accurate and faster than the models created previously, which used the syntax structure PALPS, a process calculus proposed for the spatially-explicit, individual-based modelling of ecological systems, with a translation to the probabilistic model checker Prism (a synchronous parallel operator).

This dissertation aims to create a model which will be able to simulate a larger number of birds for a longer period of time (in years) compared to previous models. Recent reports suggest that Uppaal SMC is notably faster than Prism, even with an encoding that closely matches that of Prism [11] [12], therefore this model will be created in Uppaal SMC, a model checking approach in the Uppaal family that makes it possible to reason about networks of complicated real-timed systems with a stochastic semantic. More specifically, Uppaal SMC relies on the statistical model checking approach, generalized to deal with real-time systems and detect and evaluate undesirable problems.

Furthermore, this dissertation by using the above mentioned model, aims to examine whether factor changes may aid in the preservation of the Eleonora's falcon species or threaten the survival of the species. An accurate model will be created by studying the behaviour of Eleonora's falcon species through statistical data provided by research, carried out by biologists in Greece, and by understanding its life cycle and the dangers faced by the species.

1.3 Methodology and results

In the beginning, model checking, timed automata and stochastic timed automata had to be studied and understood. Then, the Uppaal SMC had to be studied and information about the Eleonora's falcon species was gathered. Once the above knowledge was acquired, a model in Uppaal SMC that reflects the life cycle of Eleonora's falcon was created. This model was constructed in order to be as precise as possible with Eleonora's falcon's life-cycle. After the first results were produced, the factors were varied in order to find out what conditions may threaten the Eleonora's Falcon species.

1.4 Thesis Structure

In Chapter 2 of this dissertation a brief introduction is provided regarding Model Checking Technique, Transition Systems, Linear Temporal Logic, Computation Tree Logic, Timed automata, Time Computation Tree Logic, and then the main features of Uppaal SMC are analysed. In Chapter 3, information about the Eleonora's falcon and its life cycle is provided. Then, in Chapter 4, there is description on how the model was created and explanation about how the model works. In Chapter 5, the results of the simulations are presented. Finally, in the first part of Chapter 6, the results are discussed in line with the aims of the dissertation and then they are compared to the findings of previous works. Possible future extensions of this dissertation are considered in the second part of Chapter 6.

Chapter 2

Previous Work

Contents

2.1 Model Checking

As reliance of our daily lives on complex systems is growing, their smooth and reliable operation is of increasing importance too. Therefore, when a system is designed, often additional time and effort are consumed on verification than on construction. A formal verification technique which uses exhaustive exploration to verify a system is Model Checking [13]. Model Checking has proven to be a successful technique for verification of such complex systems.

A model checker, with exhaustive exploration, explores all possible states and determines if a specification is true or not. If a specification is not satisfied a counter example will be produced. The model checking process can be divided in three phases. At the beginning the system requirements and design are modeled using the model description language of model checker. The next phase is the specification phase, where all properties are stated using temporal logics. The last phase is the verification procedure, which determines if a specification is satisfied. The most common temporal logics used by model checkers are CTL, PLTL and TCTL [14] [15] [16] [17] [18].

Model Checking is a widely recognized approach to guarantee correctness of a system [19]. However, broader coverage of the systems comes with a drawback, which is that more time is needed for model checking. Furthermore, model checking suffers from the state explosion problem [20]. To avoid state explosion problem, Younes [21] and Sen [22] independently developed an approximate software verification technique. The technique was named Statistical Model Checking (SMC). The main concept of SMC is to generate a number of simulations, count the number of satisfying simulations and apply statistical methods (hypothesis testing) to determine whether the system satisfies the property or not with some degree of confidence. [23].

Model Checking has some advantages when compared to other verification techniques. Firstly, model checking is faster compared to other verification techniques [24]. In addition, the checking process is fully automatic [25] and diagnostic counterexamples are provided in case of a failure [26]. The fact that it covers all the possible cases is essential. Another advantage is that Temporal Logics are able to express in an easy way a lot of the properties that are required for reasoning about concurrent systems. However, one disadvantage is that all Model Checkers experience the state explosion problem. In addition, writing Temporal Logic specifications is hard and sometimes the specification could be complex and difficult to read [24].

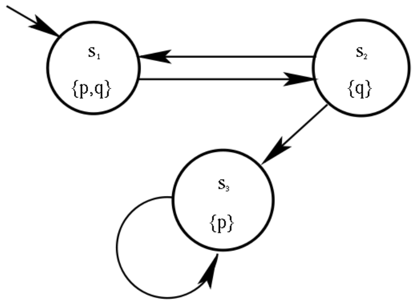

Figure 2.1: Kripke structure

2.2 Transition Systems

Transition systems are models that describe the behavior of systems. They are directed graphs, where nodes represent states and edges represent model transitions [27]. A state describes the information about a system at a specific moment in its behavior. Terminal states of a transition system, are states that do not have outgoing transitions. Transitions specify how the system is able to move from one state to another.

A variation of the transition system, the Kripke structure, which was suggested by Saul Kripke [28], is used in model checking to show the behavior of a system. A Kripke structure is a graph whose nodes represent the reachable states of the system and whose edges represent state transitions. A labelling function maps each node to a set of properties that hold in the corresponding state. Temporal logics(2.3) are usually interpreted in terms of Kripke structures. The formal definition of a Kripke structure is:

Definition 2.2.1 (Kripke Structure [29])**.** Let AP be a set of atomic propositions we define a Kripke structure over AP as a 4-tuple M = (S, I, R, L) consisting of

a finite set of states S.

a set of initial states I $\subseteq$ S.

a transition relation R $\subseteq$ S *times* S such that R is left-total,

a labeling function L: S $\rightarrow$ 2^{AP}.

Considering the Figure (2.1) S is s_1, s_2, s_3, I is $\{s_1\}$, R is $\{(s_1,s_2)$,$(s_2,s_1)(s_2,s_3)$,$(s_3,s_3)\}$ and L is $\{(s_1$, p,q), (s_2, q), (s_3, {p})$\}$. An example path may be s_1, s_2, s_3, s_3, s_3.

2.3 Linear Temporal Logic

Temporal Logic, similar to natural languages and mathematical logic, has syntax and semantics. Syntax determines the rules to build well-formed sentences, whereas semantics determine their meaning. There are sentences that are syntactically correct but incorrect from the semantic point of view. Temporal Logic takes into account grammatically correct sentences and studies whether they are semantically correct. In normal languages the context is given by dictionaries and the informal real-life understanding of the words. In logic, a statement is commonly translated in a given semantic world. If the interpretation is "true" in the given world, then the statement is "satisfied" in the given world . Each logic has a different expressive power. Some properties which could not be expressed in one logic, could be expressed in another. For example Branching-time logic is not able to express some natural fairness properties, which could be without difficulty expressed in the linear-time logic. Linear-time logic is not able express the possibility of a circumstance arising sometime in the future along some computation path [24].

Linear-time temporal logic is a temporal logic that is based on a linear-time perspective. Linear is a temporal logic, with connectives that provide us with the ability to refer to the future. It models time as a sequence of states, extending infinitely into the future. This sequence of states is called a path. Generally, the future is undetermined, so various paths are considered, representing different possible futures, any one of which might be the ‘actual’ path that is realised. Atoms stand for atomic facts that may hold in a system. The Definition via Backus Naur form is,

Definition 2.3.1 (LTL syntax [29])**.** $\phi ::= \bot \,|\, \top \,|\, p \,|\, (\neg\phi) \,|\, (\phi \wedge \phi) \,|\, (\phi \vee \phi) \,|\, (\phi \rightarrow \phi) \,|\, X\,\phi \,|\, F\,\phi \,|\, G\,\phi \,|\, \phi\, U\, \phi \,|\, \phi\, W\, \phi \,|\, \phi\, R\, \phi$

where p is any propositional atom from the set Atoms. The connectives X, F, G, U, R, and W are the temporal connectives. X stands for ‘next state,’ F stands for ‘some Future state,’ and G stands for ‘all future states (Globally)’. The next three, U, R and W are called ‘Until’, ‘Release’ and ‘Weak-until’ respectively. In addition to textual forms, connectives also have a symbolic form. For example, the symbolic form for ‘X’ is $\bigcirc$, ‘F’ is $\Diamond$ and ‘G’ is $\Box$.

Definition 2.3.2 (Transition System [29])**.** A transition system M = (S, $\rightarrow$, L) is a set of states S endowed with a transition relation $\rightarrow$ (a binary relation on S), such that every s $\in$ S has some s' $\in$ S with s $\rightarrow$ s' , and a labelling function L: S $\rightarrow$ P(Atoms).

Definition 2.3.3 (Path [29]). A path in a model M = (S, $\rightarrow$, L) is an infinite sequence of states s_1, s_2 , s_3, ... in S such that, for each i $\geq$ 1, $s_i \rightarrow s_{i+1}$. We write the path as $s_1 \rightarrow$ s2 $\rightarrow$ ...

Definition 2.3.4 (Semantics of LTL [29]). Let M = (S, $\rightarrow$, L) be a model and π = s1 $\rightarrow$ ... be a path in M. Whether π satisfies an LTL formula is defined by the satisfaction relation $\models$ as follows:

1. $\pi \models \top$
2. $\pi \models \neg$ if not $\pi \models \Phi$
3. $\pi \models$ p if p $\in L(s_1)$
4. $\pi \models \phi_1 \wedge \phi_2$ if $\pi \models \phi_1$ and $\pi \models \phi_2$
5. $\pi \models \phi_1 \vee \phi_2$ if $\pi \models \phi_1$ or $\pi \models \phi_2$
6. $\pi \models \phi_1 \rightarrow \phi_2$ if $\pi \models \phi_2$ whenever $\pi \models \phi_1$
7. $\pi \models$ X ϕ if $\pi^2 \models \phi$
8. $\pi \models$ G ϕ if, for all i $\geq$ 1, $\pi^i \models \phi$
9. $\pi \models$ F ϕ if, there is some i $\geq$ 1 such that $\pi^i \models \phi$
10. $\pi \models \phi$ U ψ if there is some i $\geq$ such that $\pi^i \models \psi$ and for all j= 1, ..., i-1 we have $\pi^j \models \phi$
11. $\pi \models \phi$ W ψ if either there is some i $\geq$ such that $\pi^i \models \psi$ and for all j= 1, ..., i-1 we have $\pi^j \models \phi$; or for all k $\geq$ 1 we have $\pi^k \models \phi$
12. $\pi \models \phi$ R ψ if either there is some i $\geq$ such that $\pi^i \models \phi$ and for all j= 1, ..., i-1 we have $\pi^j \models \psi$; or for all k $\geq$ 1 we have $\pi^k \models \psi$

From the above definition, sentence one means that among path π the formula is always true and sentence two that Φ is not satisfied among path π. Sentence four means that path π has to always satisfy ϕ_1 and ϕ_2 and sentence five that path π has to satisfy ϕ_1 or ϕ_2. Sentence 7 means that path π starting from the second state has to satisfy ϕ. Sentence 8 means that for all possible paths π, ϕ has to be always true. The ninth sentence with the connective 'F' expresses that for at least one path π, ϕ has to be always true. The next sentence means that among path π, ϕ has to be true until ψ becomes true.

2.4 Computation Tree Logic

Branching Time Temporal logic is based on a branching notion of time and not on a linear notion of time. Branching time is related to the fact that at each specific moment there could be a number of different possible futures. The temporal operators in branching temporal logic allow the expression of properties of computations of a system.

Computation Tree Logic (or CTL) is a branching-time logic, and as its name suggests is a tree-like structural model of time, in which the future is undetermined; there exist different paths in the future, any one of which could be the 'actual' path that is realised [29].

