RESEARCH ARTICLE

Model-checking ecological state-transition graphs

Colin Thomas[1,2]*, **Maximilien Cosme**[2], **Cédric Gaucherel**[2], **Franck Pommereau**[1]*

1 IBISC, Univ. Évry, Univ. Paris-Saclay, 91020 Évry-Courcouronne, France, **2** AMAP, Univ. Montpellier, INRAE, CIRAD, CNRS, IRD, Montpellier, France

* colin.thomas@univ-evry.fr (CT); franck.pommereau@univ-evry.fr (FP)

OPEN ACCESS

Citation: Thomas C, Cosme M, Gaucherel C, Pommereau F (2022) Model-checking ecological state-transition graphs. PLoS Comput Biol 18(6): e1009657. https://doi.org/10.1371/journal.pcbi.1009657

Editor: Samuel V. Scarpino, The Rockefeller Foundation, UNITED STATES

Received: November 18, 2021

Accepted: May 8, 2022

Published: June 6, 2022

Peer Review History: PLOS recognizes the benefits of transparency in the peer review process; therefore, we enable the publication of all of the content of peer review and author responses alongside final, published articles. The editorial history of this article is available here: https://doi.org/10.1371/journal.pcbi.1009657

Data Availability Statement: All relevant data are within the manuscript and its Supporting information files.

Funding: CG was partly funded by the Eranet LEAP-Agri project SESASA (n˚ 01DG18020) https://

Abstract

Model-checking is a methodology developed in computer science to automatically assess the dynamics of discrete systems, by checking if a system modelled as a state-transition graph satisfies a dynamical property written as a temporal logic formula. The dynamics of ecosystems have been drawn as state-transition graphs for more than a century, ranging from state-and-transition models to assembly graphs. Model-checking can provide insights into both empirical data and theoretical models, as long as they sum up into state-transition graphs. While model-checking proved to be a valuable tool in systems biology, it remains largely underused in ecology apart from precursory applications.

This article proposes to address this situation, through an inventory of existing ecological STGs and an accessible presentation of the model-checking methodology. This overview is illustrated by the application of model-checking to assess the dynamics of a vegetation pathways model. We select management scenarios by model-checking Computation Tree Logic formulas representing management goals and built from a proposed catalogue of patterns. In discussion, we sketch bridges between existing studies in ecology and available model-checking frameworks. In addition to the automated analysis of ecological state-transition graphs, we believe that defining ecological concepts with temporal logics could help clarify and compare them.

Author summary

Ecologists have drawn state-transition graphs representing the dynamics of ecosystems for more than a century. Model-checking is an automated method for the analysis of such graphs developed in computer science and acknowledged by a Turing award in 2007. Ecologists appear to be mostly unaware of model-checking apart from precursory applications, despite its successes in systems biology.

We propose to address this situation, through an inventory of existing ecological state-transition graphs and an accessible presentation of the model-checking methodology. We exemplify the insights provided by model-checking by applying it to a vegetation pathways model in order to assess management policies aiming to tackle savanna encroachment. We provide a catalogue of patterns to help ecologists with the difficulty of formally

leap-agri.com/?page_id=293. The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

Competing interests: The authors have declared that no competing interests exist.

expressing dynamical properties. Lastly, we sketch bridges between existing studies in ecology and available model-checking frameworks.

Model-checking can be applied to both empirical data and theoretical models, as long as they sum up into state-transition graphs. It provides automated and accurate answers to complex questions that could barely be analysed through human examination, if not impossible to answer this way. In addition, we believe that formally defining ecological concepts within the model-checking framework could help clarify and compare them.

This is a *PLOS Computational Biology* Methods paper.

Introduction

A *state-transition graph (STG)* describes the behaviour of a dynamical system, for example an ecosystem, as a graph whose nodes are the discrete states of the system and whose edges represent the transitions between those states. Ecologists have drawn STGs for more than a century, one of the earliest and best-known examples being the vegetation successions described by Clements [1]. Yet, ecology and environmental sciences appear to remain largely unaware of the *model-checking* methodology [2] developed in computer science to investigate the dynamics of a system represented as an STG. This paper aims to promote the model-checking of ecological STGs.

In ecology, STGs are typically used to represent *community pathways*, i.e. changes in the set of species or populations of an ecosystem through time. For example the successions of plant communities in boreal forests [3], or the assembly of protist communities in laboratory experiments [4]. Such STGs are mostly drawn from observations, hence their relatively small size (a few dozens of states at most). Most of the time, STGs are perceived as graphical representations of the knowledge about the dynamics of the studied system rather than as actual data.

For example, STGs have been used since the '90s as a tool for rangeland management and ecosystem conservation under the concept of *state-and-transition models (STMs)* [5, 6]. Theoretical studies also emphasise the relevance of STGs to investigate community assembly [7, 8]. Both research fields recently mentioned an interest in tools providing dynamical analysis of STGs [8, 9].

In computer science, STGs model the executions of automated systems. Computer scientists design automated tools called *model-checkers* to ensure the absence of bugs during software executions [2]. Model-checkers verify whether the pathways within an STG satisfy a given property, for example that a desired state remains reachable or that harmful behaviour is always avoided. Given a system description that can be computed into an STG and a dynamical property written as a *temporal logic formula*, a model-checker outputs whether the STG satisfies the property or not. Model-checking is an active field of research acknowledged by a Turing award in 2007, encompassing numerous concepts and resulting in a wide variety of implemented tools [10, 11].

In systems biology, STGs are outputted by models of reaction networks or regulatory networks [12]. Model-checking is extensively used to analyse those models [13, 14], proving its suitability for the study of biological systems. For example, model-checking helped validate models of nutritional stress response of *Escherichia coli* [15], T-helper cell reprogramming

[16], mammalian cell cycle [17] or BRAF inhibition pathways in two different cancers [18]. Yet, model-checking methodology appears to remain unknown to most ecologists apart from precursory applications [19, 20]. Ecology encompasses a wide variety of STGs but their analysis is often restricted to visual examination.

This article proposes to address this situation by helping the ecologists handling STGs to get acquainted with the model-checking methodology. First, it provides an inventory of existing ecological STGs and a didactic presentation of model-checking. This overview is illustrated by the application of model-checking to a model of the Borana vegetation pathways based on STM literature [21–24]. The model-checking methodology can be implemented in a wide variety of ways, bridges between existing studies in ecology and available model-checking frameworks are sketched in Discussion.

1 Materials and methods

1.1 State-transition graphs (STGs)

A *state-transition graph (STG)* [25] represents the dynamics of a system as a *graph* $G = (S, E)$, where S is a set of *nodes* (the discrete states of the system) and $E \subseteq S \times S$ is a set of *directed edges* (the transitions enabling to move from one state to another). The fact that $(s, s') \in E$ is often noted $s \rightarrow s'$.

The transitions may be labelled by their driving event or process (taken from a set of labels L), resulting in $E \subseteq S \times L \times S$, the fact that $(s, l, s') \in E$ is then noted $s \xrightarrow{l} s'$. An STG is said *deterministic* if every state has at most one outgoing transition, and *non-deterministic* otherwise. Every state of a deterministic STG is the start of a single pathway, thus the behaviour of the system is also deterministic.

In ecology, the state $s \in S$ of an ecosystem is often discretely abstracted by its *community* (i.e. restricted to its set of species or populations). Subsequently, STGs are found in a broad variety of studies focusing on the dynamics of ecological communities, historically called community succession for plants and community assembly for animals [26, 27].

Graphs are widespread in ecology, but STGs must be discriminated from *interaction networks* such as the iconic trophic networks [28, 29]. Indeed, the former grasp the temporal behaviour of an ecological system, while the latter grasp the processes taking place between its components. A node (resp. an edge) of an STG is a temporal stage (resp. an event, i.e. a state transition) of the system dynamics. Whereas a node (resp. an edge) of an interaction network is a component (resp. a flux) of the system. The model-checking methodology presented in this paper deals with the temporal changes of a discrete-event system and thus is designed to analyse STGs specifically.

1.1.1 State-transition graphs in ecology. The dynamics of ecological systems have been described as states and transitions for more than a century. For example, Clements [1] used STGs to represent ecological successions [30] between vegetation states called “seral stages”. Since then, STGs have regularly been used under various names in ecology, from “behaviour graphs” [31], to “kinematic graphs” [32], or under the generic term “pattern” [33].

More recently, STGs form the core of one of the most commonly used frameworks for ecological successions: the *state-and-transition models (STMs)*. Note that despite their orthographic proximity, “STG” and “STM” shall not be mixed up. While STGs refer to a general mathematical concept, STMs are special instances of STGs designed for particular purposes. Indeed STMs are derived from observations and are designed to cope with the non-deterministic and irreversible nature of observed dynamics [5]. STMs are also intended to be user-friendly, enabling participatory model development and collaborative management [6]. The main goal of STMs is to assist managers and scientists in collectively proposing policies driving

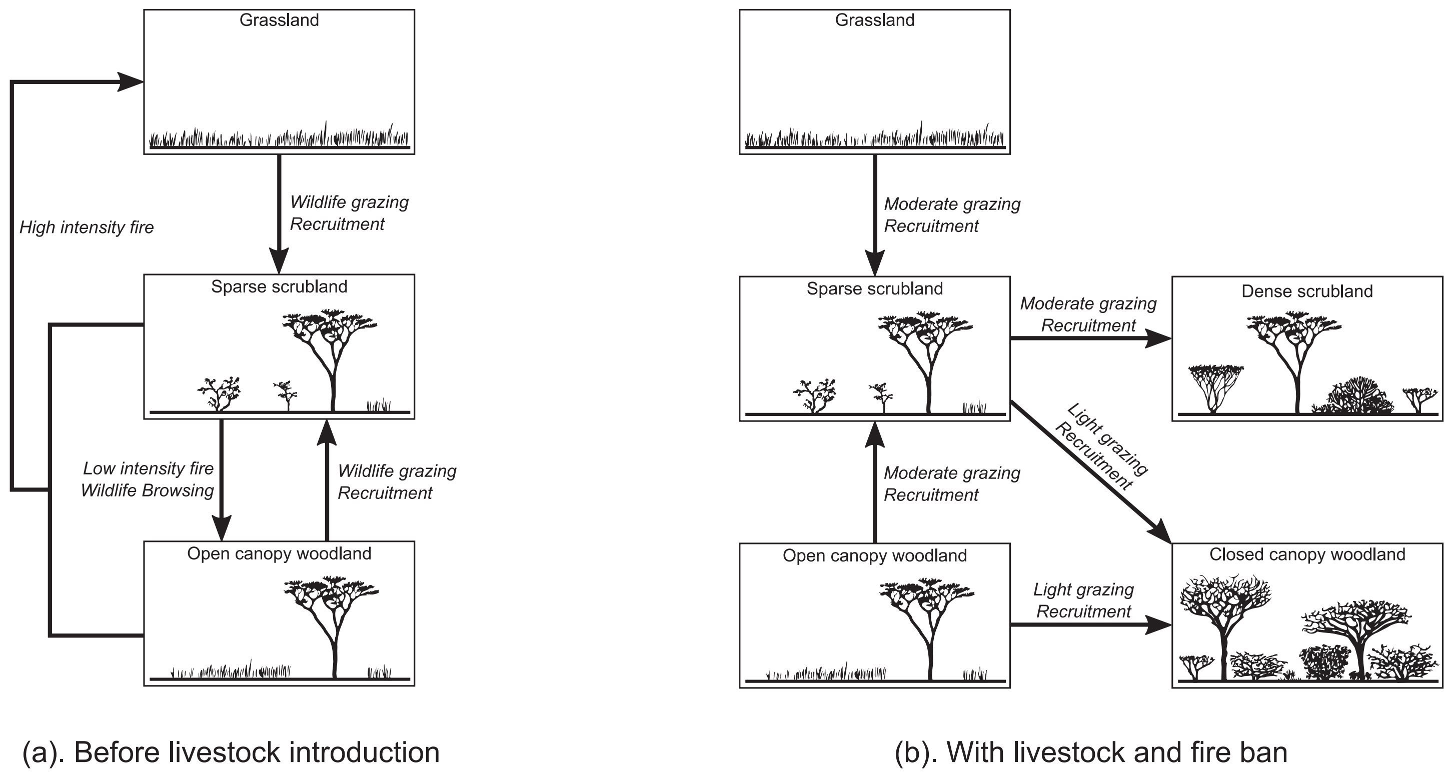


Fig 1. State-and-transition models of the Borana vegetation pathways. The states are embodied by illustrated boxes, and the transitions by arrows labelled by their main driving processes. (a) Before pastoralism, fire was the main driver of the rangeland dynamics. The combination of fire, wildlife herbivory and vegetation recruitment maintained the entire system in a loop between open canopy woodland and grassland. (b) The presence of cattle and the fire ban gave a competitive advantage to woody plants, inducing an irreversible bush encroachment. Concurrently, wildlife increasingly avoided the Borana zone because of the denser human and livestock populations. (Based on [21] with author's permission).

https://doi.org/10.1371/journal.pcbi.1009657.g001

the ecosystem through some desired pathways while avoiding others. In order to remain user-friendly, STMs sizes usually do not exceed a few dozens of states. While STMs originally stem from rangeland management [5], they are now used in many fields such as natural park management [34] (see for example the "EDIT" database housing a large catalogue of STMs [35]), geomorphology [36], or agroecology [37].

In addition, STGs are found in the field of community assembly under the concept of *assembly graphs*. In such graphs, every node is a stable species community and every edge is an invasion event. Contrarily to STMs, most studies involving assembly graphs are theoretical [7, 8], yet a few are experimental [4].

Lastly, STGs are the output of a wide diversity of modelling formalisms in ecology and environmental sciences [8, 19, 38, 39]. Recently, some studies have used Boolean models such as *Boolean networks* (i.e. systems of logical equations) to model ecological dynamics, from plant-pollinator associations [40] to spruce budworm outbreaks [41].

1.1.2 State-and-transition models of the Borana vegetation pathways. The STMs developed by Liao and Clark [21–24] describe the vegetation pathways of the Borana Zone in southern Ethiopia (Fig 1). Open canopy woodland (a savanna-like vegetation class encompassing a grass layer with sparse trees) was formerly the most prominent vegetation class in Borana [22]. But since the fire ban in the 1970s, the region has been undergoing a rapid increase in the density of woody plants (known as *bush encroachment*). As local people predominantly practice pastoralism, the reduction in herbaceous cover threatens their livelihood. Hence understanding the vegetation pathways is critical to help pastoralists and policymakers mitigate bush encroachment [23].

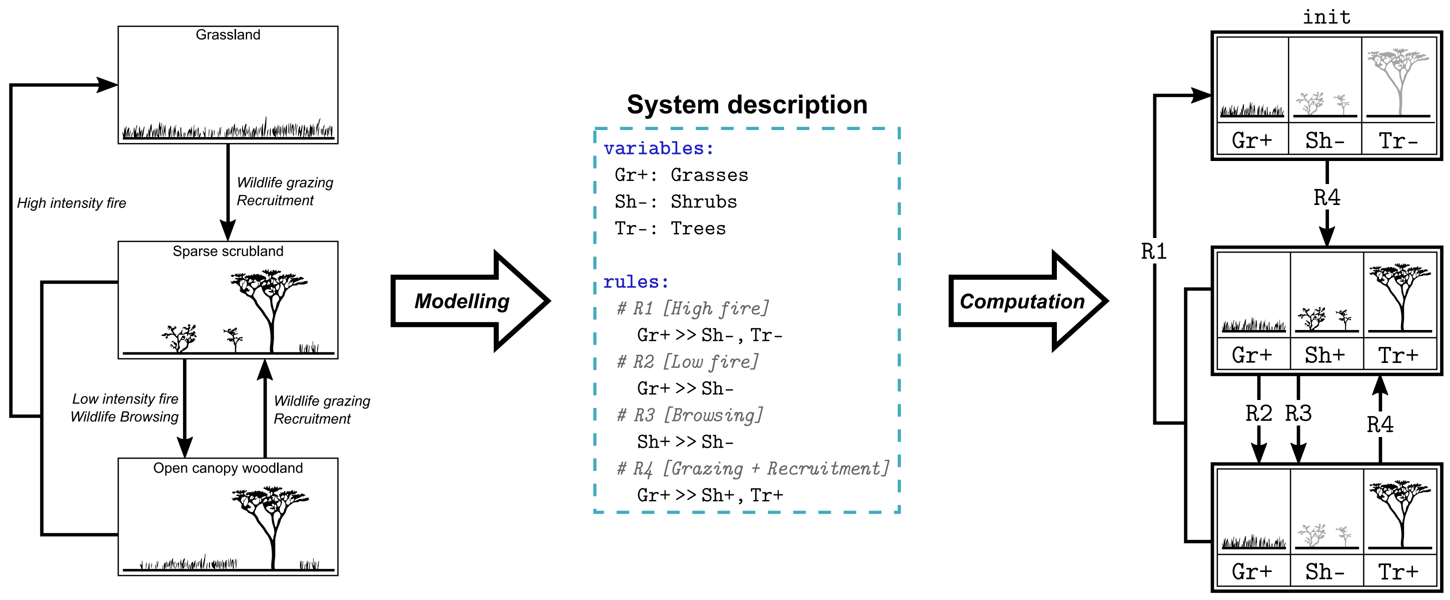
The states of the STMs represent vegetation classes (Fig 1), see [22] for their exhaustive definitions. The transitions are labelled by their main drivers, as is often the case in the STM framework. As these STMs consist of nodes and edges representing the vegetation dynamics, they indeed form STGs. The STGs of Fig 1 are non-deterministic because some states have more than one outgoing transition. Moreover, the STG representing bush encroachment (Fig 1b) is called *irreversible* because some pathways are one-way only: for example grasslands cannot be reached from any encroached state (dense scrubland or closed canopy woodland).

1.1.3 Building a model of Borana vegetation pathways with if-then rules. Most STGs found in ecology are directly drawn from observations (for example the two STMs of Fig 1). We propose to illustrate the model-checking methodology on an STG generated from a model. Based on the literature [21–24], we built a description of the Borana vegetation pathways, called "*Borana model*" in the following for concision, from which an STG can be computed. Our goal is twofold: first to show that a complex STG can be computed from a compact system description, enabling the construction of models not only based on past observations but also forecasting novel behaviours [9]; second to illustrate the scalability of the model-checking toolbox. Indeed while each STM of Fig 1 represents an observed scenario, the *Borana model* embraces the same historical scenarios as well as recommended management scenarios in order to foresee their cascading effects.

Each state of the *Borana model* consists of a vector of Boolean variables representing the functional presence (noted +) or absence (−) of the components of the system. A variable is considered functionally present if its presence has an observable influence on the system, and functionally absent otherwise. Variables influencing the system without being influenced in turn are called *controls*, for example climatic conditions or management policies. Controls remain constant along the dynamics, thus two states with distinct control values are out of reach from one another.

The transitions of the *Borana model* are generated from the execution of a rule-based formalism. More precisely, we use *if-then rules* (if the *condition* is fulfilled, then the *consequence* may arise), a methodology previously proposed [42, 43] and implemented [44] to model expert knowledge about ecosystem dynamics. Every if-then rule $\mathtt{R}$ whose condition is fulfilled in a given state $s \in S$ and whose consequence is not yet fulfilled in s, generates an outgoing transition $s \xrightarrow{\mathtt{R}} s'$ toward the state $s' \neq s \in S$ resulting from the application of the consequence of $\mathtt{R}$ to s. Thus loops from one state to itself are excluded. Starting from a set of initial states, the full set of reachable states is computed by the cascading applications of rules. This modelling approach is exemplified by a toy model replicating the STM without encroachment (Fig 1a) and involving only three variables and four rules (Fig 2). This toy model aims for developing a fine intuition of the if-then rule modelling, note that a formal definition of if-then rule modelling is given in S1 Appendix.

The complete *Borana model* consists of 15 variables, including seven controls (Table 1), and 19 rules (S1 Table). Justifications of the modelling choices assumed by the *Borana model* are given in S2 Appendix. Each valuation of the variables describes a state of the Borana ecosystem, that can be classified into vegetation classes [22] (see S2 Table). Each valuation of the controls defines a specific *scenario* for the Borana dynamics (i.e. a combination of altitude and management policies), inspired from historical management and recommendations to limit encroachment [23]. The control variables never change in consequence of the rules (S1 Table), hence they influence the system without being influenced by it. The *Borana model* has $2^7 = 128$ initial states, one for each scenario (i.e. one for each valuation of the 7 control variables),



```
variables:
 Gr+: Grasses
 Sh-: Shrubs
 Tr-: Trees

rules:
 # R1 [High fire]
   Gr+ >> Sh-, Tr-
 # R2 [Low fire]
   Gr+ >> Sh-
 # R3 [Browsing]
   Sh+ >> Sh-
 # R4 [Grazing + Recruitment]
   Gr+ >> Sh+, Tr+
```

Fig 2. Toy model illustrating the if-then rule modelling. Modelling of the STM of Fig 1a (left) into a *if-then* rule model (middle: system description) from which an STG can be computed (right). Each state of the STG is a vector of three Boolean variables defined in the "`variables`" section of the system description: grasses (`Gr`), shrubs (`Sh`) and trees (`Tr`), noted with + if present and with − if absent. The initial values of the variables are noted next to their symbol in the system description, defining the initial state from which the STG is computed (`Gr+, Sh-, Tr-`). The "`rules`" section of the system description defines the if-then rules describing the transitions. For example, the first rule `R1` embodies that *if* grasses are present (`Gr+`) *then* (`>>`) they can fuel a high intensity fire burning down shrubs and trees (`Sh-, Tr-`), as grasses resprout first they do not disappear in the fire consequence. This rule corresponds in the STG to the transitions labelled by `R1` from the middle and bottom states toward the top state. The cascading applications of every rule whose condition is fulfilled and whose consequence is not to every reachable state compute the STG. Compared to the STM, the computed STG is more explicit: there are two transitions from the middle state toward the bottom one because two distinct events may lead the system from the former to the latter.

https://doi.org/10.1371/journal.pcbi.1009657.g002

corresponding to the grassland vegetation class [21] (only grasses are present, see Table 1 and S2 Appendix). A subgraph is generated from each initial state by the cascading applications of the if-then rules. Those subgraphs are disconnected (no rule change the controls) and form together the full STG computed from the *Borana model*. Note that the toy model of Fig 2 does not have any control, thus it consists of only one connected STG.