The definition via Backus Naur form is,

Definition 2.4.1 (Syntax CTL [29])**.** $\phi ::= \bot \,|\, \top \,|\, p \,|\, (\neg\phi) \,|\, (\phi \wedge \phi) \,|\, (\phi \vee \phi) \,|\, (\phi \rightarrow \phi) \,|\, AX\,\phi \,|\, EX\,\phi \,|\, AF\,\phi \,|\, EF\,\phi \,|\, AG\,\phi \,|\, EG\,\phi \,|\, A[\phi\, U\, \phi] \,|\, E[\phi\, U\, \phi]$

It can be noticed that there are pairs of symbols. The first letter of each pair is 'A' or 'E'. 'A' means for all paths and 'E' means there exists at least one path. The next letter of each pair may be X, F, G and U. 'X' means next state, 'F' some future state, 'G' globally all future states and 'U' until. The binding priorities for the CTL connectives are, the unary connectives (consisting of $\neg$ and the temporal connectives AG, EG, AF, EF, AX and EX) bind most tightly. Next in order come $\vee$ and $\wedge$ and after that $\rightarrow$, AU and EU.

Definition 2.4.2 (Semantics of CTL [30])**.** Let M = (S, $\rightarrow$, L) be a model for CTL, s in S, ϕ a CTL formula. The relation M, s $\models \phi$ is defined by structural induction on ϕ:

1. M, s $\models$ p if p $\in$ Label(s)

2. M, s $\models \neg\phi$ if not item M, s $\models \phi$

3. M, s $\models \phi \vee \psi$ if only M, s $\models \phi$ or M, s $\models \psi$

4. M, s $\models$ E ϕ if M, w $\equiv \phi$ for a path w that starts from s

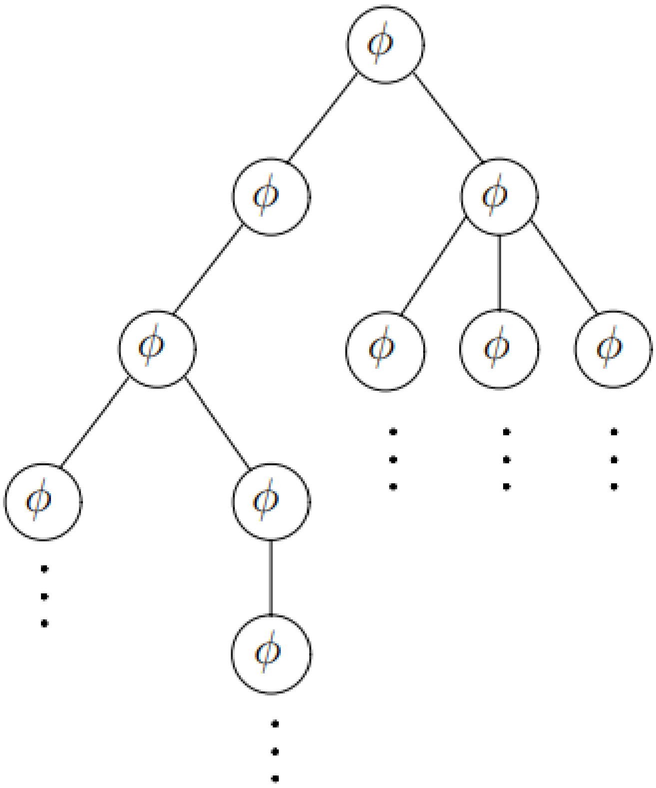

Figure 2.2: A system whose starting state satisfies AG ϕ

5. M, s $\models$ A ϕ if M, w $\equiv$ ϕ for every path w that starts from s

Fom the above definition M, w $\equiv$ ϕ only if formula ϕ is satisfied in path w of structure M [30].

2.5 Timed automata

Timed automata (TA) were introduced as an extension of the automata-theoretic approach to the modeling of real-time systems by Alur and Dill [31]. They model the behaviour of time-critical systems. Time-critical systems are systems whose correctness depends on the logical result of the computation and also on the time in which the results are produced. In fact, a timed automaton is a program graph with a finite set of clocks. It can be assumed that clocks are like stopwatches. Clocks can only be inspected or reset to zero. Clocks are used to formulate the real-time assumptions about system behavior. An edge in a timed automaton is labelled with guard, action and a set of clocks. A location invariant constrains the amount of time that can be spent on a location. The amount of time that can be spent on a location is called delay. For modeling complex systems, parallel composition of timed automata is used with a set of handshaking actions. Important properties for timed automata are time-lock, zenoness, time divergence, reachability, safety and liveness. A state ‘s’ contains a time-lock in which there exist no divergent paths starting from ‘s’ state. Time divergence in a path is when the sum of the delays over this path is infinite. Zenoness is when infinitely many actions take place in finite time.

In the theory of Timed Automata, actions occur in zero time. This means, that nothing precludes executions of infinitely many actions in finite time. That is, a timed automaton may have time-convergent paths with an infinite number of actions. A time automaton is non-zeno if there does not exist an initial zeno path in transition system TS (TA). Time-locks and zeno paths should be avoided. Reachability properties ask whether a given state formula ϕ, can be satisfied by any reachable state or whether there exists a path starting at the initial state such that ϕ is eventually satisfied along that path [32]. Reachability properties do not guarantee the correctness, but do validate the basic behaviour of the model. Safety properties are of the form "something bad will never happen". Liveness properties are of the form that something will eventually happen.

Definition 2.5.1 (Time Automaton [29])**.** A time automaton is a tuple $TA = (Loc, Act, C, \hookrightarrow, Loc_0, Inv, AP, L)$ where

- Loc is a finite set of locations, $Loc_0 \subseteq Loc$ is a set of initial locations,
- Act is a finite set of actions,
- C is a finite set of clocks,
- $\hookrightarrow \subseteq Loc \times CC(C) \times$ Act $\times 2^C \times Loc$ is a transition relation,
- Inv:Loc$\rightarrow CC(C)$ is an invariant-assigment function,
- AP is a finite set of atomic propositions, and
- L:Loc $\rightarrow 2^{AP}$ is a labeling function for the locations

Considering the Figure 2.3 the informal syntax of this Timed automaton is the following:

- Locations are C and S

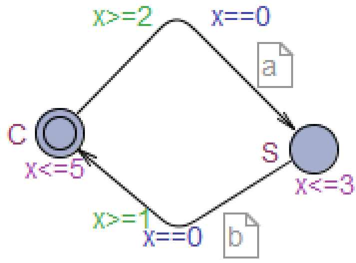

Figure 2.3: A timed automaton created on Uppaal

- Initial Location is C
- Actions are a and b
- Clock is x
- Guards is x>=2 and x>=1
- Invariant is x<=5 and x<=3
- Clock Reset is x==0

Guards, invariant and Clock reset are written in Uppaal and they seem different from the general Timed Automata syntax. For guards x>=2 is equal with $x \geq 2$, for invariant x<=3 is equal with $x \leq 3$ and for clock reset x==0 is equal with $x := 0$

2.6 Timed Computation Tree Logic

In CTL we can write a formula EF p, which means along some computation path, p eventually becomes true. CTL does not provide a way to put a bound on the time at which p will become true. A natural and simple extension is to put subscripts on the temporal operator. This approach, is used to introduce explicit time in the syntax of Timed Computation Tree Logic (TCTL) [33].

TCTL is a logic to reason about Timed Automata. TCTL is a real-time variant of CTL which aims to express properties of timed automata. Timed CTL extends CTL with atomic clock constraints over the clocks in C. TCTL also adds branching time: 'A' and 'E' are called path quantifiers. They enable us to state that, in a given state a property must hold for all paths starting in the state or for some path starting in the state. The until operation is equipped with an interval J of real numbers [27].

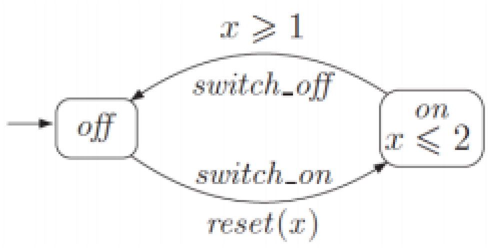

Figure 2.4: Light Switch Timed Automaton

Definition 2.6.1 (Syntax of TCTL [27])**.** Formulae in TCTL are either state or path formulae. TCTL state formulae over the set AP of atomic propositions and set C of clocks are formed according to the following grammar:
$\Phi ::= true \,|\, a \,|\, g \,|\, \Phi \wedge \Phi \,|\, \neg\Phi \,|\, E\phi \,|\, A\phi$
where a $\in$ AP, g $\in$ ACC(C) and ϕ is a path formula defined by: $\phi ::= \Phi U^j \phi$
where J $\subseteq$ R$\geq$0 is an interval whose bounds are natural numbers.

Definition 2.6.2 (Semantics of TCTL [27])**.** Let the TA be a timed automaton, a $\in$ AP, g $\in$ ACC(C), and J $\subseteq$ $IR_{\geq 0}$. For state s = $\langle \lambda, \eta \rangle$ in TS(TA), state formula Φ and Ψ, and path formula ϕ the satisfaction relation $\models$ is defined by:

1. s $\models$ true
2. s $\models \alpha$ if $\alpha \in L(\lambda)$
3. s $\models$g if $\eta \models$ g
4. s $\models \neg\Phi$ if s not $\models \Phi$
5. s $\models \Phi \bigwedge \Psi$ if s$\models \Phi$ and s $\models \Psi$
6. s $\models \exists\phi$ if $\pi \models \phi$for some $\pi \in Paths_{div}(s)$
7. s $\models \forall\phi$ if $\pi \models \phi$for all $\pi \in Paths_{div}(s)$

Examples of TCTL: Consider the Light Switch automaton (Figure 2.4), with the two states on and off. We can express the properties “the light cannot be continuously switched on for more than 2 minutes” and “the light will stay on for at least 1 time unit and then switch off” with
$\forall\square(on \rightarrow \forall\Diamond^{>2}\neg$ on) and $\forall\square(on \wedge (x = 0)) \rightarrow (\forall\square^{<=1} on \wedge \forall\Diamond^{>1}$off) respectively.

2.7 Uppaal

2.7.1 Introduction

Uppaal is a tool box for the modeling, simulation and verification of real-time systems modelled as networks of timed automata. It was developed by Uppsala University and Aalborg University and it was first released in 1995 [34]. Uppaal consists of three parts: a description language, a simulator and a model-checker [35]. The description language is used to facilitate modeling in graphical and textual formats. The modeling formalism extends basic timed automata with discrete variable over basic, structured and user-defined types that can be modified by user-defined functions written in an Uppaal specific, C-like imperative language. The simulator allows the dynamic behaviour of a system to be examined. For example the user is able to trace through a graphical simulator a specific execution trace, which may result in a system error. Model checking is designed to check for invariant and reachability properties. Using the verifier it can be checked if a property is satisfied or not.

2.7.2 Timed automata in Uppaal

Uppaal modeling language extends timed automata with some additional features, which are shown below:

- Template: Automata are defined with a set of parameters. These parameters are substituted with given argument in the process declaration.
- Constants: Integers, booleans and arrays over integers and booleans can be marked constant by prefixing the type with the keyword const. For example they can be declared using **const** name value, for example **const** delay 2.
- Bounded integer variables: Declaration is **int**[min, max] name. Guards invariants and assignments may contain expressions ranging over bounded integer values.
- Binary synchronization: Declaration is chan c. There are two types of channel c! and c?. An edge labelled with c! has to be synchronized with another edge labelled with c?
- Broadcast channels: Declaration is broadcast chan c. In broadcast channel a sender c! can synchronize with an arbitrary number of receivers c?. A sender always

continues its execution even if there is no receiver (non blocking). A broadcast channel can be used with an id, for example `channel c[id]!` will synchronize with the `channel c[id]?` and communication via specific template only can be accomplished.

- Urgent Location: Time is not allowed to pass when a process is in an urgent location.
- Committed Location: More restrictive than urgent location. If any process is in a commitment location the next transition must involve an edge from one of the committed locations.
- Arrays: Arrays are allowed for clocks, channels, constants and integer variables.
- Record types: Declaration is like C with the word **struct**.
- Custom types: Declaration is like C with the word **typedef**.
- Functions: Can be defined locally or globally. Locally for specific template or globally for every template. Templates parameters are accessible from local functions. Syntax is similar to C without pointer.
- Urgent synchronization: Delays cannot occur if a channel is declared urgent and edges that use urgent channels cannot have time constrains

Expressions in Uppaal are Select, Guard, Synchronization, Update and Invariant.