1.2 Model-checking

Model-checking is an automated method for the analysis of any dynamical system that can be modelled by states and transitions [45]. Its goal is to check that a given automated system (hardware or software), modelled as an STG, satisfies a given dynamical property, usually

Table 1. Variables and controls of the *Borana model*. The initial values of the variables are noted next to their symbols.

Variable	Description	Control	Description
`Gr+`	Grasses	`Alt`	Altitude
`Sh-`	Shrubs	`Fb`	Fire ban
`Tr-`	Trees	`Cb`	Crop ban
`Sa-`	Tree saplings	`Wl`	Wildlife presence
`Cr-`	Crops	`Ps`	Pastoralism
`Lv-`	Livestock	`Ig`	Intensive grazing
`Gz-`	Wild grazers	`BLv`	Browsing Livestock
`Bw-`	Wild browsers		

https://doi.org/10.1371/journal.pcbi.1009657.t001

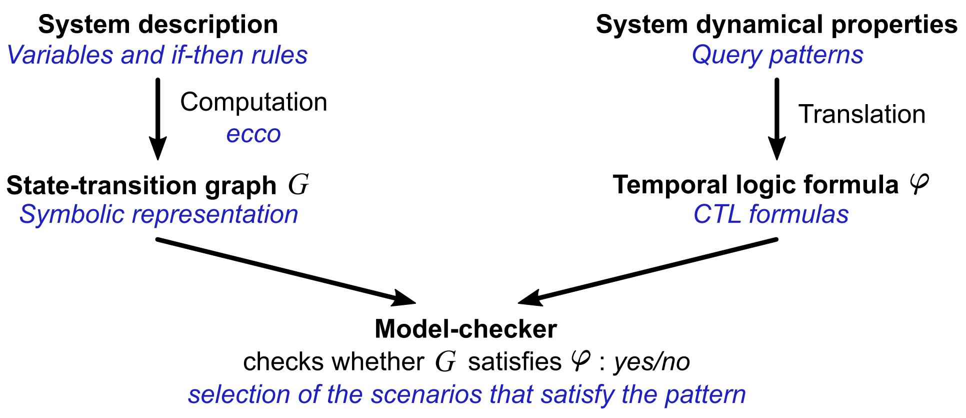

System description
Variables and if-then rules
Computation
ecco
State-transition graph G
Symbolic representation
System dynamical properties
Query patterns
Translation
Temporal logic formula φ
CTL formulas
Model-checker
checks whether G satisfies φ : yes/no
selection of the scenarios that satisfy the pattern

Fig 3. The model-checking methodology. In black the general model-checking outline. In blue italic the implementation described in this article that consists of a particular choice of techniques and tools among the available ones. (Adapted from [45]).

https://doi.org/10.1371/journal.pcbi.1009657.g003

written as a *temporal logic formula* (Fig 3, in black). A *model-checker* is a software performing this operation, returning a *yes/no* output depending on whether the STG satisfies the property or not, generally with a counterexample pathway for negative output. In the field of computer science dealing with model-checking, STGs are mainly named *Kripke structures* or *labelled transition systems*, depending if either or both of their states and transitions are labelled. As in systems biology [17, 25], we will keep calling them state-transition graphs (STGs) for clarity. While model-checking is a wide and active field of research [2], the scope of this paper is limited to exhibiting the potential of model-checking in ecological applications. Only one implementation of the model-checking methodology will be detailed in this paper (Fig 3, blue italic annotations), while the diversity of the relevant implementations will be sketched in the Discussion.

To our knowledge, model-checking has been very scarcely used in ecology, despite its extensive application in systems biology [13]. So far, most formal analyses of STGs in ecology have been limited to graph measures [46] and topology analyses [40, 41, 44]. We identified only a few precursory applications of model-checking in ecology [19, 20]. Those studies introduce a specific implementation of the model-checking methodology based on timed automata to model the dynamics of ecosystems, such as coral reef fisheries.

Besides a modelling language that enables a description of the system that can be computed into a STG (for example the if-then rules presented above), model-checking also requires a formal language to express the dynamical properties to be checked, such as temporal logics (Fig 3).

1.2.1 Expressing properties using Computation Tree Logic (CTL). *Computation Tree Logic (CTL)* is one of the most popular temporal logics [2] because it is particularly fitted to express properties of branching dynamics with alternative pathways. We chose to present CTL in our implementation because ecological STGs often involves alternative pathways. A CTL formula describes a property over *computation trees*, noted *CTs* as in the beginning of CTL. A CT is rooted at a given state of the STG, and its branches are the alternative pathways starting from this state (Fig 4). In computer science, an STG represents the behaviours of a software system, thus every branch of a CT represents an alternative software computation, hence the name "computation tree". Here are two examples of CTL dynamical properties to foster intuition: (1) all the CT's pathways eventually lead to an encroached state, (2) at least one of the CT's pathways maintains grasses. A CTL model-checker checks whether the CT rooted in each

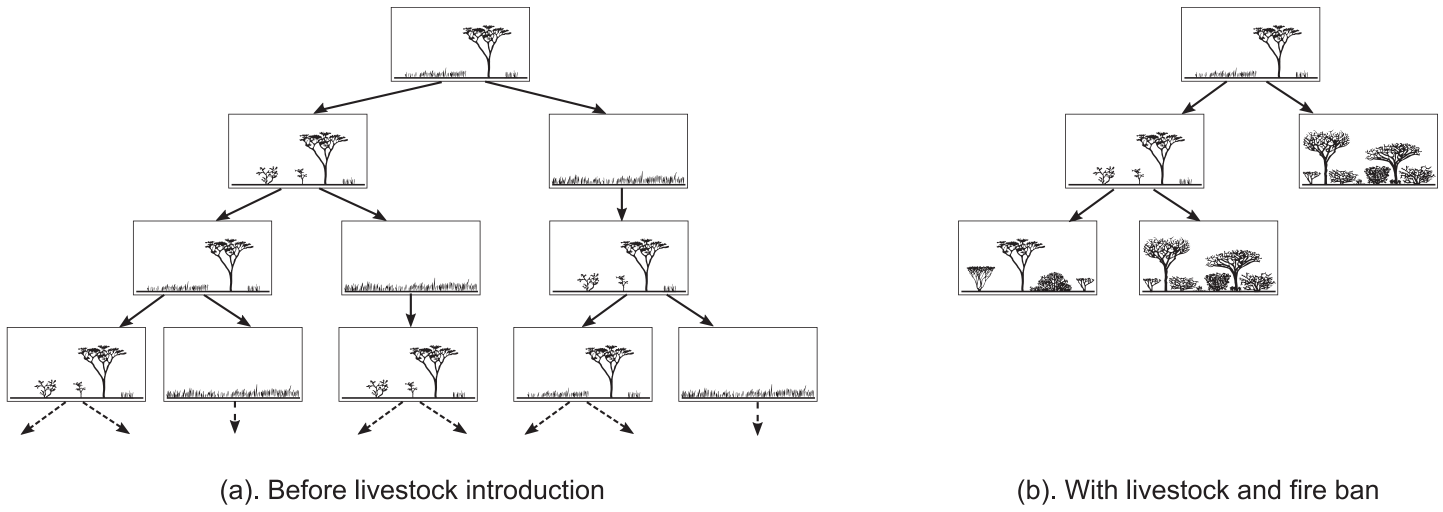

(a). Before livestock introduction
(b). With livestock and fire ban

Fig 4. Computation trees rooted in the STMs of Fig 1. Each branch descending from the root represents a possible pathway in the corresponding STM. (a) The CT rooted in the open canopy woodland state of the STM of Fig 1a. As the pathways are infinite in this STM, the branches of the CT are also infinite, and thus the CT itself. (b) The CT rooted in the open canopy woodland state of the STM of Fig 1b. The grassland state is not reachable from the open canopy woodland state in this STM, and thus it does not appear in its CT. As the pathways are finite in this STM, the branches of the CT are also finite, and thus the CT itself. Formally the branches of a CT are usually assumed to be infinite, so that its pathways always carry on. In order to tackle this issue, the dead-end leaves of a CT can be interpreted as infinite pathways remaining in the same state.

https://doi.org/10.1371/journal.pcbi.1009657.g004

state satisfies a CTL formula or not. Thus a CTL model-checker discriminates between the states whose CT satisfies a given property and those whose CT does not.

The syntax and semantics of CTL are given in Fig 5. A *state property p* is a Boolean property mapping over states. For example, the presence of shrubs is a state property noted `Sh+`, and in Fig 2 it is only True (noted by $\top$) over sparse scrubland (middle state). More complex state descriptions are built by combining state properties using the *Boolean logical operators*: not ($\neg$), and ($\wedge$), or ($\vee$). For example, encroachment could be defined as the absence of grasses, and the presence of shrubs or trees: `Gr-` $\wedge$ (`Sh+` $\vee$ `Tr+`). Other Boolean logical operators can be built on top of the three ones above, such as the implication ($\Rightarrow$) that is defined such that $p \Rightarrow q$ is equivalent to $(\neg p) \vee q$.

The *temporal operators* of CTL are always the combination of two types of operators: first a *quantifier* ($\exists$ or $\forall$) dealing with branching by quantifying over the pathways starting from a given state, second a *modality* (F, G, or U) specifying the order of properties along a pathway. Temporal operators can thus be separated between existential and universal operators. *Existential operators* ($\exists F$, $\exists G$, or $\exists U$, see Fig 5) specify that their modality has to be verified by *at least one branch* of the CT (thus by at least one pathway of the STG starting from its root state). *Universal operators* ($\forall F$, $\forall G$, or $\forall U$, see Fig 5) specify that their modality has to be verified by *every branch* of the CT (thus by every pathway of the STG starting from its root state). Modality F specifies that the property *finally* becomes true at one step of the pathway. Modality G specifies that the property is *globally* true all along the pathway. Modality U specifies that the left-hand-side property remains true along the pathway *until* the right-hand-side property finally becomes true. Modality *next* X has been omitted from this paper to simplify the presentation.

For example, in the CTs rooted in the Borana STMs (Fig 4):

- the CTL formula $\exists F$ `Tr-` specifies that a state without trees (the vegetation ones) is reachable from the root of the CT, which is satisfied in Fig 4a but not in Fig 4b;

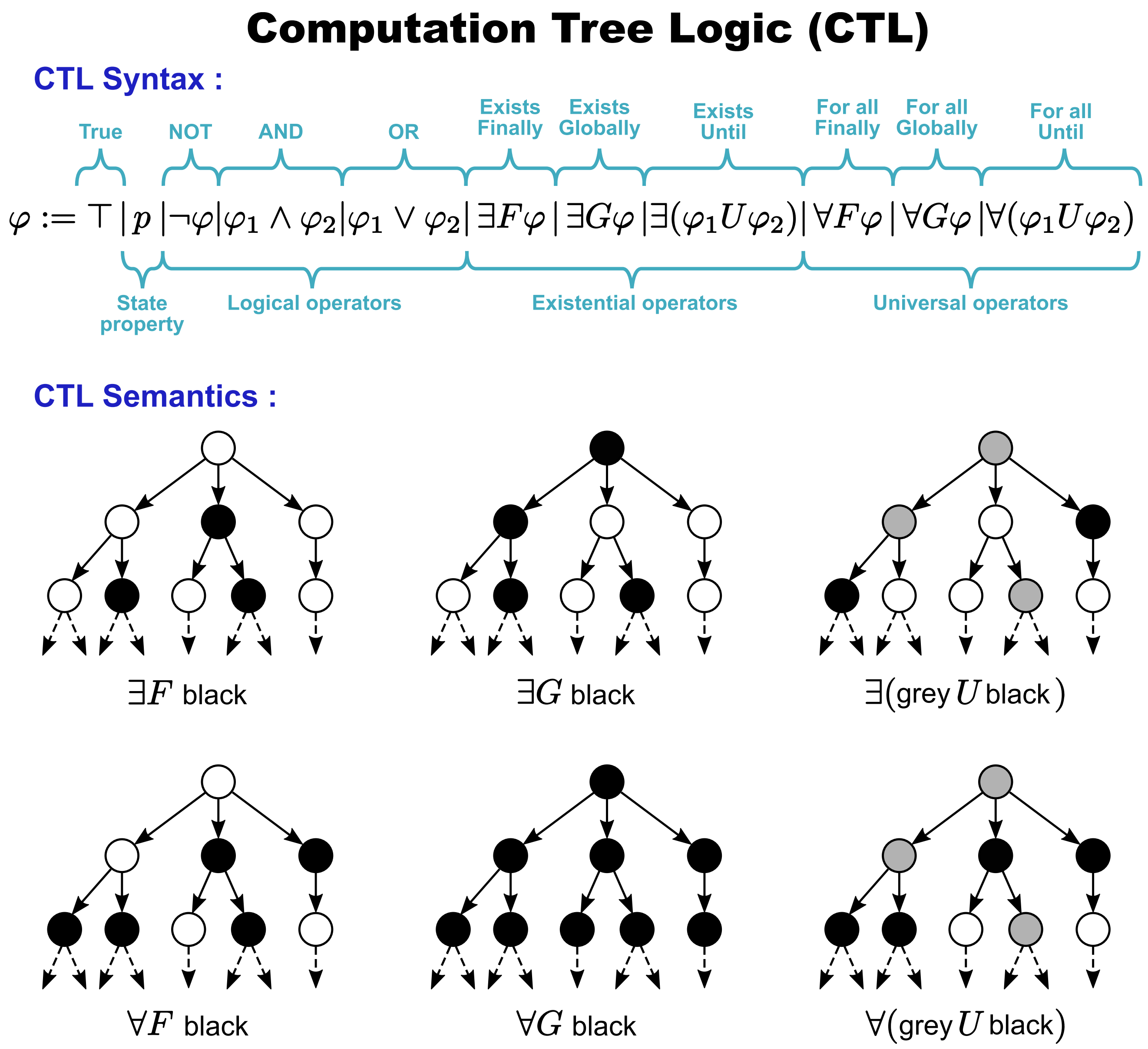

Computation Tree Logic (CTL)
CTL Syntax :
True
NOT
AND
OR
Exists Finally
Exists Globally
Exists Until
For all Finally
For all Globally
For all Until
$\varphi := \top \,|\, p \,|\, \neg\varphi \,|\, \varphi_1 \wedge \varphi_2 \,|\, \varphi_1 \vee \varphi_2 \,|\, \exists F\varphi \,|\, \exists G\varphi \,|\, \exists(\varphi_1 U \varphi_2) \,|\, \forall F\varphi \,|\, \forall G\varphi \,|\, \forall(\varphi_1 U \varphi_2)$
State property
Logical operators
Existential operators
Universal operators
CTL Semantics :
$\exists F$ black
$\exists G$ black
$\exists$(grey U black)
$\forall F$ black
$\forall G$ black
$\forall$(grey U black)

Fig 5. Syntax and semantics of Computation Tree Logic (CTL). The syntax defines how state properties and operators (logical, existential or universal) can be combined into a formula. The semantics describes the meaning of formulas. The semantics presented here is intuitive and given through example CTs satisfying basic CTL formulas. (Adapted from [47].) See [2] for a formal semantics of CTL.

https://doi.org/10.1371/journal.pcbi.1009657.g005

- the CTL formula $\exists G$ `Tr+` specifies that trees are always present along at least one branch of the CT, which is satisfied for the left-most branch in Fig 4a but not for its other branches, thus this CTL property is satisfied in both Fig 4a and 4b;
- the CTL formula $\forall G$ `Tr+` specifies that trees are always present all along every branch of the CT, which is satisfied in Fig 4b, but not in Fig 4a.

Lastly, CTL operators can be nested to express even subtler temporal behaviour. For example, $\forall G(\exists F$ `Tr-`$)$ specifies that: all along every pathway ($\forall G$), the pathway can always branch off to reach a future state ($\exists F$) without trees (`Tr-`). While $\forall G(\exists F$ `Tr-`$)$ holds in Fig 4a, the simpler property $\forall F$ `Tr-` does not because trees never disappear in the left-most branch of the CT.

Translating a dynamical property (i.e. a description of an ecosystem behaviour) written in English into a CTL formula can turn out to be a delicate exercise for non-expert users [48]. One possible way to simplify this task is to provide users with a catalogue mapping query patterns to their translations in CTL (Table 2) [19, 49].

Note that we have illustrated the semantics of CTL by evaluating formulas with respect to a single root state and its CT. More generally, the output of a CTL model-checker is the *set* of all the states of the STG whose CT satisfies the formula. This amounts in theory to consider every state of the STG as the root of a CT, and to evaluate the formula for each CT. In practice, CTL model-checkers use much more efficient techniques to obtain this result and avoid this nested enumeration.

1.2.2 Implementation of a scalable model-checker. We instantiated the general model-checking methodology from Fig 3 (in black) as indicated by the blue italic annotations, and implemented it within toolkit `ecco`. `ecco` is a Python [50] library intended to be used within Jupyter notebooks [51] and aimed at providing tools for the formal modelling and analysis of ecosystems. `ecco` has been developed and used for years to model and analyse varied ecosystems [39, 44, 52–56]. `ecco` is available as a free software released under the GNU LGPL and is hosted at http://github.com/fpom/ecco. In particular, it features an implementation of the if-then rules language presented in Section 1.1.3, as well as an efficient STG structure implemented on the top of `ITS-tools` and `libDDD` [57, 58]. At its core, `ecco` allows to compute a symbolic representation of the states of an if-then model: individual states are not explicitly enumerated, but instead a compact data structure (based on *Data Decision Diagram*, *DDD* [59]) gathers sets of states from which sets of successor (or predecessor) states can be efficiently

Table 2. Catalogue mapping query patterns to their translations in CTL. Dynamical properties relevant to ecological systems are gathered into patterns. The patterns are written in English and translated into CTL formulas. *x* and *y* are place-holders for state properties. (Adapted from [49]).

English description of the pattern	CTL formula φ
Reachability pattern	
An *x* state *can* be reached	$\exists F(x)$
An *x* state *cannot* be reached	$\neg\exists F(x)$
Consequence pattern	
If an *x* state is reached, then it is *possibly* followed by an *y* state	$\forall G(x \Rightarrow \exists F(y))$
If an *x* state is reached, then it is *necessarily* followed by an *y* state	$\forall G(x \Rightarrow \forall F(y))$
Sequence pattern	
An *y* state is reachable and is *possibly* preceded *at some time* by an *x* state	$\exists F(x \wedge \exists F(y))$
An *y* state is reachable and is *possibly* preceded *all the time* by an *x* state	$\exists(xUy)$
An *y* state is reachable and is *necessarily* preceded *at some time* by an *x* state	$\exists F(y) \wedge \neg\exists(\neg xUy)$
An *y* state is reachable and is *necessarily* preceded *all the time* by an *x* state	$\exists F(y) \wedge \forall G(\neg x \Rightarrow \forall G(\neg y))$
Invariance pattern	
x states *can* persist forever	$\exists G(x)$
x states *must* persist forever	$\forall G(x)$
x states *possibly* remain forever reachable	$\exists G(\exists F(x))$
x states *necessarily* remain forever reachable	$\forall G(\exists F(x))$
x states are *necessarily* reached infinitely often	$\forall G(\forall F(x))$
Reachability & Invariance pattern	
It is *possible* to reach a state from which *x* states *can* persist forever	$\exists F(\exists G(x))$
It is *possible* to reach a state from which *x* states *must* persist forever	$\exists F(\forall G(x))$

https://doi.org/10.1371/journal.pcbi.1009657.t002 computed [60]. This symbolic approach can mitigate the *combinatorial explosion problem* (i.e. the exponential growth of the number of states with the number of variables) that is inherent to state-based approaches [61].

Our CTL model-checker, which has been integrated into `ecco`, is symbolic as well and based on the algorithm from [61]: its computes as a DDD the set of states of the STG satisfying a query formula, and a *yes/no* answer can be obtained by intersecting this set with the set of initial states. Being fully symbolic, our implementation is highly scalable and we have experimented with models up to a few billion states.

In the next Section, we use `ecco` to analyse the *Borana model* by selecting the scenarios satisfying a given CTL query, i.e. the control valuations for which the CTL formula is satisfied in the corresponding initial states. The symbolic approach enables the model-checking of a given formula at once for all the possible control valuations of the *Borana model*. By extracting only the control variables from the resulting DDD, we are able to select the scenarios satisfying a given formula. This information is then transformed into an equivalent Boolean formula that is finally transformed into a *canonical form* using SymPy [62]. This latter step has been streamlined by adding it to `ecco` as a routine directly usable on STGs.

2 Results

From the 128 initial states of the *Borana model*, one per scenario, an STG of 1185 states was computed by the cascading applications of the if-then rules. As controls cannot change during the dynamic, scenarios are out of reach from one another and form disconnected subgraphs of the STG. Thus every initial state spawned its own subgraph representing the dynamics of the system along the corresponding scenario and disconnected from the rest of the STG. The largest scenario subgraph has 26 states.