2.7.3 Simulator

The simulator can be used in 'random mode', 'manual mode' and 'traced mode'. In random mode the system will run on its own: the user can only modify the simulation speed. Using the option "manual simulation", the user can manually choose which transitions to take. The last option (traced) is to import a simulation from verifier or use a saved simulation and go through a trace to see how certain states are reachable. The control part is used to choose and fire enable transitions and the variable view shows the variable value of the current state.

2.7.4 Verifier

The user can model-check one or various properties and insert or remove properties. Status shows the connection with the server, the time and the memory needed to check the

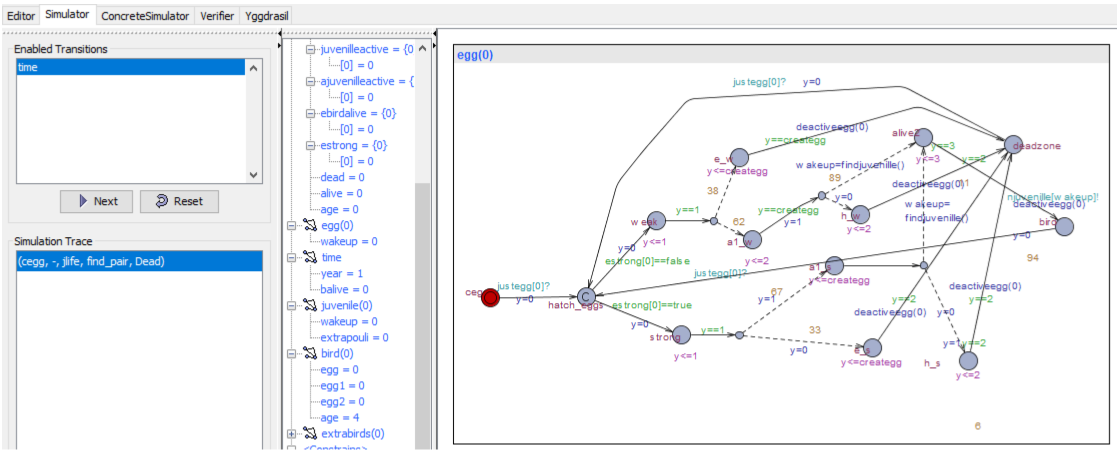

Figure 2.5: Example of simulator

property. On the overview list, a green light will appear if the property is satisfied and a red one if not. Reachability, safety and liveness properties can be checked. In Uppaal extension SMC, additional properties like Probability Estimation, Hypothesis Testing and Probability Comparison can be computed and checked. Uppaal specification language is Time Computation Tree Logic.

2.7.5 Uppaal SMC

Uppaal SMC is an extension of Uppaal which proposes to represent systems via networks of automata whose behaviours may depend on both stochastic and non-linear dynamic feature [36]. Uppaal SMC is a model checker that uses the SMC verification approach. This approach was explained in Paragraph 2.1 above. As mentioned above (Paragraph 2.1), SMC was proposed to solve the state explosion problem that was caused from the large number of states in complex systems. However, SMC also faces problems with complex systems, when the results demand a high level of confidence. To achieve the high level of confidence, a high number of simulation runs is needed and that causes this technique to be extremely time consuming.

Some important additional features of Uppaal SMC are the weighted discrete probabilistic transitions, rate of expotential, statistical properties and statistical parameters. The weight over branch is a non-negative integer denoting (that express) the probabilistic likely-hood of the branch being executed. The probability of a particular branch is determined as a ratio of its weight over the sum of weights of all branches emanating from the same branch node. The Rate of exponential is a ratio expression which specifies the Rate of exponential Probability distribution. Some statistical properties are Probability Estimation, Value Estimation, Hypothesis Testing (Qualitative Model Checking) and Probability Comparison. With Statistical parameters menu option, parameters can be changed. The lower probabilistic deviation ($-\delta$) specifies the lower bound of indifference region from the specified probability and is used in hypothesis testing. In hypothesis testing is also used, the upper probabilistic deviation ($+\delta$) which specifies the upper bound of indifference region of the specified probability. The probability of false negatives (α) and the probability of false positives(β) are both used in probability estimation and hypothesis testing and specify the level of significance. Probability uncertainty (ε) constrains the probability interval and is used in probability estimation. The last two, Ratio lower(u_0)/upper(u_1) bound are used in comparison of two probabilities [37].

As mentioned above SMC verifier can provide, Probability Estimation, Hypothesis Testing, Probability Comparison and Probability Confidence Interval Estimation [36]. The Probability Estimation algorithm computes the number of runs needed in order to produce an approximation interval $[p-\varepsilon, p+\varepsilon]$ for $p = P(\psi)$ with a confidence $1-\alpha$. Hypothesis Testing reduces the qualitative question to e test the null-hypothesis. Probability Estimation can be calculated with the query $Pr[bound](\phi)$, Probability Comparison with $Pr[bound1](\phi 1) >= Pr[bound2](\phi 2)$ and Hypothesis Testing with $Pr[bound](\phi) >= p_0$ $H : p = Pm(\phi) >= \theta$ against the alternative hypothesis $K : p = Pm(\psi) < \theta$ (where ϕ is the formula) [36].

Uppaal SMC provides the opportunity for dynamic creation of processes, with declaration of dynamic templates. Creation of a template is achieved with the word `spawn Template()` and a template can only be destroyed by itself-with the word `exit ()`. Unfortunately, Uppaal SMC does not support dynamic templates with simulator, so there is a trade off, using dynamic templates instead of static templates.

Chapter 3

Eleonora's falcon

Contents

3.1 General Information

Eleonora's falcon is a rare, medium-sized, migratory raptor. It was named after Giudicessa Eleonora de Arborea, a female judge in Sardinia (1350-1404), who protected this species. [38].

Greece is thought to be the most important country for the conservation of Eleonora's

Kingdom	*Phylum*	*Class*	*Order*	*Family*
Animalia	Chordata	Aves	Falconiformes	Falconidae

Table 3.1: Taxonomy of Eleonora's falcon

BIOMETRICS:
Length: 36-42 cm
Wingspan: M: 90 cm – F: 105 cm
Weight: M: 350 g – F: 390 g

Table 3.2: Biometrics of Eleonora's falcon

falcon species, as Greece hosts more than 80% of the global population during the breeding season [8]. In Cyprus, Eleonora's falcon is on the strictly-protected species list of Law No 24 of 1988 (the law ratifying the Convention for the Conservation of European Wildlife and Natural Habitats, the Bern Convention) [39].

The three cliffs in Cyprus that are occupied by the species have been IBAs (Important Bird and Biodiversity Areas) since 1988. Episkopi and Akrotiri cliffs are within the U.K. Sovereign Base Area and are Permanent Game Reserves. Cape Aspro cliffs are not a Game Reserve area but are not accessible from land and, hence, are naturally protected [40]. The three criteria that the International Union for Conservation of Nature uses to set the level of concern for species are: range size, population trend and population size. Regarding the range size criterion, Eleonora's falcon has a very big range and therefore does not reach the thresholds for "vulnerable". The population size is moderately small to large so it does not approach the thresholds for population size criterion. The last criterion population trend, Eleonora's falcons population seems to be increasing so it is not considered as "vulnerable" under the population trend criterion. For the above mentioned reasons the Eleonora's falcon is evaluated as of "Least Concern" [7]. Eleonora's falcons, are essential bio-indicators of healthy environments and further help to decrease pest species, such as grasshoppers and rodents, which cause damage to human crops.

3.2 Migration

Eleonora's falcons are largely migratory and usually travel long distances in groups [41]. They abandon their breeding colonies between mid-October and mid-November to arrive at their wintering grounds in Madagascar, where the rainy season brings plenty of insects [42]. They go back to their breeding grounds in late April/May. Since the 1950s, it has been believed that Eleonora's falcons follow a species-specific migration route, taking

Figure 3.1: Eleonora's falcon

them down the entire Mediterranean towards Suez [43], down the Red Sea coast, around the Horn of Africa and along the East African coast, before arriving in Madagascar. With the help of technology and especially satellite telemetry, the migration route of Eleonora's falcons can be tracked. Figure 3.2 shows four tracked birds, showing that Eleonora's falcons migrate across Africa.

3.3 Breeding reproduction

Eleonora's falcon is a monogamous falcon that breeds once a year and later than almost any other northern hemisphere bird. The reason for this late breeding is the abundant food supply (migratory songbirds) during this period. Breeding sites are occupied starting in late April, though breeding does not start until late July. The young hatch in late August to early September. The nest is located on the ground or on a cliff, often in a small cavity or under a small bush, sheltered from wind. Normally 2 or 3 eggs are laid [44]. Incubation lasts between 28 and 30 days, and the young fledge after another 35 to 40 days [45]. It is believed that experienced falcons tend to be more successful than first year breeders. The reason for this is that, more experienced males will select better nesting sites, will hunt more efficiently and thus will mate with the best quality females. Average productivity differs among different colonies and can range from 1.26 fledglings per year to 2.6 young per year. Young males usually remain near their parents' home range, while females disperse further.

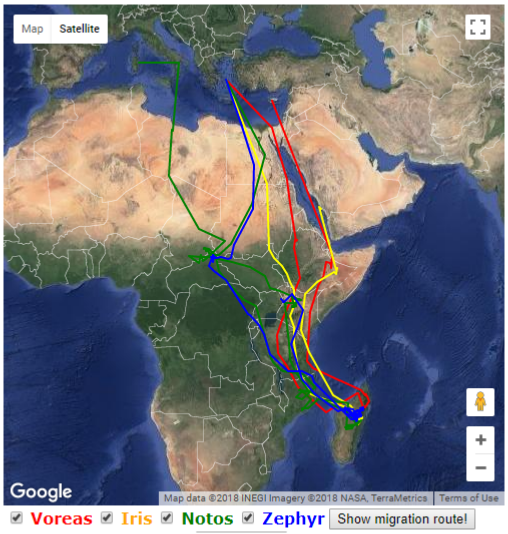

Figure 3.2: Migration route of 4 tracked Eleonora's falcons from October 2009 to October 2010 [1]

3.4 Food and hunting

Eleonora's falcons live in groups and are not often seen hunting alone. Until August they feed on large insects, especially beetles. Later on, they start hunting small birds, migrating to the south, which provide an ideal food for their chicks [46]. Eleonora's falcons keep some of their captured prey alive, by keeping or 'imprisoning' some small birds in a relatively deep cavity or rock fissure. To ensure that their prey cannot escape, the falcons either pull out all flight (both wing and tail) feathers before placing them in the fissures, or alternatively keep them 'trapped' in a tight and deep hole which renders them unable to move. Keeping prey alive for one or two days will provide the falcon with fresh food when needed, because dead prey brought to the nest but unconsumed is wasted, as it dries out too quickly.

Incidents reported in Cyprus, have shown, cannibalistic behaviour. Eleonora's falcons do not usually have the opportunity to steal a nestling from another nest, as the parents will be protecting their young. However, given the opportunity, when parents leave the nest for a moment, a nestling is easy prey, due to its limited strength to fight back, and the fact that it cannot fly to escape [47].

3.5 Threats

Threats to Eleonora's falcon consist mostly of human disturbance at its breeding sites. Modern transport and the development of infrastructure for tourism have brought these sites that used to be inaccessible, within easy reach of tourist resorts. Human disturbance near colonies is considered to make the falcons abandon their eggs, or to move to more remote sites [48]. Additionally, deforestation and intensive agriculture in Madagascar can harm the falcon. Introduction of other species to the breeding islands also poses a threat, for example with introduced cats and rats feeding on eggs, as well as young and adult birds, and introduced livestock disturbing the birds from their nests [40] [49].