The subgraph corresponding to the scenario before livestock introduction at high altitude is given in Fig 6 as an example. This subgraph outputted by the *Borana model* can be compared to the corresponding STM (Fig 1a) by gathering the states belonging to the same vegetation classes (S2 Table). The subgraph and the STM are almost identical. The only difference is that in the subgraph, the transition from sparse scrubland to grassland is additionally labelled by "low intensity fire" and "browsing", because those events may happen in sparse scrubland before the establishment of trees.

The three available STMs [21] drawn from observations were compared to the subgraphs computed by the *Borana model* for the corresponding scenarios (see S1 Notebook). The first two STMs describing: (1) the pathways with wildlife herbivory and fire (Fig 1a), and (2) the pathways with extensive grazing and fire, were almost identical to their corresponding subgraphs, except for the additional labels mentioned above. The third STM describing intensive grazing with fire ban (Fig 1b), presents more differences from its corresponding subgraph. Yet the additional vegetation classes and most of the additional transitions in the subgraph were empirically observed [22]. Moreover, the subgraph showcases the main features of the STM: encroachment is not reversible, and open canopy woodland is not reachable from grassland.

We designed six CTL queries relevant to the management of the Borana ecosystem and covering all five pattern types introduced in Table 2. Those queries are built upon the following state properties:

- Closed canopy woodland is a vegetation class [22] modelled by the presence and absence of some plant variables (see S2 Table for an exhaustive definition of the Borana vegetation classes as state properties):

$$\texttt{ClosedCanopyWoodland} = \texttt{Gr}- \wedge \texttt{Sh}- \wedge \texttt{Tr}+ \wedge \texttt{Cr}-$$

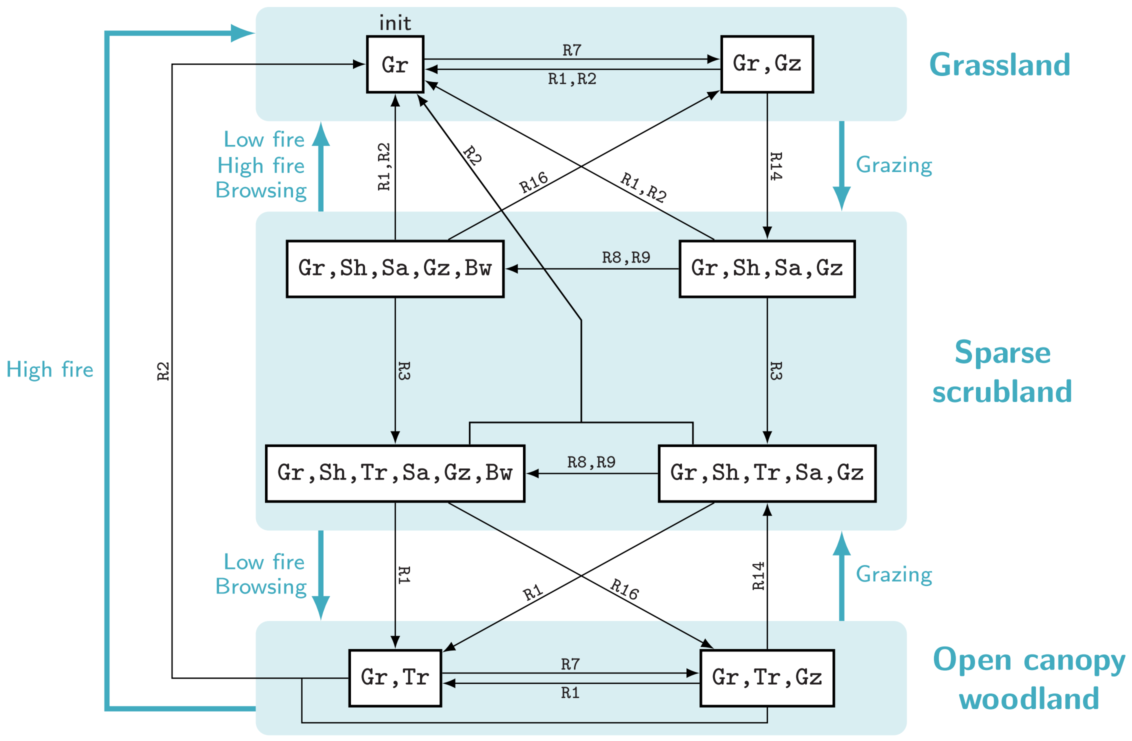


Fig 6. Scenario subgraph computed by the *Borana model*. This subgraph corresponds to the scenario with wildlife and fire at high altitude (Alt+, Fb-, Cb+, Wl+, Ps-, Ig-, BLv-), the scenario at low altitude is similar (see S1 Notebook). The states are displayed as white squares, the variables within a square represent the variables valuated "+" in this state. From the initial state (top left), the subgraph is computed by the cascading applications of the if-then rules (S1 Table). The transitions are labelled by the rules that produced them, if several rules produced the same transition then it is labelled by all of them, separated by commas. The states belonging to the same vegetation class (S2 Table) are gathered inside a blue rounded box labelled by the name of the vegetation class. Each transition from a state in a vegetation class to a state in another vegetation class, gives rise to a transition between the two classes in the same direction and labelled by the tags of the corresponding rules. For instance, transition $\{\mathtt{Gr},\mathtt{Sh},\mathtt{Sa},\mathtt{Gz},\mathtt{Bw}\} \xrightarrow{\mathtt{R1,R2}} \{\mathtt{Gr}\}$ gives rise to the transition from class "Sparse scrubland" to class "Grassland" that is labelled by "low fire" (tag of R1) and "high fire" (tag of R2). The additional tag "browsing" comes from transition $\{\mathtt{Gr},\mathtt{Sh},\mathtt{Sa},\mathtt{Gz},\mathtt{Bw}\} \xrightarrow{\mathtt{R16}} \{\mathtt{Gr},\mathtt{Gz}\}$.

https://doi.org/10.1371/journal.pcbi.1009657.g006

- Encroachment [23] corresponds to the vegetation classes with shrubs or trees but without grass nor crop (closed canopy woodland, dense scrubland and bushland):

$$\mathtt{Encroachment} = (\mathtt{Sh}+ \lor \mathtt{Tr}+) \land \mathtt{Gr}- \land \mathtt{Cr}-$$

- Subsistence production [63] corresponds to the states with crops or livestock:

$$\mathtt{Subsistence} = \mathtt{Cr}+ \lor \mathtt{Lv}+$$

We used model-checking to select the control valuations (scenarios) satisfying each of the queries (Table 3). For each query and for each scenario, the model-checker tests whether the initial state of the scenario exhibits the temporal behaviour specified by the query or not, returning a *yes/no* output. We selected the valuations of the controls for which the associated model-checking output is *yes*, the omitted controls have no impact on the model-checking output. The first query exhibits a straightforward answer to the simplest pattern: encroached states are only reachable under pastoralism with intensive grazing. The second query shows that a simple pattern can have a complex answer. The remaining queries propose a general

Table 3. Scenario selection by model-checking. For each of the six queries we show: (1) its pattern type and CTL formula, (2) its translation into English, (3) ☑ the scenario selection (i.e. control valuations) for which the associated model-checking output of the query is *yes*, (4) an English interpretation of this scenario selection.

1)	**Reachability pattern**: $\exists F$ Encroachment An encroached state can be reached. ☑ $\texttt{Ps+} \wedge \texttt{Ig+}$ *Encroachment can only happen under the scenarios encompassing pastoralism* Ps+ *with intensive grazing* Ig+.
2)	**Reachability pattern**: $\exists F$ ClosedCanopyWoodland Closed Canopy Woodland can be reached. ☑ $\texttt{Ps+} \wedge \texttt{Ig+} \wedge (\texttt{Alt+} \vee \texttt{Fb-} \vee \texttt{Wl+} \vee \texttt{BLv+})$ *Closed Canopy Woodland can only happen under pastoralism* Ps+ *with intensive grazing* Ig+ *and with at least one of the following factors: high altitude* Alt+, *no fire ban* Fb-, *presence of wildlife* Wl+, *browsing livestock* BLv+.
3)	**Reachability + Consequence pattern**: $(\exists F\ \texttt{Encroachment}) \wedge (\forall G(\texttt{Encroachment} \Rightarrow \exists F \neg \texttt{Encroachment}))$ An encroached state is reachable, and whenever it is reached it is possibly followed by an unencroached state. ☑ $\texttt{Ps+} \wedge \texttt{Ig+} \wedge \texttt{Alt+} \wedge \texttt{Cb-}$ *If an encroached state is reachable (see output* $\texttt{Ps+} \wedge \texttt{Ig+}$ *from query 1) and if the system is at high altitude* Alt+ *with crops allowed* Cb-, *then whenever an encroached state is reached it is possibly followed by an unencroached state, i.e. encroachment is reversible.*
4)	**Sequence pattern**: $\exists F(\texttt{Encroachment} \wedge \exists F \neg \texttt{Encroachment})$ An unencroached state is reachable and is possibly preceded at some time by an encroached state, i.e. at least some encroachment pathways are reversible. ☑ $\texttt{Ps+} \wedge \texttt{Ig+} \wedge (\texttt{BLv+} \vee \texttt{Wl+} \vee (\texttt{Alt+} \wedge \texttt{Cb-}))$ *If an encroached state is reachable (*$\texttt{Ps+} \wedge \texttt{Ig+}$*, see query 1), there are three set of scenarios where at least some encroachment pathways are reversible: (1) with browsing livestock* BLv+, *(2) with wildlife* Wl+, *(3) at high altitude* Alt+ *with crops allowed* Cb-.
5)	**Invariance pattern**: $\forall G(\exists F\ \texttt{Subsistence})$ Subsistence states necessarily remain forever reachable. ☑ $(\texttt{Ps+} \wedge \texttt{Ig-}) \vee (\texttt{Alt+} \wedge \texttt{Cb-} \wedge \texttt{Ps+}) \vee (\texttt{Alt+} \wedge \texttt{Cb-} \wedge \texttt{Wl+})$ *There are three sets of scenarios where subsistence remains reachable whatever happens: (1) under pastoralism* Ps+ *without intensive grazing* Ig-, *(2) at high altitude* Alt+ *with crops allowed* Cb- *and with pastoralism* Ps+, *or (3) at high altitude* Alt+ *with crops allowed* Cb- *and with wildlife* Wl+.
6)	**Reachability & Invariance pattern**: $\exists F(\exists G\ \texttt{Subsistence})$ It is possible to reach a state from which the subsistence can persist forever. ☑ $(\texttt{Ps+} \wedge \texttt{BLv+}) \vee (\texttt{Alt-} \wedge \texttt{Ps+}) \vee (\texttt{Fb+} \wedge \texttt{Cb+} \wedge \texttt{Ps+} \wedge \texttt{Ig-})$ *There are three sets of scenarios where it is possible to reach a state from which subsistence can persist forever: (1) under pastoralism* Ps+ *with browsing livestock* BLv+, *(2) at low altitude* Alt- *with pastoralism* Ps+, *or (3) with fire banned* Fb+ *as well as crops* Cb+ *and with pastoralism* Ps+ *but without intensive grazing* Ig-.

https://doi.org/10.1371/journal.pcbi.1009657.t003

survey of the patterns (Table 2) with answers of various complexity. Computing all the model-checking results (S1 Notebook) took only a few seconds on a modern laptop (Linux 5.4 Mint/Ubuntu, 32G RAM, CPU Intel Core i7–7820HQ 2.9GHz).