3.6 Eleonora's falcon life-cycle analysis

The information provided below regarding the Eleonora's falcon is based on 2 articles. The first article is ‘Variation in breeding parameters of Eleonora's falcon (Falco eleonorae) and factors affecting its reproductive performance’. It embodies of a four-year study of Eleonora's falcons on Greek Islands. Scientists in Greece were able to observe 690 breeding pairs [2]. The second article is ‘Synchronous Parallel Composition in a Process Calculus for Ecological Models'. This paper was an attempt to model the Eleonora's Life-cycle in Prism [50]. The population of Eleonora's falcon in the world is concentrated in a small number of colonies, which means that the loss of one colony could have a large impact on the world population of the falcon.

The species overwinters in Madagascar and East Africa and breeds colonially during late summer on rocky cliffs on uninhabited islets of the Mediterranean. Aerial displays by male falcons begin as soon as the birds arrive at nesting sites. Every pair of Eleonora's falcons lays one to three eggs [51]. The expected number of eggs per pair is 2.43. Incubation lasts between twenty eight to thirty days and the rate of success is 75% [52]. Fledging takes thirty five to forty days and the survival rate for fledglings is 0.92%. The total breeding success is estimated at 0.60%. When the breeding period is over, Eleonora's falcons return to Madagascar.

The young Eleonora's Falcons reach sexual maturity at three years old. The mortality rate before reaching sexual maturity is 78% [53] [54]. The mortality rate in adults is 18% per year [55]. This species is not threatened in Madagascar, but Eleonora's falcon is clas-

Breeding parameters	2004	2005	2006	2007	Total
No. of breeding pairs	78	165	243	204	690
No. of successful pairs	65	145	183	154	547
No. of eggs	183	393	614	485	1,675
No. of hatchlings	126	294	364	305	1,089
No. of fledglings	125	289	342	243	999
Clutch size	2.35 ± 0.64	2.38 ± 0.59	2.53 ± 0.60	2.38 ± 0.63	2.43 ± 0.62
Hatching success	0.69 ± 0.38	0.75 ± 0.34	0.59 ± 0.39	0.61 ± 0.40	0.64 ± 0.39
Fledging success[a]	0.80 ± 0.40	0.85 ± 0.36	0.70 ± 0.44	0.60 ± 0.47	0.72 ± 0.44
Breeding success	0.68 ± 0.38	0.74 ± 0.34	0.55 ± 0.38	0.50 ± 0.41	0.60 ± 0.39
Productivity	1.60 ± 0.94	1.75 ± 0.87	1.41 ± 0.99	1.19 ± 1.01	1.45 ± 0.99
Flight rate	1.92 ± 0.67	2.01 ± 0.60	1.90 ± 0.63	1.84 ± 0.60	1.92 ± 0.62

[a]Including nests with 0 eggs hatched

Figure 3.3: Breeding parameters and mean success rates of Eleonora's falcon colonies in the Aegean Sea [2]

sified as rare in Europe and loss of forest habitat and persecution pose a minor threat to it [56]. In figure 3.4 Eleonora's falcons' life-cycle is presented. The possibility of death is presented in the figure. Experienced adult falcons and non experienced adult falcons are separated in breeding. The reason is that experienced adult falcons will select better nesting sites and will have better productivity rate. Productivity rate will have effect for an offspring to survive. Juveniles migrate like adult falcons but do not breed and migrate back to Madagascar. In the occasion that a juvenile is three years old it starts the adult falcon life-cycle.

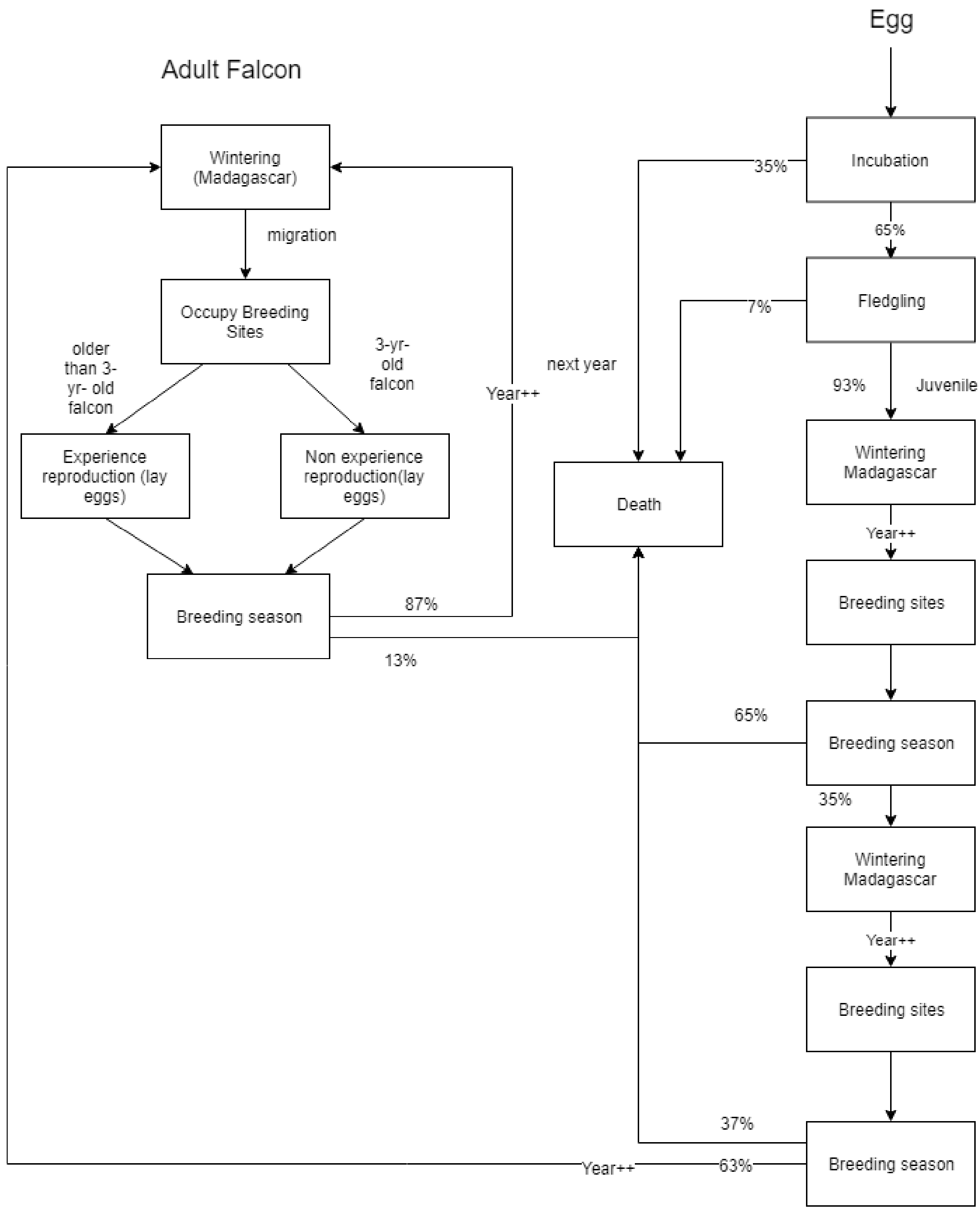

Figure 3.4: Eleonora's Life-cycle, separated in Adult Falcons and eggs

Chapter 4

Model creation

Contents

4.1 Model explanation

The Model is based on five synchronized templates which are birds, extrabirds, egg, time and juvenile. Communication among templates is accomplished via broadcast channels. Global arrays for every template to check active/inactive were used. There is one boolean array for each template (see figure 4.1). Each template has a unique identification, which is the template's position in the array. If the template is active the value in the array position will be true, otherwise it will be false. Static instantiation was chosen instead of dynamic instantiation. The reason was that with static instantiation testing was easier and more efficient, due to the capability of using simulator rather than using verifier queries to discover possible mistakes. Figures 2.5 and 4.2, show the benefits of using the simulator for testing the model behavior. Some of these are: variable values at any possible moment

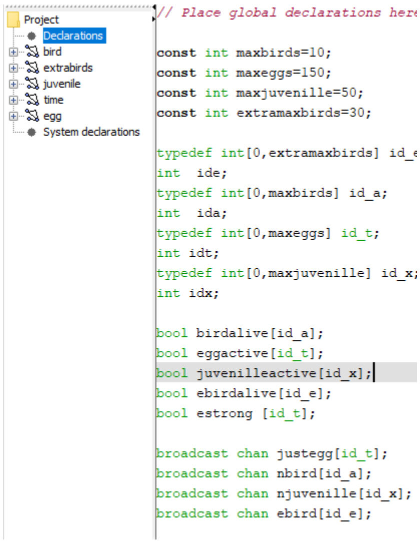

Figure 4.1: Global variables

can be checked and synchronization between templates and unexpected behaviour can be seen via the graphical representation of templates. If there is an incorrect estimation of falcons, eggs or juveniles that may be needed for simulations, a wrong message will be returned. The trade-off between using static instantiation instead of dynamic instantiation of templates was more memory consumption.

As mentioned above testing of the model was done with Uppaal Smc simulator (see paragraph 2.7.3) manual and random trace. With hypothetical scenarios we tested the behaviour of the model for many years. With manual trace, we could select which available transition to take, so we were able to create scenarios that cover almost everything that may happen within the model. With random mode, random scenarios were generated from Uppaal. Figure 2.5 shows an example where the Simulator is used for only one Template. When we run the simulator all the templates are presented, each one in the form of Figure 2.5. The red color in Figure 2.5 indicates the "active" state for each template. Enabled transitions for all templates are shown in the enabled transitions window. "Stopping" the time in critical moments, like the creation of an egg, aided in the verification of the model correctness.

4.1.1 Bird template

The 'Birds' template reflects the adult falcon's life-cycle. The constant maxBirds in declaration initialises the sample of birds. 'find_pair' is the initial state and uses the function initialize () to ensure that the bird is alive and active for calculations. The next state sep-

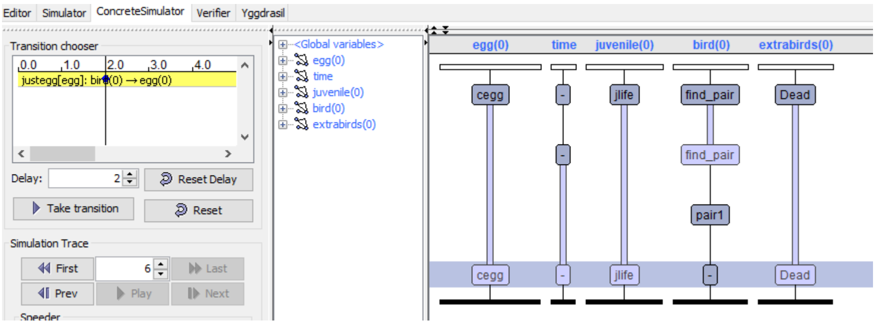

Figure 4.2: Example of concrete simulation

arates males and females with probabilities of 50% each. Females will continue to lay eggs and males will wait until the breeding period is over. In pair state the simulation will continue with 45% probability to lay two eggs, 6% to lay one egg and 49 % to lay three eggs. The eggs are created with a broadcast channel, with their id using `findegg(id)` function to find available eggs and wake them up. If the falcon is four years old or older, the falcon will continue to experienced reproduction and the egg will be strong, otherwise the falcon will continue to nonexperienced reproduction and the egg will be weak. After that, birds from both states will meet in 'immigration' state. In immigration state, there is a possibility of 13% that a falcon will die and 87% that it will survive. If the bird dies, it will go to death state, otherwise it will go to 'find_pair' state and age will be increased. That is the loop for the birds' template, which is displayed in Figure 4.3.