The scenario selection (Table 3) exhibits how the model-checking methodology could help better understand the Borana vegetation pathways and choose adequate management policies. The first two queries select the scenarios enabling bush encroachment. The answer to the first query shows that intensive grazing is the necessary condition for encroachment. This may seem counter-intuitive because fire has a strong influence on bush encroachment, yet bush encroachment has continued in Borana despite the lift of the fire ban in the 2000s [22, 23]. The answer to the second query shows that at least one of the following controls is additionally needed in order to reach closed canopy woodland: Alt+, Fb-, Wl+, or BLv+. Each of those controls enables one of the rules removing shrubs without changing grasses nor trees (R5, R1, R16, or R17 respectively, see S1 Table). Thus when combined with intensive grazing (R15), grasses and shrubs are removed without removing trees, resulting in closed canopy woodland.

The third and fourth queries select the scenarios making bush encroachment reversible. The third query selects the scenarios where encroachment is always reversible (from any encroached state, there is a pathway toward an unencroached state). The answer to the third query shows that crop cultivation at high altitude ($\texttt{Alt+} \wedge \texttt{Cb-}$ corresponding to R18, R19) is the only management policy making bush encroachment always reversible. Although this phenomenon has been observed [22], it is thought to be unfeasible at a large scale in the long term [23] due to the cost of the required inputs and the tensions between crop and livestock agriculturists. The fourth query selects the scenarios where at least some encroachment pathways are reversible (i.e. from some encroached states, there is a pathway toward an unencroached state). The answer to the fourth query shows that in addition to crop cultivation at high altitude (`Alt+` ∧ `Cb-`), two management policies make some encroachment pathways reversible: the presence of wildlife `Wl+` and browsing livestock `BLv+`. Indeed pastoralists in Borana have increased their holding of browsing livestock (goats and camels) in order to mitigate bush encroachment [21, 23].

The fifth and sixth queries select the scenarios enabling subsistence. The fifth query selects the scenarios resulting in chronic subsistence (food is not constantly but only recurrently reachable). The answer to the fifth query shows that three management policies result in chronic subsistence: (1) extensive pastoralism (`Ps+` ∧ `Ig-`), (2) pastoralism with crop cultivation at high altitude (`Alt+` ∧ `Cb-` ∧ `Ps+`), and (3) crop cultivation with wildlife at high altitude (`Alt+` ∧ `Cb-` ∧ `Wl+`). The first management policy corresponds to the traditional management policy in the Borana zone (nomadic pastoralism [23]), while the second policy corresponds to one of the current management policies (mixed crop-livestock systems [23]), the third management policy correspond to crop cultivation with fallow periods (which is thought to be unfeasible in the long term in drylands [23]). The sixth query selects the scenarios enabling continuous subsistence (there is a pathway along which food is constantly available). The answer to the sixth query shows that three management policies enable continuous subsistence: (1) pastoralism with browsing livestock (`Ps+` ∧ `BLv+`), (2) pastoralism at low altitude (`Alt-` ∧ `Ps+`), and (3) extensive pastoralism without crop nor fire (`Fb+` ∧ `Cb+` ∧ `Ps+` ∧ `Ig-`). This last result should be cautiously considered as continuous subsistence may be restricted to a single pathway, yet uncontrolled events may prevent humans to fully enforce this desired pathway in a real system.

The experiments on the *Borana model* illustrate the insights that model-checking can provide to ecology. Model-checking was used to validate the model by comparing its properties to empirical observations. In addition, model-checking provided a prospective analysis of the temporal behaviour of the system by foreseeing the cascading effects of management policies in order to select the ones making bush encroachment reversible or enabling subsistence.

3 Discussion

3.1 Model-checking ecological state-transition graphs

Model-checking performs efficient and automated analysis of the temporal behaviour of ecological STGs, answering a recently expressed interest in such tools [8, 9]. Given an STG and a temporal behaviour expressed as a temporal logic formula, a model-checker returns a *yes/no* output depending on whether the STG displays the behaviour or not. Model-checking is a multipurpose tool that can be used both to investigate the temporal behaviour of STGs (representing empirical data or resulting from modelling) and to validate models outputting STGs. Since model-checking is automated, it can process STGs that are too large to be examined by hand. For example, CTL model-checking was applied in systems biology to STG models made of hundreds of variables [64].

Even on the *Borana model* that consists of only 15 variables, answering questions like "*which scenarios result in chronic subsistence?*" (Table 3, query 5) would probably be hardly feasible without resorting to model-checking. Not only this question actually corresponds to a not-so-simple CTL property, but its answer is also surprisingly complex. A human examination may certainly detect the importance of pastoralism, crop ban and altitude, but would most likely fail to accurately relate them within a reasonable amount of time. On the other hand, model-checking is fully automated and provides the exact answer in a matter of seconds.

Human work is then limited to the design of temporal logic formulas, which is beneficial to scientific rigour and science reproducibility by removing ambiguity in definitions.

Yet, model-checking ecological STGs also has limitations. First, to unveil the full potential of model-checking, the size of the STG must be sufficiently large to exceed human abilities and to require an automated method. This is not always the case, especially for STMs that are often designed to be user friendly and to enable participatory practice. Yet, we believe that even in this participatory context, model-checking can still provide an adequate and rigorous conceptual framework for thinking about the dynamical properties of the STGs. Moreover, the relatively small size of most existing ecological STGs may be explained by the current lack of automated analysis tools, a lack that can be addressed with model-checking. Second, as with every automated method, the computing time of model-checking scales up with the size of its inputs (the size of the STG and the complexity of the formula). Yet the sizes of empirical STGs are limited, and studies in systems biology [16, 64] demonstrate that model-checking is able to deal with complex models. Lastly and most importantly, model-checking provides *yes/no* output (does the STG exhibits the queried behaviour or not), generally with a counterexample for negative output. The desired result may be more nuanced, for example in the *Borana model* experiments we used those *yes/no* outputs to select the scenarios where the model-checking output is *yes*. Thus the model-checking methodology may have to be slightly tweaked in order to derive more complex results from its *yes/no* output.

3.2 Model-checking the *Borana model*

To illustrate the model-checking of ecological STGs, we instantiated its general outline (Fig 3, in black) with a particular choice of methods and tools (Fig 3, blue italic annotations).

The system description of the *Borana model* (Fig 3, left-most half) is built upon if-then rules (Fig 2), a methodology previously proposed to model ecosystem dynamics from expert knowledge [39, 42–44]. We chose a description of the system based on events (if-then rules) because it is suited to the available data in the STM literature [6], i.e. the list of the transitions between states and their main drivers (see S2 Appendix).

In this paper, we introduced a chosen set of patterns (Table 2) and their translation into one very popular temporal logic: the Computation Tree Logic (CTL) that expresses branching properties between alternative pathways (Fig 5). We chose to represent the properties of the *Borana model* with CTL, because management actions can be represented as choices between alternative pathways. We then inputted those patterns into a model-checker in order to both validate the *Borana model* and select the scenarios achieving various management objectives (making bush encroachment reversible or enabling subsistence).

Yet this implementation of the model-checking methodology (Fig 3) is only one of the many possible implementations. For example, the system description of the *Borana model* is built upon Boolean variables, which are either present or absent. Yet in general, the model-checking methodology can be applied to any discrete-state model description computing into an STG. Thus variables can be multivalued, which is typically used in systems biology to model phenomena where a reactant regulates distinct reactions that occur at distinct thresholds. In the Borana ecosystem, fire is rare when woody plant cover is above a threshold of 40% [23, 65], thus trees could be more precisely described as multivalued rather than as Boolean: $\texttt{Tr} \in \{none, low, high\}$ corresponding respectively to 0%, $< 40\%$ and $\geq 40\%$. If-then rule modelling could be extended with multivalued variables, thus keeping the *Borana model* to Boolean variables was not a technical limitation but a modelling choice.

As with the system description, numerous possibilities exist to translate the dynamical properties of a system into formulas in one of the existing temporal logics (Fig 3, right-most half). Another very popular temporal logic is the *Linear-time Temporal Logic* (LTL) [2] that expresses complex properties about a single pathway (hence its linear representation of time) by nesting temporal modalities (the same *F*, *G*, *U* we presented for CTL). For example, LTL could be used to validate models because the available observations often consist of particular pathways [66]. We did not use LTL formulas on the *Borana model* because the existing observations of the Borana ecosystem are not linear.

3.3 Bridges between studies and model-checking

We chose to base our overview of the model-checking methodology on the very broad STG concept [25] in order to give a uniting framework that can be specified into any particular implementation. Computer science provides with a large range of modelling formalisms computing STGs [13, 14, 67], each fitted to specific features and linked to specific model-checking software. As an exhaustive inventory of those modelling formalisms would be tedious, we limit ourselves here to sketch bridges between various existing studies in ecology handling STGs and model-checking frameworks already used in biology.

First, we emphasise that a computational model, i.e. both the system description and the computation step (Fig 3), is not mandatory in the model-checking methodology. Complex STGs can be found directly inside empirical studies [4, 68], without being computed from any underlying mathematical system description. Hence model-checking is not only a tool for the analysis of mathematical models, but can also assist the automated investigation of empirical data. In order to query such empirical STGs with a model-checking software, the STGs may be encoded into a computational model such as for example if-then rules (every transition $s \rightarrow s' \in E$ would be encoded as *if s then s′*).

In ecology, STGs are often computed from interaction networks, such as differential equations [8] or Boolean networks [40, 69], with possible bridges between both [41]. Model-checking tools manipulating biological networks have been designed in the field of systems biology [67, 70], for example GINsim [71] handling Boolean networks. An example of this implementation of the model-checking methodology in systems biology is given in [17], associating a system description based on Boolean network and dynamical properties written as CTL formulas to analyse and validate a model of the mammalian cell.

When the duration of the transitions between states are available and of interest [72], they may be incorporated into the STG by labelling the transitions with their durations. Timed automata [19, 73, 74] is a modelling formalism computing such STGs, that can be implemented with the software Uppaal [75] incorporating a model-checker. A complete example of this implementation of the model-checking methodology in ecology is given in [19], associating a system description based on timed automata and dynamical properties written as Timed CTL formulas (an extension of CTL handling quantitative time) to analyse scenarios of a coral reef ecosystem.