4.1.2 Extra Birds template

The 'Extra birds' template is similar to the 'birds' template, with small differences. Constant "extramaxbirds" is the maximum number of adult falcons. The initial state is 'death', waiting for a signal to make it active and move to the next state. An extra state has been added between the 'Death' state and the 'find_pair' state. This state's only purpose is to save memory. To be more specific in the 'juveniles' template there are juveniles from one year old to three years old. It is synchronized with the new adult falcons (from 'juvenile' template to 'extrabirds' template), it will meet falcons in 'find_pair' state when falcons finish breeding and immigration. The same thing happens with the creation of a juvenile, (from 'egg' template to 'juvenile' template). The new juvenile will start when the adult falcons are in 'find_pair' state. The function that searches for free templates always happens in the state previous to the one that sends the broadcast channel. That causes the problem of having juveniles for three years instead of two. The problem was solved using the extra state in the 'extrabirds' template, that creates the adult falcons before the new

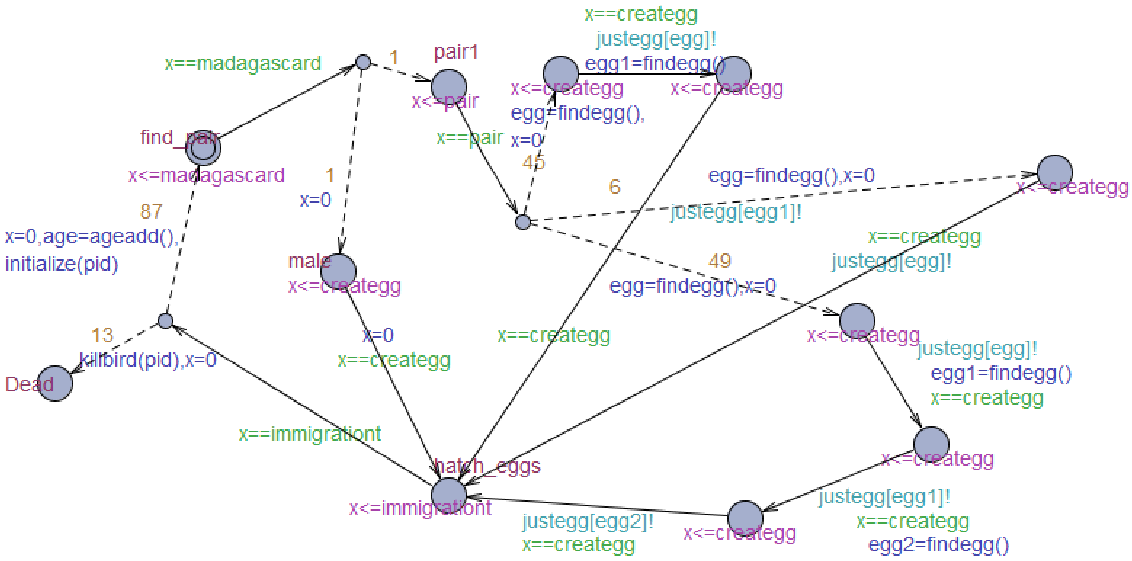

Figure 4.3: Bird Template

search function occurs (when we use the word search function we refer to every function with main purpose to find free templates, those functions are findebird (), findegg (), findjuvenille (). The adult falcons just wait in the extra state for two periods of time, reset the clock and meet the others in 'find_pair' state.

4.1.3 Egg Template

This template models life from incubation until the fledging. All eggs are in 'cegg' state, waiting for a signal to proceed to 'hatch egg' state. 'Hatch egg' state is a commitment state and has to move to the 'weak' or 'strong' state without delay. There is a guard which checks if the egg is strong or weak in order to choose the state. The model is as described in Section 3.6, divided into strong/weak and egg lose/hatch lose. If the egg fails to become a juvenile, it goes to 'deadzone' state, it becomes inactive and waits for a signal to start over. If it succeeds it will create a juvenile (from a function that will find an available juvenile and send a broadcast channel to wake it up). The next state is 'bird' state, where it will be inactive waiting for a broadcast channel to start over.

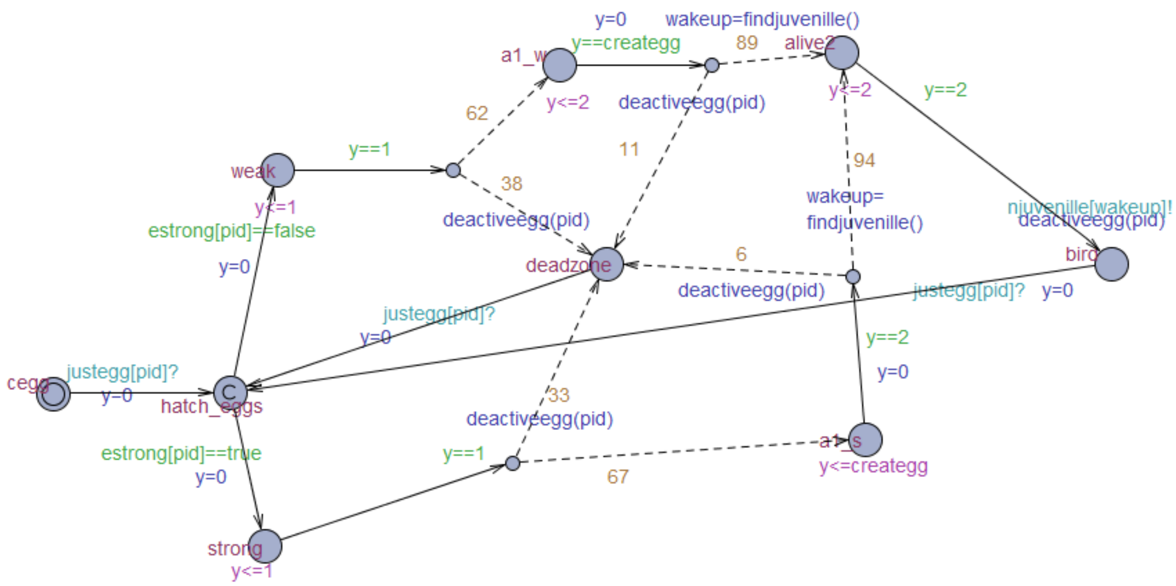

Figure 4.4: Egg Template

4.1.4 Juvenile template

This template models the life of a juvenile from the time that it leaves the breeding site until the third year of its life. The initial state is ‘jlife’ and waits for a signal. Once the signal is received it moves to ‘firstyear’ state and waits for a year. Then with probability, it continues to ‘secondyear’ state and it stays there for another year. Again with probability, it continues to the next state. The next state is a commitment state which sends a broadcast channel to wake up an available bird. The findebird () function (a function that looks for available birds) sets the age to three years old for the bird that will wake up. Then it returns to ‘jlife’ state and waits for a signal to start over. If the juvenile dies in the ‘firstyear’ or ‘secondyear’ state it returns to ‘jlife’ state and waits for a signal to start over.

4.1.5 Time template

The time template keeps track of the time within the simulation in years. Each year is considered as ten time units. The time template has only two states, which every year will change and calculate the number of live birds. With small alterations to the allbirds () function, the number of juveniles and adult birds can be computed. The initial state has a delay on purpose. This delay is used so that only alive birds will be calculated. We need to keep time separate, because with every template cycle time is set to zero. Bird and extra

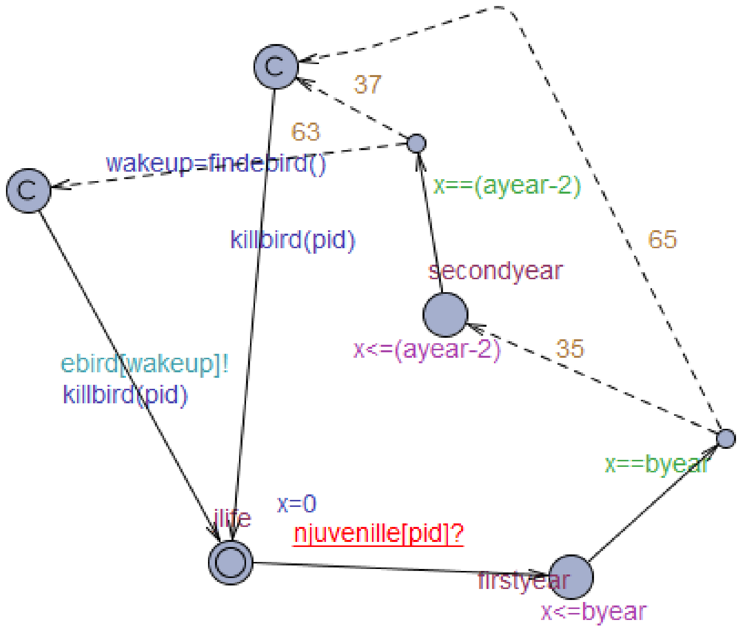

Figure 4.5: Juvenile Template

bird templates are the only templates that don‘t need broadcast channel to continue the cycle (the only occasion on which the cycle stop is death). An adult falcon needs ten time units to finish a cycle. At the 6th time unit a falcon will send a broadcast channel to create an egg (as explained above) and the egg needs four time units to reach ‘bird’ or ‘deadzone’ state. If it reaches ‘bird’ state a broadcast channel will be sent to the ‘juvenile’ template, instructing it to start. As can be observed, juvenile starts at the same time that the bird template finishes the cycle. Juvenile waits for two years (twenty time units) to send the broadcast channel to ‘extra birds’ template (so that inactive extra birds will start). The new falcons will be in the same state ‘find_pair’ as the older falcons. As was mentioned in the description of the Extra Birds template, we only need to have juveniles for two years. So juveniles send the broadcast channel two time units before the ‘correct’ cycle time, exactly before egg template will look for free juveniles. The new falcon will wait in the intermediate state, that was created especially for that reason, two time units and will start the cycle with the other falcons. It is important to mention that the intermediate state between the death state and find_pair state is not in the adult falcon cycle and is enabled only when a falcon is inactive and receives a broadcast channel to start.

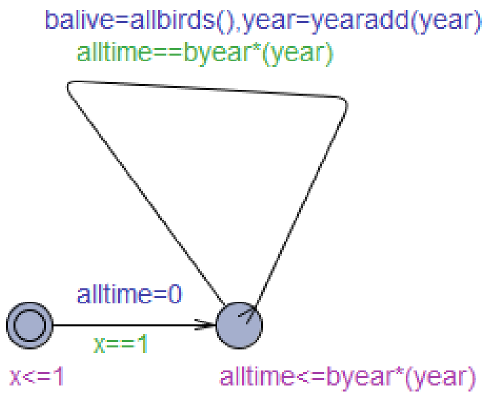

Figure 4.6: Time Template

Chapter 5

Results

Contents

5.1 Results

In this section the results of the simulations will be presented. The results were obtained using Uppaal SMC verifier. The query that was used is E[time.alltime<=x;5000] (max: time.balive). This query returns the result of the time template function allbirds (). With the function allbirds () the number of juveniles or adult falcons or of all falcons can be counted by changing the return value.With E Uppaal SMC determines the expected value bounds that are reached throughout the simulation runs [37]. The time template clock (alltime) was used, which, as described in the previous section, it is the only clock in the whole system that never resets its value and keeps track of time. The value x is the years that we want to simulate multiplied by ten, as each year is considered as ten time units. For example, if we want to simulate five years, x has to be fifty. The number of simulations (runs) for that query is five thousand. A higher number of runs, means lower interval bounds. To be more specific, for the query a confidence interval is given by using the measurements follow Student's t-distribution (approaching Normal distribution when N $\rightarrow \infty$) [36]. The trade off for using a higher number of runs is time. Figures 5.9 and 5.10

nof metric	Adult Falcons	Juveniles	All Falcons
1	119	141	260
2	91	108	199
3	76	89	165
4	61	73	134
5	40	47	87
6	31	36	67

Table 5.1: Initial population of adults and juveniles used on metrics

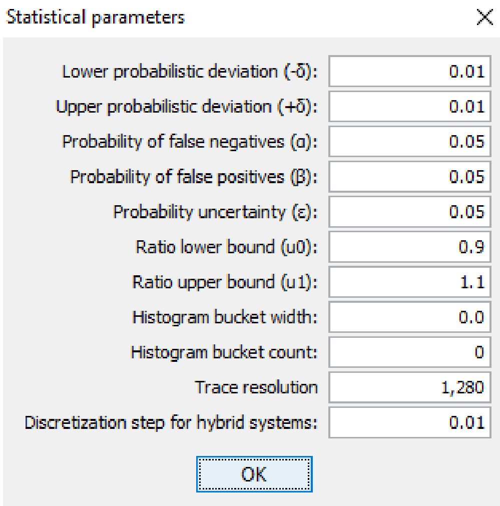

Figure 5.1: Global Statistical Parameters

compare the values for one simulation with those for twenty-five thousand simulations. The global statistical parameters that were used for the simulation are demonstrated in Figure 5.1 and explained in Paragraph 2.7.5.