4 Conclusion

This article promotes model-checking of the ecological state-transition graphs that are found in various fields of ecology, from state-and-transition models to assembly graphs. Given an STG and a temporal behaviour described as a temporal logic formula, a model-checker returns an automated *yes/no* answer to the question *"does the STG exhibit this behaviour?"*. In addition to the automated analysis of ecological STGs, we believe that definitions based on temporal logic would help clarify and compare the various concepts used in the related fields of ecology. Model-checking can be performed on both theoretical models and empirical data, as long as they sum up into an STG. The main limitation of model-checking is its *yes/no* output, which may need to be further processed into a more nuanced answer.

Although STGs are common in ecology, the model-checking methodology remains widely unused apart from precursory studies. Yet model-checking proved in systems biology to be a valuable automated tool for the analysis of STGs, resulting into many already available software packages. The model-checking methodology encompasses a broad range of concepts and tools, thus its implementation can be fitted to the specific features of the system under study.

The main contribution of this paper is the proposition to use model-checking to assess the temporal behaviour of the various STGs found in ecology. First, we provide an inventory of ecological STGs, from historical occurrences to modern STMs and assembly graphs. Then we give a general overview of the model-checking framework, detailing the CTL temporal logic, and provide a catalogue of dynamical patterns translated into CTL. We exemplify the insights offered by model-checking through its application to a model of the Borana vegetation pathways in order to select management scenarios. Lastly, we sketch bridges between existing studies in ecology and available model-checking frameworks.

Supporting information

S1 Appendix. If-then rule modelling.
(PDF)

S2 Appendix. Justification of the *Borana model*.
(PDF)

S1 Table. Ruleset of the *Borana model*. The 19 if-then rules describing the vegetation dynamics in Borana.
(PDF)

S2 Table. Borana vegetation classes as state properties. Borana vegetation classes [22] translated into state properties (presence or absence of vegetation variables).
(PDF)

S1 Notebook. Python notebook covering the *Borana model* analysis. Zip archive containing: (1) "`README`" is a text file explaining how to install `ecco` (Section 1.2.2), (2) "`Borana_model.rr`" is a text file containing the system description of the *Borana model* (Table 1 and S1 Table), (3) "`S1_notebook.ipynb`" is a Jupyter notebook covering the *Borana model* analysis (Section 2), (4) "`S1_notebook.html`" is a static HTML preview of this notebook.
(ZIP)

Acknowledgments

We warmly thank Yann Thierry-Mieg for the fruitful discussions about the ITS-tools framework, as well as Chuan Liao for his kind permission to redraw his STMs.

Author Contributions

Conceptualization: Colin Thomas, Maximilien Cosme, Cédric Gaucherel, Franck Pommereau.

Formal analysis: Colin Thomas, Maximilien Cosme.

Funding acquisition: Cédric Gaucherel, Franck Pommereau.

Investigation: Colin Thomas, Maximilien Cosme.

Methodology: Colin Thomas, Maximilien Cosme, Cédric Gaucherel, Franck Pommereau.

Project administration: Cédric Gaucherel.

Resources: Cédric Gaucherel.

Software: Colin Thomas, Franck Pommereau.

Supervision: Cédric Gaucherel, Franck Pommereau.

Validation: Colin Thomas, Maximilien Cosme, Franck Pommereau.

Visualization: Colin Thomas, Franck Pommereau.

Writing – original draft: Colin Thomas.

Writing – review & editing: Maximilien Cosme, Cédric Gaucherel, Franck Pommereau.

References

1. Clements FE. Plant succession: an analysis of the development of vegetation. 242. Carnegie Institution of Washington; 1916.
2. Clarke EM, Jr, Grumberg O, Kroening D, Peled D, Veith H. Model Checking. 2nd ed. Cyber Physical Systems Series. Cambridge, MA, USA: MIT Press; 2018.
3. Chen HY, Popadiouk RV. Dynamics of North American boreal mixedwoods. Environmental Reviews. 2002; 10(3):137–166. https://doi.org/10.1139/a02-007
4. Warren PH, Law R, Weatherby AJ. Mapping the Assembly of Protist Communities in Microcosms. Ecology. 2003; 84(4):1001–1011. https://doi.org/10.1890/0012-9658(2003)084%5B1001:MTAOPC%5D2.0.CO;2
5. Westoby M, Walker B, Noy-Meir I. Opportunistic management for rangelands not at equilibrium. Journal of Range Management. 1989; 42(4):266–274. https://doi.org/10.2307/3899492
6. Bestelmeyer BT, Ash A, Brown JR, Densambuu B, Fernández-Giménez M, Johanson J, et al. State and Transition Models: Theory, Applications, and Challenges. In: Briske DD, editor. Rangeland Systems: Processes, Management and Challenges. Springer Series on Environmental Management. Cham: Springer International Publishing; 2017. p. 303–345. Available from: https://doi.org/10.1007/978-3-319-46709-2_9.
7. Hang-Kwang L, Pimm SL. The Assembly of Ecological Communities: A Minimalist Approach. Journal of Animal Ecology. 1993; 62(4):749–765. https://doi.org/10.2307/5394
8. Serván CA, Allesina S. Tractable models of ecological assembly. Ecology Letters. 2021; 24(5):1029–1037. https://doi.org/10.1111/ele.13702 PMID: 33773006
9. Walker B, Westoby M. Past, present and future of state and transition language. Rangeland Journal. 2020; 42(1):71–72. https://doi.org/10.1071/RJ20020
10. Schröter C, Schwoon S, Esparza J. The Model-Checking Kit. In: van der Aalst WMP, Best E, editors. Applications and Theory of Petri Nets 2003. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer; 2003. p. 463–472.
11. Kordon F, Bouvier P, Garavel H, Hillah LM, Hulin-Hubard F, Amat N, et al. Complete Results for the 2020 Edition of the Model Checking Contest; 2021. Available from: http://mcc.lip6.fr/2021/results.php.
12. Wang RS, Saadatpour A, Albert R. Boolean modeling in systems biology: an overview of methodology and applications. Physical Biology. 2012; 9(5):055001. https://doi.org/10.1088/1478-3975/9/5/055001 PMID: 23011283
13. Brim L, Češka M, Šafránek D. Model Checking of Biological Systems. In: Bernardo M, de Vink E, Di Pierro A, Wiklicky H, editors. Formal Methods for Dynamical Systems. SFM 2013. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer; 2013. p. 63–112. Available from: https://doi.org/10.1007/978-3-642-38874-3_3.
14. Bartocci E, Lió P. Computational Modeling, Formal Analysis, and Tools for Systems Biology. PLOS Computational Biology. 2016; 12(1):e1004591. https://doi.org/10.1371/journal.pcbi.1004591 PMID: 26795950
15. Batt G, Ropers D, de Jong H, Geiselmann J, Mateescu R, Page M, et al. Validation of qualitative models of genetic regulatory networks by model checking: analysis of the nutritional stress response in Escherichia coli. Bioinformatics. 2005; 21(suppl_1):i19–i28. https://doi.org/10.1093/bioinformatics/bti1048 PMID: 15961457

16. Abou-Jaoudé W, Monteiro PT, Naldi A, Grandclaudon M, Soumelis V, Chaouiya C, et al. Model Checking to Assess T-Helper Cell Plasticity. Frontiers in Bioengineering and Biotechnology. 2015; 2:86.

17. Traynard P, Fauré A, Fages F, Thieffry D. Logical model specification aided by model-checking techniques: application to the mammalian cell cycle regulation. Bioinformatics. 2016; 32(17):i772–i780. https://doi.org/10.1093/bioinformatics/btw457 PMID: 27587700

18. Béal J, Pantolini L, Noël V, Barillot E, Calzone L. Personalized logical models to investigate cancer response to BRAF treatments in melanomas and colorectal cancers. PLOS Computational Biology. 2021; 17(1):e1007900. https://doi.org/10.1371/journal.pcbi.1007900

19. Largouët C, Cordier MO, Bozec YM, Zhao Y, Fontenelle G. Use of timed automata and model-checking to explore scenarios on ecosystem models. Environmental Modelling & Software. 2012; 30:123–138. https://doi.org/10.1016/j.envsoft.2011.08.005

20. Cordier MO, Largouët C, Zhao Y. Model-Checking an Ecosystem Model for Decision-Aid. In: 2014 IEEE 26th International Conference on Tools with Artificial Intelligence; 2014. p. 539–543.

21. Liao C, Clark PE. Rangeland vegetation diversity and transition pathways under indigenous pastoralist management regimes in southern Ethiopia. Agriculture, Ecosystems & Environment. 2018; 252:105–113. https://doi.org/10.1016/j.agee.2017.10.009

22. Liao C, Clark PE, DeGloria SD. Bush encroachment dynamics and rangeland management implications in southern Ethiopia. Ecology and Evolution. 2018; 8(23):11694–11703. https://doi.org/10.1002/ece3.4621 PMID: 30598767

23. Liao C, Agrawal A, Clark PE, Levin SA, Rubenstein DI. Landscape sustainability science in the drylands: mobility, rangelands and livelihoods. Landscape Ecology. 2020; 35(11):2433–2447. https://doi.org/10.1007/s10980-020-01068-8

24. Liao C. Complexity In The Open Grazing System: Rangeland Ecology, Pastoral Mobility And Ethnobotanical Knowledge In Borana, Ethiopia [PhD Thesis]. Cornell University; 2016. Available from: https://hdl.handle.net/1813/43578.

25. Abou-Jaoudé W, Traynard P, Monteiro PT, Saez-Rodriguez J, Helikar T, Thieffry D, et al. Logical Modeling and Dynamical Analysis of Cellular Networks. Frontiers in Genetics. 2016; 7. https://doi.org/10.3389/fgene.2016.00094 PMID: 27303434

26. Young TP, Chase JM, Huddleston RT. Community succession and assembly comparing, contrasting and combining paradigms in the context of ecological restoration. Ecological Restoration. 2001; 19 (1):5–18. https://doi.org/10.3368/er.19.1.5

27. Chang C, HilleRisLambers J. Integrating succession and community assembly perspectives. F1000Research. 2016; 5:F1000 Faculty Rev–2294. https://doi.org/10.12688/f1000research.8973.1 PMID: 27785355

28. May RM. Network structure and the biology of populations. Trends in Ecology & Evolution. 2006; 21 (7):394–399. https://doi.org/10.1016/j.tree.2006.03.013 PMID: 16815438

29. Pilosof S, Porter MA, Pascual M, Kéfi S. The multilayer nature of ecological networks. Nature Ecology & Evolution. 2017; 1(4):1–9.

30. Prach K, Walker LR. Four opportunities for studies of ecological succession. Trends in Ecology & Evolution. 2011; 26(3):119–123. https://doi.org/10.1016/j.tree.2010.12.007 PMID: 21295370

31. Patten BC. A primer for ecological modeling and simulation with analog and digital computers. In: Systems Analysis and Simulation in Ecology (ed. Patten BC) vol. I. vol. 1. Academic Press; 1971. p. 3–121.