Another thing that was checked, in order to have a better understanding of whether Uppaal SMC is capable of simulating metapopulation, is the time needed to produce results for different years. The two above metrics were done for an initial population of fifty-three adult birds. Memory consumption is directly related to time and, if we assume that the population of Eleonora's falcon increases over the years, more falcons, juveniles and eggs will be needed. That will result in more memory consumption and in addition, the extra cycle per year increases the time for simulation. It should be noted again that, although the correctness of the results is important, the time required to produce the results is also important.

The way in which the results were obtained is as follows: initially, the desired number of birds was set to simulate using the global constant max bird. At this point the model only had non experienced (three year-old) falcons. However, this did not reflect on the

Eleonora's falcon life cycle, as described in Chapter 3.6 above. In real life, experienced and non-experienced falcons will breed or die every year; juveniles will grow up or die and new juveniles will be born. After the first and second year of simulation only non-experienced or only experienced birds will breed or die or grow up, while no juveniles will grow up or become adults. This problem could be solved with the initialisation of juveniles, but, with this, another problem occurs. The number of juveniles in real life, corresponds to the number of birds in the area. The way this problem was solved, is by setting more birds at the beginning and by not counting them in the results. We run the simulation for two years and the results of the second year give the number of birds that was analysed (initial population on graphs). In the third year, the model will have experienced adult falcons, non-experienced adult falcons and juveniles of different ages. The results after the third year reflect the population dynamics and the third year of the simulation is presented as the first year on the graphs. A detailed table 5.1 with initial population for every simulation is provided. In each graph in Figures 5.2, 5.3, 5.4, 5.6, 5.7 the series in population graphs trends represent the same Eleonora's falcon initial population. For example, green series is the initial population for 76 adult falcons 90 juveniles and 166 falcons in total.

A property that we were keen to examine is the sensitivity of the population to changes in local conditions. Local conditions could affect the probabilities associated with reproduction and, especially, the survival rate of the offspring of a falcon pair. To study this property, the effect of varying reproduction rates, in both exposed and less-exposed nests was analysed. More specifically, we decreased and increased the probability of hatching by 2% and 4% for non-experienced falcons and by 1% and 2% for experienced falcons. Furthermore, the probability for a fledgling surviving was increased and decreased by 2% and 4% for non-experienced falcons and by 1% and 2% for experienced falcons. In table 5.2 the final values for the four cases explained above are presented.

The results of these four cases are illustrated in Figure 5.7. The perception behind those changes was that more experienced pairs settle early during the breeding season, taking the most suitable nesting territories. In such a case, a first time breeder would have to occupy inferior nesting sites. Their nest would be more exposed and threatened by weather or human/rats disturbance.

Case	EH	EF	NEH	NEF
normal	67%	94%	62%	89%
a	66%	93%	60%	87%
b	65%	92%	58%	85%
c	68%	95%	64%	91%
d	69%	96%	66%	93%

Table 5.2: Percentages of successful Experienced Hatching(EH), Experienced Fledgling (EF), Non-experienced Hatching(NEH) and Non-experienced Fledgling(NEF)

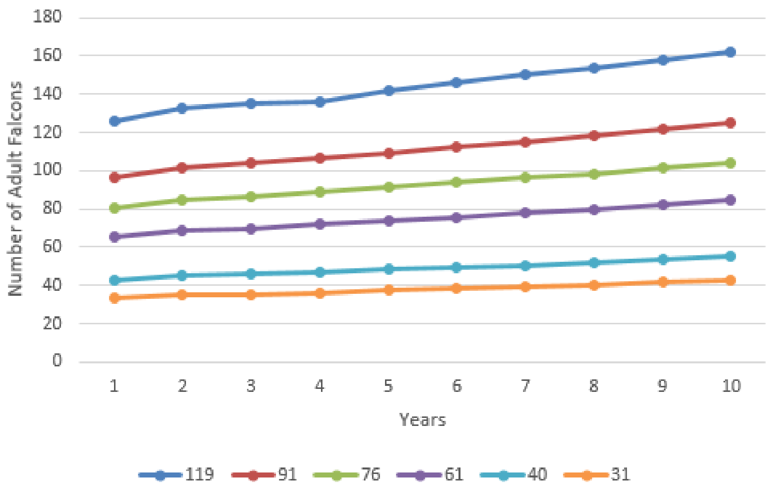

Figure 5.2: Expected number of total adult falcons, for initial population of 119, 91, 76, 61, 40, 31 adult falcons

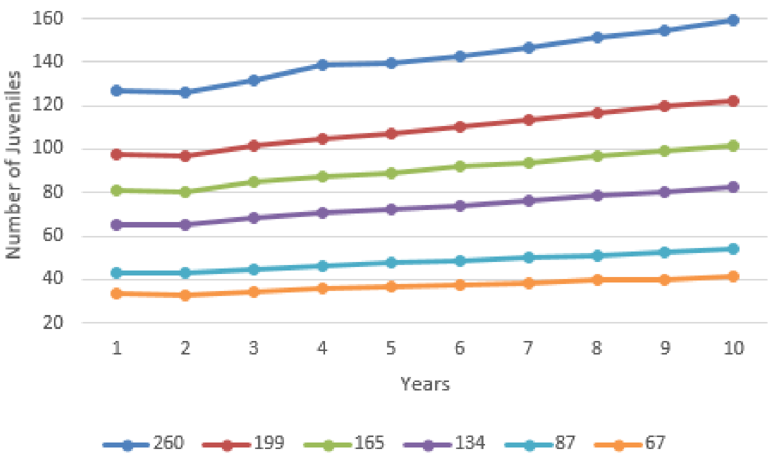

Figure 5.3: Expected number of total juveniles, for initial population of 260, 199, 165, 134, 87, 67 falcons

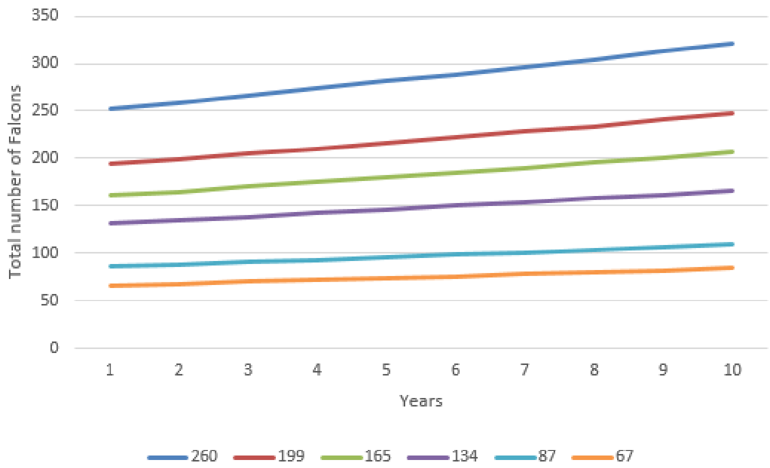

Figure 5.4: Expected number of total Eleonora's falcons, for initial population of 260, 199, 165, 134, 87, 67 falcons

5.2 Result Analysis for Eleonora's falcon

For the present dissertation the future growth rate of Eleonora's population was predicted. In order to achieve that, a population analysis was conducted. Initial populations of thirty one to one hundred nineteen adult falcons were considered for a duration of ten years. If growth has a positive value then the size of a population increases, otherwise it decreases. According to figure 5.6, population growth rate was positive and its value ranged from $2 \leq growthrate \leq 3.5$. To be more specific, the average growth rate value ranged from $2.70 \leq growthrate \leq 2.76$. The first year is not taken into consideration because it is an extreme case and does not represent the real population trend. According to the above, it is safe to conclude that Eleonora's falcon species is not in danger of extinction. Despite the high number of juveniles, only a small number of juveniles mature as mentioned in paragraph 3.6 above. This is the major reason that population growth value is slightly above zero. Furthermore, by observing the range of Eleonora's falcon population growth value for different initial populations, it can be noticed that the value does not change significantly. This observation implies that it is possible to use small subgroups of Eleonora's falcon population to produce accurate predictions.

Another aspect of Eleonora's falcon life-cycle that had to be investigated was the small changes in local conditions that affect the breeding period. The graph in figure 5.7 demonstrates the results taken using the model setting in table 5.2. The following information can be extracted from figure 5.7. The changes producing ‘d’ returned the most falcons and changes producing ‘b’ returned the lowest number of falcons, with 3.5 and 2.0 growth

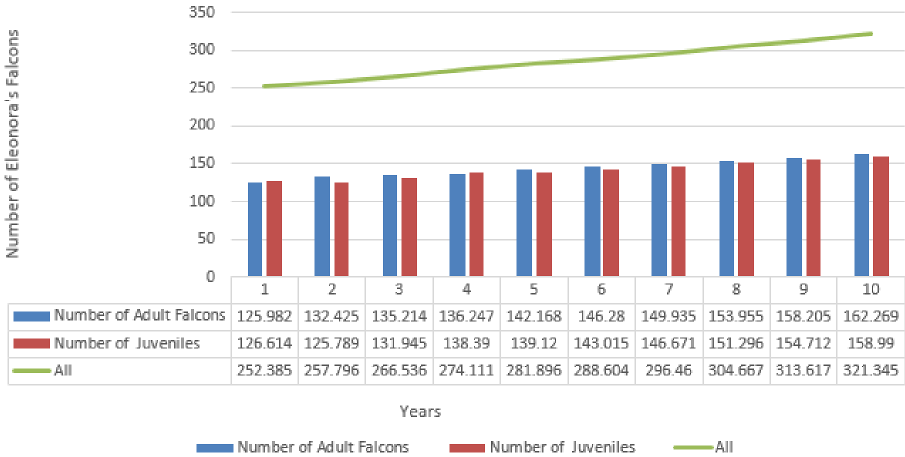

Figure 5.5: Expected number of total Eleonora's falcons, juveniles and pairs for initial population of 119 adult falcons

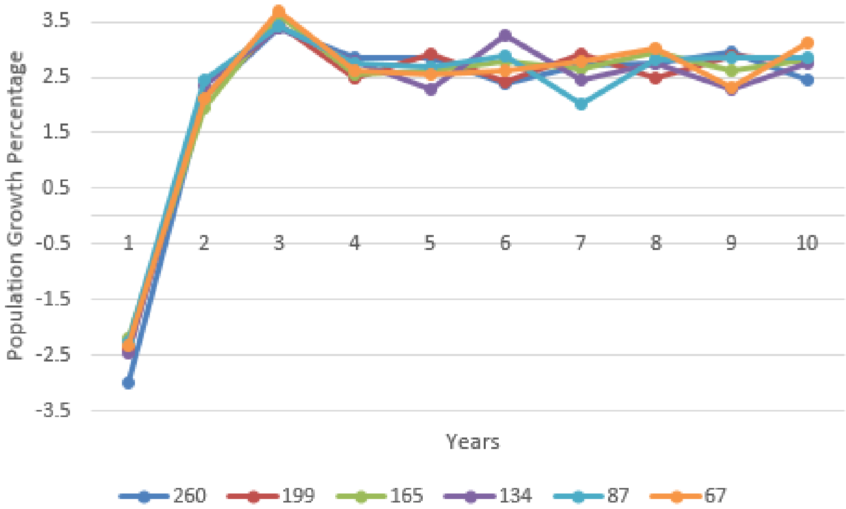

Figure 5.6: Expected population growth percentage for initial population of 260, 199, 165, 134, 87, 67 falcons

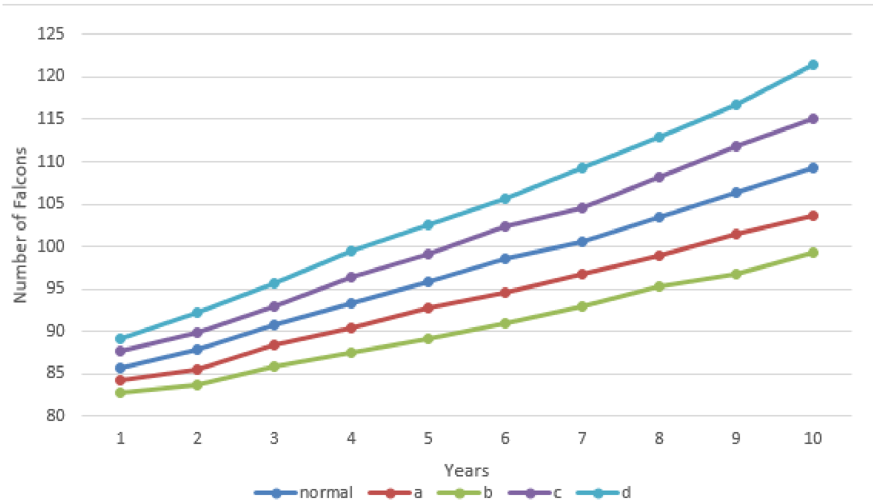

Figure 5.7: Expected population of total Eleonora's falcons with different percentages of successful breeding, for initial population of 87 falcons

rates respectively. The difference between them is twenty two falcons over the span of ten years.