32. Londo G. Successive mapping of dune slack vegetation. Vegetatio. 1974; 29(1):51–61. https://doi.org/10.1007/BF02390895

33. Bergeron Y, Chen HYH, Kenkel NC, Leduc AL, Macdonald SE. Boreal mixedwood stand dynamics: ecological processes underlying multiple pathways. Forestry Chronicle. 2014; 90(2):202–213. https://doi.org/10.5558/tfc2014-039

34. Caudle D. Interagency ecological site handbook for rangelands. US Department of the Interior, Bureau of Land Management; 2013. Available from: https://jornada.nmsu.edu/sites/jornada.nmsu.edu/files/InteragencyEcolSiteHandbook.pdf.

35. Bestelmeyer BT, Williamson JC, Talbot CJ, Cates GW, Duniway MC, Brown JR. Improving the Effectiveness of Ecological Site Descriptions: General State-and-Transition Models and the Ecosystem Dynamics Interpretive Tool (EDIT). Rangelands. 2016; 38(6):329–335. https://doi.org/10.1016/j.rala.2016.10.001

36. Phillips JD, Van Dyke C. State-and-transition models in geomorphology. CATENA. 2017; 153:168–181. https://doi.org/10.1016/j.catena.2017.02.009

37. Tittonell P. Assessing resilience and adaptability in agroecological transitions. Agricultural Systems. 2020; 184:102862. https://doi.org/10.1016/j.agsy.2020.102862

38. Salles P, Bredeweg B. Modelling population and community dynamics with qualitative reasoning. Ecological Modelling. 2006; 195(1):114–128. https://doi.org/10.1016/j.ecolmodel.2005.11.014

39. Mao Z, Centanni J, Pommereau F, Stokes A, Gaucherel C. Maintaining biodiversity promotes the multifunctionality of social-ecological systems: holistic modelling of a mountain system. Ecosystem Services. 2021; 47:101220. https://doi.org/10.1016/j.ecoser.2020.101220

40. Campbell C, Yang S, Albert R, Shea K. A network model for plant–pollinator community assembly. Proceedings of the National Academy of Sciences. 2011; 108(1):197–202. https://doi.org/10.1073/pnas.1008204108 PMID: 21173234

41. Robeva R, Murrugarra D. The spruce budworm and forest: a qualitative comparison of ODE and Boolean models. Letters in Biomathematics. 2016; 3(1):75–92. https://doi.org/10.30707/LiB3.1Robeva

42. Rykiel EJ. Artificial intelligence and expert systems in ecology and natural resource management. Ecological Modelling. 1989; 46(1):3–8. https://doi.org/10.1016/0304-3800(89)90066-5

43. Starfield AM. Qualitative, Rule-Based Modeling. BioScience. 1990; 40(8):601–604. https://doi.org/10.2307/1311300

44. Gaucherel C, Pommereau F. Using discrete systems to exhaustively characterize the dynamics of an integrated ecosystem. Methods in Ecology and Evolution. 2019; 10(9):1615–1627. https://doi.org/10.1111/2041-210X.13242

45. Clarke EM, Henzinger TA, Veith H. Introduction to Model Checking. In: Handbook of Model Checking. Cham: Springer International Publishing; 2018. p. 1–26. Available from: https://doi.org/10.1007/978-3-319-10575-8_1.

46. Phillips JD. The structure of ecological state transitions: Amplification, synchronization, and constraints in responses to environmental change. Ecological Complexity. 2011; 8(4):336–346. https://doi.org/10.1016/j.ecocom.2011.07.004

47. Baier C, Katoen JP. Principles of Model Checking. Cambridge, MA, USA: MIT Press; 2008.

48. Dwyer MB, Avrunin GS, Corbett JC. Patterns in property specifications for finite-state verification. In: Proceedings of the 21st international conference on Software engineering. ICSE'99. New York, NY, USA: Association for Computing Machinery; 1999. p. 411–420. Available from: https://doi.org/10.1145/302405.302672.

49. Monteiro PT, Ropers D, Mateescu R, Freitas AT, de Jong H. Temporal logic patterns for querying dynamic models of cellular interaction networks. Bioinformatics. 2008; 24(16):i227–i233. https://doi.org/10.1093/bioinformatics/btn275 PMID: 18689830

50. Python Software Foundation. The Python language;. Available from: http://www.python.org.

51. Perkel JM. Why Jupyter is data scientists' computational notebook of choice. Nature. 2018; 563(7732):145–147. https://doi.org/10.1038/d41586-018-07196-1 PMID: 30375502

52. Gaucherel C, Pommereau F, Hély C. Understanding Ecosystem Complexity via Application of a Process-Based State Space rather than a Potential Surface. Complexity. 2020; 2020:e7163920. https://doi.org/10.1155/2020/7163920

53. Di Giusto C, Gaucherel C, Klaudel H, Pommereau F. Analysis of Discrete Models for Ecosystem Ecology. In: Roque A, Tomczyk A, De Maria E, Putze F, Moucek R, Fred A, et al., editors. Biomedical Engineering Systems and Technologies. Cham: Springer International Publishing; 2020. p. 242–264.

54. Gaucherel C, Carpentier C, Geijzendorffer IR, Noûs C, Pommereau F. Discrete-event models for conservation assessment of integrated ecosystems. Ecological Informatics. 2021; 61:101205. https://doi.org/10.1016/j.ecoinf.2020.101205

55. Cosme M, Hély C, Pommereau F, Pasquariello P, Tiberi C, Treydte A, et al. Qualitative Modeling for Bridging Expert-Knowledge and Social-Ecological Dynamics of an East African Savanna. Land. 2022; 11(1):42. https://doi.org/10.3390/land11010042

56. Pommereau F, Thomas C, Gaucherel C. Petri nets semantics of Reaction Rules (RR). In: Proc. of PETRINETS'22. vol. to appear of LNCS. Springer; 2022.

57. Thierry-Mieg Y. Homepage of ITS-tools;. Available from: http://lip6.github.io/ITSTools-web.

58. Thierry-Mieg Y. Symbolic Model-Checking Using ITS-Tools. In: Baier C, Tinelli C, editors. Tools and Algorithms for the Construction and Analysis of Systems. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer; 2015. p. 231–237.

59. Couvreur JM, Encrenaz E, Paviot-Adet E, Poitrenaud D, Wacrenier PA. Data Decision Diagrams for Petri Net Analysis. In: Esparza J, Lakos C, editors. Application and Theory of Petri Nets 2002. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer; 2002. p. 101–120.

60. Bryant RE. Binary Decision Diagrams. In: Clarke EM, Henzinger TA, Veith H, Bloem R, editors. Handbook of Model Checking. Cham: Springer International Publishing; 2018. p. 191–217. Available from: https://doi.org/10.1007/978-3-319-10575-8_7.

61. Burch JR, Clarke EM, McMillan KL, Dill DL, Hwang LJ. Symbolic model checking: 1020 States and beyond. Information and Computation. 1992; 98(2):142–170. https://doi.org/10.1016/0890-5401(92)90017-A

62. Meurer A, Smith CP, Paprocki M, Čertík O, Kirpichev SB, Rocklin M, et al. SymPy: symbolic computing in Python. PeerJ Computer Science. 2017; 3:e103. https://doi.org/10.7717/peerj-cs.103

63. Wharton CR. Subsistence Agriculture: Concepts and Scope. In: Subsistence Agriculture & Economic Development. Routledge; 1969. p. 12–20. Available from: https://doi.org/10.4324/9781315130408.

64. Chabrier N, Fages F. Symbolic Model Checking of Biochemical Networks. In: Priami C, editor. Computational Methods in Systems Biology. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer; 2003. p. 149–162.

65. Archibald S, Roy DP, Van WILGEN BW, Scholes RJ. What limits fire? An examination of drivers of burnt area in Southern Africa. Global Change Biology. 2009; 15(3):613–630. https://doi.org/10.1111/j.1365-2486.2008.01754.x

66. Mateus D, Gallois JP, Comet JP, Le Gall P. Symbolic modeling of genetic regulatory networks. Journal of Bioinformatics and Computational Biology. 2007; 05(02b):627–640. https://doi.org/10.1142/S0219720007002850 PMID: 17636866

67. Naldi A, Monteiro PT, Müssel C, the Consortium for Logical Models and Tools, Kestler HA, Thieffry D, et al. Cooperative development of logical modelling standards and tools with CoLoMoTo. Bioinformatics. 2015; 31(7):1154–1159. https://doi.org/10.1093/bioinformatics/btv013

68. Barrio IC, Hik DS, Thórsson J, Svavarsdóttir K, Marteinsdóttir B, Jónsdóttir IS. The sheep in wolf's clothing? Recognizing threats for land degradation in Iceland using state-and-transition models. Land Degradation & Development. 2018; 29(6):1714–1725. https://doi.org/10.1002/ldr.2978

69. LaBar T, Campbell C, Yang S, Albert R, Shea K. Global versus local extinction in a network model of plant–pollinator communities. Theoretical Ecology. 2013; 6(4):495–503. https://doi.org/10.1007/s12080-013-0182-8

70. Carrillo M, Góngora PA, Rosenblueth D. An overview of existing modeling tools making use of model checking in the analysis of biochemical networks. Frontiers in Plant Science. 2012; 3:155. https://doi.org/10.3389/fpls.2012.00155 PMID: 22833747

71. Chaouiya C, Naldi A, Thieffry D. Logical Modelling of Gene Regulatory Networks with GINsim. In: van Helden J, Toussaint A, Thieffry D, editors. Bacterial Molecular Networks: Methods and Protocols. Methods in Molecular Biology. New York, NY: Springer; 2012. p. 463–479. Available from: https://doi.org/10.1007/978-1-61779-361-5_23.

72. Cattelino PJ, Noble IR, Slatyer RO, Kessell SR. Predicting the multiple pathways of plant succession. Environmental Management. 1979; 3(1):41–50. https://doi.org/10.1007/BF01867067

73. Pettersson P. Modelling and Verification of Real-Time Systems Using Timed Automata: Theory and Practice [PhD Thesis]. Uppsala University, Department of Computer Systems; 1999.

74. Alur R, Dill DL. A theory of timed automata. Theoretical Computer Science. 1994; 126(2):183–235. https://doi.org/10.1016/0304-3975(94)90010-8

75. Behrmann G, David A, Larsen KG. A Tutorial on Uppaal. In: Bernardo M, Corradini F, editors. Formal Methods for the Design of Real-Time Systems: International School on Formal Methods for the Design of Computer, Communication, and Software Systems, Bertinora, Italy, September 13-18, 2004, Revised Lectures. Berlin, Heidelberg: Springer; 2004. p. 200–236. Available from: https://doi.org/10.1007/978-3-540-30080-9_7.