The cases of ‘d’ and ‘b’ have small differences, yet these setting changes were enough to yield a big disparity in the results. A beneficial change to local conditions may come with an increase of population size, but we are more concerned for the negative changes that can happen to the local conditions. The small changes done in the simulation model resulted to a decrease of the growth rate. The change may be small, but with an already small growth rate to begin with, the Eleonora's falcon species seems to be threatened. One can only imagine what may happen if these changes are of larger magnitude.

5.3 Result Analysis for Uppaal Statistical Model-Checker

The metrics below were conducted in order to find out if Uppaal SMC is suitable for modeling ecological systems. The metrics results can be separated in two parts. The first one is about time and the other about expected value confidence bound (confidence interval). We are strict with time, but we can be flexible with confidence interval in ecological systems. It is important to note that static instantiation was used for the templates and the metrics were obtained at the University of Cyprus Student Labs. The System Specifications are the following:

- Operating System: Windows 7 Enterprise

Adult Falcons	Runs	Time(s)
40	2350	283
80	4500	2265
120	6500	7360
160	9200	18524
200	13000	>32000
300	18000	>32000

Table 5.3: A table for expected time of different number of initialized falcons for 6 years,with total confidence bound less than 0.5

- Processor: Intel(R) Core(TM) i5-4590 CPU 3.30 GHz
- Ram: 8.00GB
- System Type: 64-bit Operating System

From figure 5.8 it can be observed that the increment of time among years is low. However, the increment of time among the number of simulations does not allow us run more than 10000 simulations. As mentioned above, we can be flexible with confidence interval, but always according to the size of the population to be analysed. The initial population that was used consisted of only 53 adults, so the confidence bounds can be acceptable only if it is less than 0.5. The need to have more than 2500 thousand simulations can be deduced from figures 5.3 and 5.10. As a result of the above analysis the first restriction on modeling ecological systems appeared. We are limited to a number of simulations with lower bound 2500 and a higher bound 10000 and we have not examined yet the time needed for different number of initialized falcons.

In order to examine the time needed for different number of initialized falcons we checked the time needed to produce results for different initial size of falcons with confidence bound less than 0.5 and less than 1. One can understand that the span of displayed sample that probability distribution function uses to calculate confidence bound increases with the increment of initialized falcons. As a consequence more runs are needed to produce the same confidence bound with less initialized falcons. From tables 5.3 and 5.4 the second restriction is obtained, the number of initial population. For small number of falcons Uppaal SMC is relatively fast and precise, but for initial population larger than one hundred sixty falcons, there will be a compromise with confidence bound value around one. Furthermore, results for more than two hundred fifty falcons cannot be produced within a reasonable amount of time.

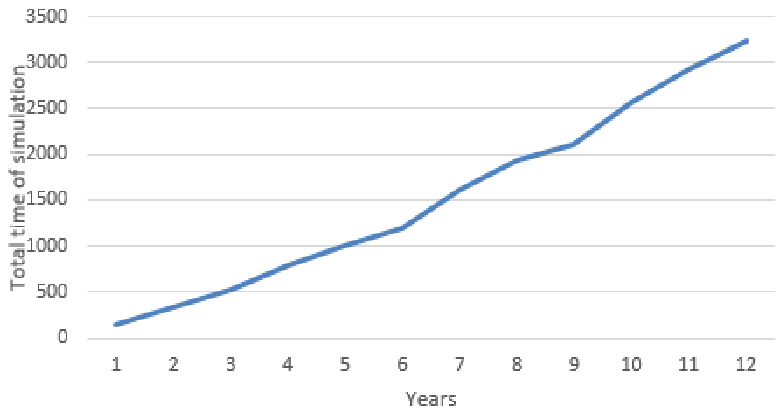

Figure 5.8: Expected total time (seconds) for 1 to 12 years for initial population of 53 adult falcons

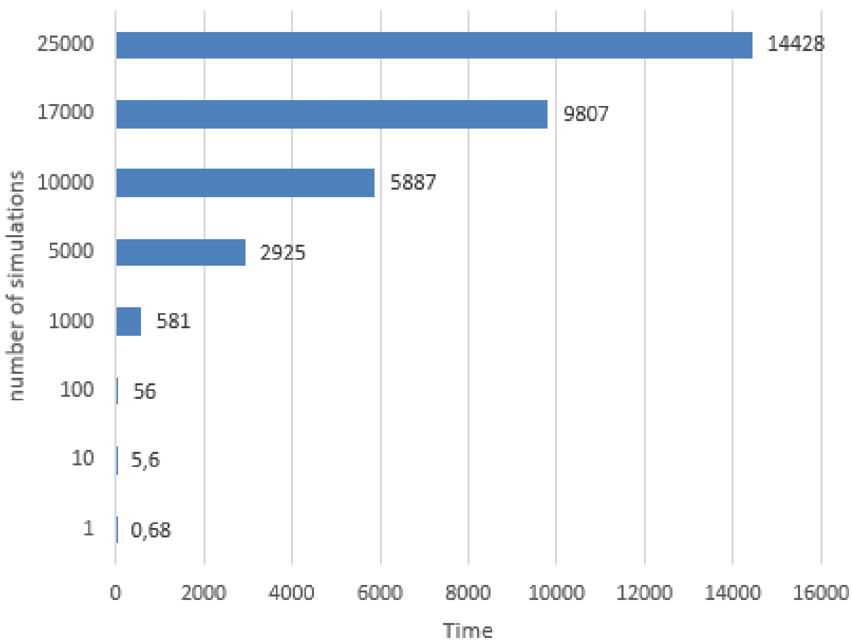

Figure 5.9: Expected total time (seconds) for 1, 10, 100, 1000, 5000, 10000, 17000, 25000 simulations runs for 11 years and initial population of 53 adult falcons.

Adult Falcons	Runs	Time(s)
40	600	74
80	1300	140
120	2000	2300
160	2500	5034
200	3000	9519
300	3500	25736

Table 5.4: A table for expected time of different number of initialized falcons for 6 years, with total confidence bound less than 1

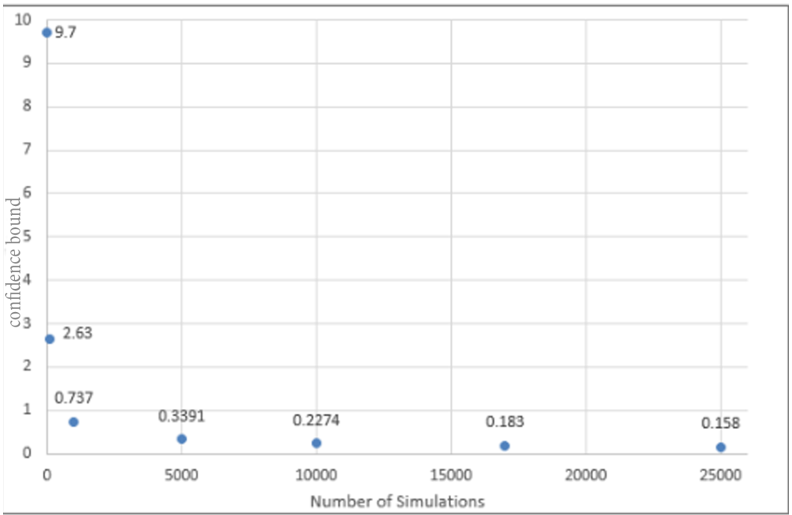

Figure 5.10: Expected confidence bound for 10, 100, 1000, 5000, 10000, 17000, 25000 simulations, for initial population of 53 adult falcons and 11 years

Chapter 6

Conclusion

Contents

6.1 Discussion and concluding remarks

This paper gives an overview of the features of UPPAAL-SMC. The tool has been applied to the case study of Eleonora's falcons. As outlined in this paper, Uppaal-SMC has a large potential for future work and applications. Uppaal-SMC provides powerful visualization capabilities valuable to biologists and an accurate method for statistical model-checking.

Uppaal SMC is undoubtedly a powerful tool to create models of systems in the real world and is user-friendly. Uppaal offers GUI, Graphical Specification and Counter example visualization. The ability to visualize variables and other expressions is unquestionably useful for developing models and showing results. Additionally, Uppaal modeling languages are Timed Automata and C subset language, which are both easy to learn and work with. However, Uppaal SMC needed a large amount of time to produce results for Eleonora's falcon future behaviour. Another disadvantage, is that Uppaal SMC does not have an organized community and documentation on Uppaal-SMC is insufficient. There is, on Yahoo, an online group of six hundred members which answers queries. However, the structure of yahoo answers compared to other websites like stackoverflow, is not helpful. It makes it difficult for developers to find quick solutions to their problems. The

Uppaal SMC tool was released on July 1st, 2014 Uppaal 4.1.19 (version with SMC extension) and there have been no updates since that date. The problem with the compatibility of simulator and dynamic instantiation of templates is still not solved.

It is essential to compare the results from previous thesis regarding Palps and S-Palps process algebra translated to Prism model checker, with the results of the present dissertation. The metrics that were obtained using Palps and S-Palps translated in Prism were for an initial population of twenty-five adult falcons, for eleven years and one thousand number of simulations were run [9]. With Palps process algebra the time needed to simulate was 19500 seconds and with S-Palps fifty seconds. With S-Palps a simplification was done to the model, which may have partially affected the results. Through using Uppaal SMC for the present dissertation, it can be stated that Uppaal SMC appears to be notably faster than the solution proposed with Palps translated in Prism. The solution in Palps was capable to simulate only an initial number of up to seventy falcons and the time needed to simulate was more than the time that Uppaal SMC needed to simulate with a larger number of initial falcons. The Uppaal SMC tool needed 581 seconds to simulate an initial number of fifty-three adult falcons, for eleven years and one thousand simulations. The difference between S-Palps translated in Prism and Uppaal SMC may seem large, but we cannot safely conclude that S-Palps solution in Prism is faster than Uppaal SMC, as we do not have data for the behaviour of S-Palps for an initial population of fifty three adult falcons and how much did the simplification on S-Palps algebra affect the time required for the simulations and the results. However, it should be noted that both, S-Palps (Prism) and Uppaal SMC are capable to model an initial population for more than 200 falcons. Unfortunately, it is not possible to compare the accuracy of their results, as there is not adequate information available regarding the previous thesis.

Referring to the initial question of the present dissertation, whether Uppaal SMC is suitable to model complicated ecological systems, it is entirely up to the biologists to decide. If there is statistical data available for a species, Uppaal SMC can produce accurate results but a large amount of time will be needed to produce the results. Through a comparison of the approach followed by the present thesis, with other approaches that biologists use to predict the future behaviour of species, the biologists will be at a better position to determine if Uppaal SMC is suitable for modeling complicated ecological systems.

According to the results produced for the Greek population of Eleonora's falcons, it can be safely stated that the species is correctly listed as a “Least Concern” from the International Union for Conservation. This dissertation only examined the population trend criterion, and more specifically it examined the fact mentioned in paragraph (3.1) that "the population appears to be increasing, and past declines are not believed to have been sufficiently rapid to approach the thresholds for vulnerable under the population trend criterion" [7]. Through the analysis of the results in paragraph 5.2 above, it was found that the population of Eleonora's falcons does, indeed, appear to be increasing. However, the population appears to grow slowly. It was concluded that the problem was not the breeding success rate, but the mortality rate before sexual maturity. Extra tests were run in order to verify the breeding impact on the population. It was assumed that local conditions do not change easily and we focused on local condition changes that were not caused directly by humans. We ascertained that the results were reasonable and it was determined that Eleonora's falcon is still safe. Nevertheless, Eleonora's falcon is most probably threatened from humans. Deforestation, global warming, tourist resorts near breeding colonies and illegal hunting are all human threats to Eleonora's falcon. Threats posed by humans can increase the danger of extinction of Eleonora's falcons, therefore they have to be protected.

6.2 Future Work

There are two directions in which this dissertation could be expanded. The first one has to do with the population of Eleonora's falcon. It would be interesting to have the results of the dissertation analysed by biologists. In fact, this would actually prove whether this modelling approach is useful to them and in what ways. Such an analysis will show whether there is any deviation, and to what extent, from the real numbers of population today, as the model was based on statistical data from 2004 – 2007. If there is a large difference in the predictions produced compared to the actual population today, changes could be made, with the invaluable aid of the biologists, to the parameters so that they reflect the real situation. If, on the other hand, there appears to be no difference and the model actually predicts the future of a population given specific parameters, it could be used to predict the future of other species as well.

The second direction in which this dissertation could be extended has to do with the tool used, Uppaal-SMC. It would be interesting to translate the static creation of birds to a dynamic one and then reinvestigate whether this tool is suitable for the modelling of ecological systems. Furthermore, if the model appears to be useful and accurate, the issue of time consumption could be further examined in order to find a solution.

Bibliography

[1] Hellenic Ornithological Society. Tracking the migration of eleonora's falcon, 2011.

[2] C. Kassara D. Portolou A. Dimalexis G. Karris C. Barboutis P. Latsoudis S. Bourdakis E. Kakalis S. Sfenthourakis S. M. Xirouchakis, J. Fric. Variation in breeding parameters of eleonora's falcon (falco eleonorae) and factors affecting its reproductive performance. *The Ecological Society of Japan 2011*, 1(27):407–416, 2011.

[3] Michael S. Teitelbaum. Population, 2018.

[4] Mary Robinson. A companion to the philosophy of biology. 23(2):34–34, 2009.

[5] Schacht Robert M. Two models of population growth. *American Anthropologist*, 82(4):782–798, 1980.

[6] BONSALL MICHAEL B. and HASTINGS ALAN. Demographic and environmental stochasticity in predator–prey metapopulation dynamics. *Journal of Animal Ecology*, 73(6):1043–1055, 2004.

[7] IUCN. Falco eleonorae (eleonora's falcon), 2018.

[8] Portolou Danae Latsoudis Panagiotis Karris Giorgos Fric Jacob Georgiakakis Panagiotis Barboutis Christos Bourdakis Stratis Ivovič Milica Dimalexis Anastasios, Xirouchakis Stavros et al. The status of eleonora's falcon (falco eleonorae) in greece. *Journal of Ornithology*, 149(1):23–30, 2008.

[9] Panayiota Papamichael. Comparison of process algebras palps and s-palps for modeling ecological systems through the study of eleonora's falcon life-cycle. Master thesis, Universtity of Cyprus, May 2015.

[10] Kyriakos Zantis. Modeling and alasis of biological population of eleonora's falcon with upaal smc model checker. Undergraduate thesis, Universtity of Cyprus, May 2017.

[11] Gulstrand Larsen Marius Mikučionis Danny Bøgsted Poulsen Axel Legay Zheng Wang Peter Bulychev, Alexandre David. «uppaal-smc: Statistical model checking for priced timed automata», 2009.

[12] Benoit Bardot. Comparison of statistical model checkers», 2013.

[13] Marie-Claude Gaudel. Formal methods and testing: Hypotheses, and correctness approximations. In *International Symposium on Formal Methods*, pages 2–8. Springer, 2005.

[14] Sergio Yovine. Kronos: A verification tool for real-time systems. *International Journal on Software Tools for Technology Transfer*, 1(1-2):123–133, 1997.

[15] Claus Schröter, Stefan Schwoon, and Javier Esparza. The model-checking kit. In *International Conference on Application and Theory of Petri Nets*, pages 463–472. Springer, 2003.

[16] Gerd Behrmann, Alexandre David, and Kim G Larsen. A tutorial on uppaal. In *Formal methods for the design of real-time systems*, pages 200–236. Springer, 2004.

[17] Rodolfo Sabás Gómez and Juan Carlos Augusto. Automatic translation of paradigm models into pltl-based programs. *Proc. SEKE02*, pages 497–503, 2002.

[18] Soren Christensen and Kjeld H Mortensen. Design/cpn ask-ctl manual. *University of Aarhus*, 1996.

[19] Edmund M Clarke, Orna Grumberg, and Doron Peled. *Model checking*. MIT press, 1999.

[20] Edmund M. Clarke, E Emerson, and Joseph Sifakis. Model checking. 52, 11 2009.

[21] Håkan L.S.Younes. Statistical probabilistic model checking with a focus on time-bounded properties. *Information and Computation*, 204:1368–1409, 2006.

[22] Agha G Sen K., Viswanathan M. Statistical model checking of black-box probabilistic systems. *CAV*, 3114(4):202–215, 2004.

[23] Axel Legay, Benoît Delahaye, and Saddek Bensalem. Statistical model checking: An overview. In *International Conference on Runtime Verification*, pages 122–135. Springer, 2010.

[24] Edmund M Clarke. The birth of model checking. In *25 Years of Model Checking*, pages 1–26. Springer, 2008.

[25] Edmund M Clarke. Model checking. In *International Conference on Foundations of Software Technology and Theoretical Computer Science*, pages 54–56. Springer, 1997.

[26] Edmund Clarke, Somesh Jha, Yuan Lu, and Helmut Veith. Tree-like counterexamples in model checking. In *Logic in Computer Science, 2002. Proceedings. 17th Annual IEEE Symposium on*, pages 19–29. IEEE, 2002.

[27] Christel Baier and Joost-Pieter Katoen. *Principles of model checking*. 2008.

[28] Saul Kripke. Semantical considerations of the modal logic. 2007.

[29] Michael Huth and Mark Ryan. *Logic in Computer Science: Modelling and reasoning about systems*. Cambridge university press, 2004.

[30] Anna Philippou. System validation and verification,lecture10-11, 2012.

[31] Rajeev Alur and David L. Dill. A theory of timed automata. *Theoretical Computer Science*, 126(2):183 – 235, 1994.

[32] Patricia Bouyer. An introduction to timed automata. 2005.

[33] Peter BulychevAlexandre DavidKim Guldstrand LarsenAxel LegayMarius MikučionisDanny Bøgsted Poulsen. Model-checking in dense real-time. *Information and Computation*, 104:2–34, 1993.

[34] Wang Yi Kim G. Larse, Paul Pettersson. Uppaal in a nutshell. *International Journal on Software Tools for Technology Transfer*, 1:134–152, 1997.

[35] Wang Yi Kim G Larsen, Paul Pettersson. Uppaal: Status developments. *Journal Computer Aided Verification: 9th International Conference, CAV'97*, pages 456–459, 1997.

[36] Alexandre David Marius Mikuăionis Danny BØgsted Poulsen Kim G. Larse, Axel Legay. Uppaal smc tutorial. *International Journal on Software Tools for Technology Transfer (STTT)*, 17(4):397–415, 2015.

[37] Sibylle Sascha Lehmann, Schupp and M Sc Xintao Ma. Online model checking with uppaal smc. 2016.

[38] ScienceDaily. Migratory route of eleonora's falcon revealed for first time, 2009.

[39] Stratis Bourdakis Dr. Dietrich Ristow. International species action plan for eleonora's falcon falco eleonorae, 1999.

[40] D Ristow and S Bourdakis. International species action plan eleonora's falcon (falco eleonorae). *Birdlife International, Council of Europe, Cambridge*, 1999.

[41] Marion Gschweng, Elisabeth K.V Kalko, Ulrich Querner, Wolfgang Fiedler, and Peter Berthold. All across africa: highly individual migration routes of eleonora's falcon. *Proceedings of the Royal Society of London B: Biological Sciences*, 275(1653):2887–2896, 2008.

[42] Dietrich Ristow. On the insect diet of eleonora's falcon falco eleonorae and its importance for coloniality. In *Raptors worldwide: proceedings of the VI World Conference on Birds of Prey and Owls. WWGBP/MME, Berlin*, pages 705–712, 2004.

[43] Pascual López-López, Rubén Limiñana, and Vicente Urios. Autumn migration of eleonora's falcon falco eleonorae tracked by satellite telemetry, 2009-07.

[44] M Nicholls and R Clarke. *Biology and conservation of small falcons*. University of Kent: University of Kent press, 1991.

[45] GANGOSO L., GRANDE J. M., DUCREST A.-L., FIGUEROLA J., BORTOLOTTI G. R., ANDRÉS J. A., and ROULIN A. Mc1r-dependent, melanin-based colour polymorphism is associated with cell-mediated response in the eleonora's falcon. *Journal of Evolutionary Biology*, 24(9):2055–2063, 2011.

[46] A. Hedenstrom, M. Rosen, S. Akesson, and F. Spina. Flight performance during hunting excursions in eleonora's falcon falco eleonorae. *Journal of Experimental Biology*, 202(15):2029–2039, 1999.

[47] Julianna Photopoulos. There is a bird that eats its neighbours babies, 2017.

[48] Dr Dietrich Ristow. Eleonora's falcon (falco eleonorae), 2009.

[49] Dietrich Ristow, C Wink, and M Wink. Biology of eleonora's falcon (falco eleonorae): 1. individual and social defense behavior. *Raptor Res*, 16(3):65–70, 1982.

[50] Christina Kassara Mauricio Toro, Anna Philippou and Spyros Sfenthourakis. Synchronous parallel composition in a process calculus for ecological models. *Theoretical Aspects of Computing*, pages 424–441, 2014.

[51] Michael Wink, Dietrich Ristow, and Coralie Wink. Biology of eleonora's falcon (falco eleonorae): Variability of clutch size, egg dimensions and egg coloring. *Raptor Research*, 19:8–14, 1985.

[52] Yehezkel S. Resheff Sondra Feldman Turjeman Damaris Zurell Nir Sapir Ute Eggers Andrea Flack Wolfgang Fiedler Florian Jeltsch Martin Wikelski Ran Nathan

Christiaan Both Shay Rotics, Michael Kaatz. The challenges of the first migration: movement and behaviour of juvenile vs. adult white storks with insights regarding juvenile mortality. *Journal of Animal Ecology*, 85(4):938–947, 2016.

[53] D Ristow, W Scharlau, and M Wink. Population structure and mortality of eleonora's falcon falco eleonorae. *Age*, 8:6, 1989.

[54] Dietrich Ristow and Michael Wink. Breeding success and conservation management of eleonora's falcon. *ICBP Technical Publication*, 5:147–152, 1985.

[55] Michael Wink and Dietrich Ristow. Biology and molecular genetics of eleonora's falcon falco eleonorae, a colonial raptor of mediterranean islands. *Raptors at Risk. World Working Group on Birds of Prey, Hancock House*, pages 653–668, 2000.

[56] Sama Zefania. Observation of sooty and eleonora's falcons in madagascar. In *Wings over Africa, Proc. Int. Seminar on Bird Migration: Research, Conservation, Education and Flight Safety*, pages 151–159, 2001.